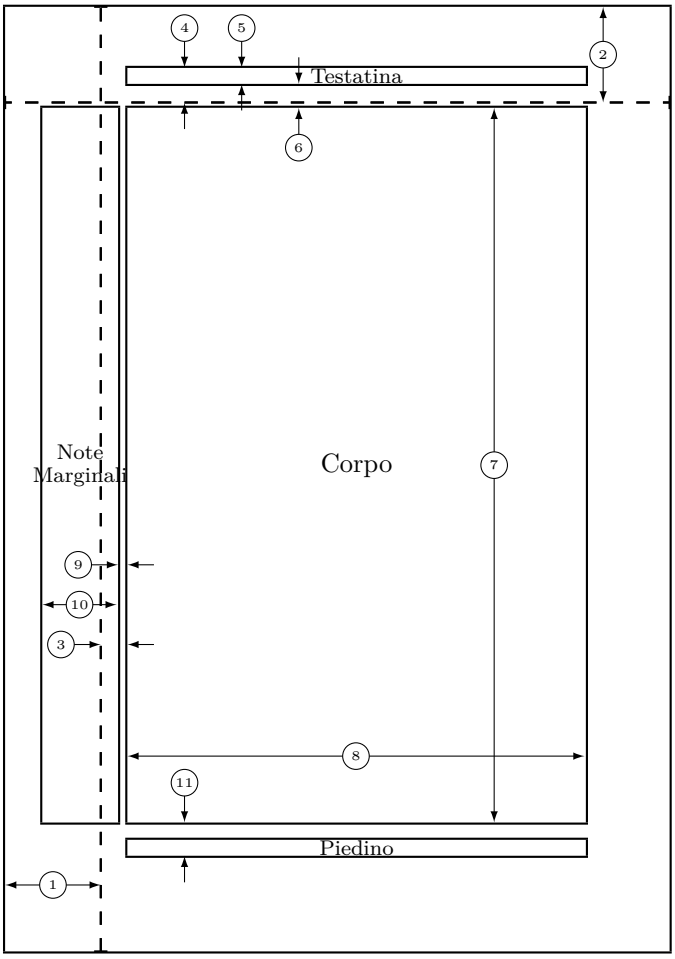
Gruppo Utilizzatori Italiani di TEX

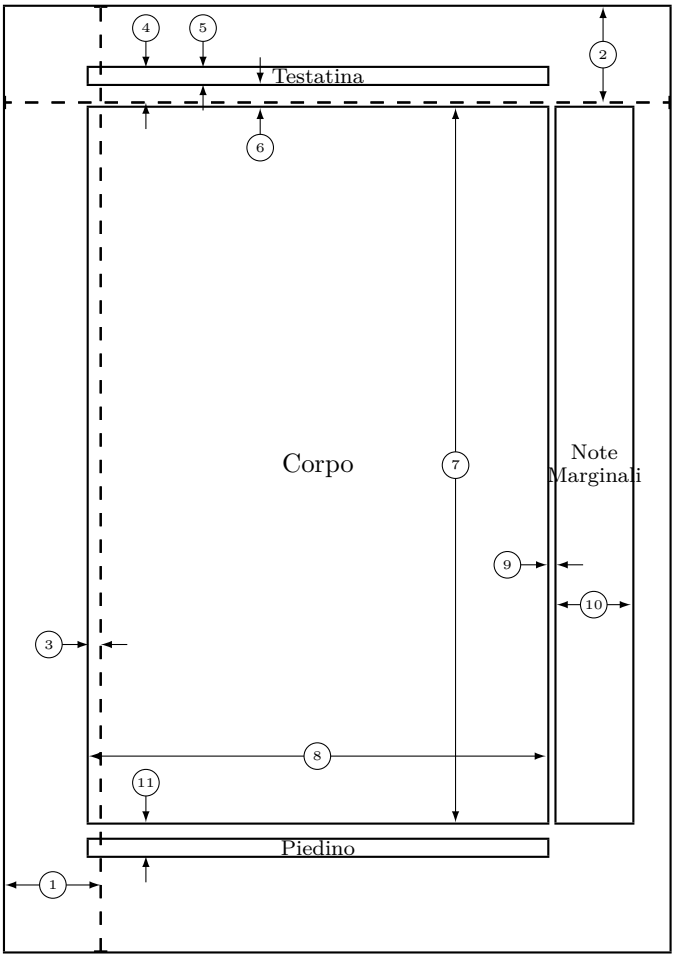
Introduzione all'arte della composizione tipografica con LATEX

Versione B5-0.99.50 — 2021-04-05
Pdf minor version = 5

Introduzione all'arte della composizione tipografica con LaTeX



1	un pollice + \hoffset	2	un pollice + \voffset
3	\evensidemargin = 20pt	4	\topmargin = -26pt
5	\headheight = 12pt	6	\headsep = 18pt
7	\textheight = 538pt	8	\textwidth = 345pt
9	\marginparsep = 7pt	10	\marginparwidth = 57pt
11	\footskip = 25pt		\marginparpush = 5pt (non mostrato)
	\hoffset = 0pt		\voffset = 0pt
	\paperwidth = 500pt		\paperheight = 711pt



1	un pollice + \hoffset	2	un pollice + \voffset
3	\oddsidemargin = -9pt	4	\topmargin = -26pt
5	\headheight = 12pt	6	\headsep = 18pt
7	\textheight = 538pt	8	\textwidth = 345pt
9	\marginparsep = 7pt	10	\marginparwidth = 57pt
11	\footskip = 25pt		\marginparpush = 5pt (non mostrato)
	\hoffset = 0pt		\voffset = 0pt
	\paperwidth = 500pt		\paperheight = 711pt

Introduzione all'arte della composizione tipografica con LaTeX

Volume II

GuIt

Versione B5-0.99.50 — 2021-04-05
Pdf minor version = 5

Quest'opera è soggetta alla Creative Commons Public License versione 2.5 o posteriore. L'enunciato integrale della Licenza in versione 2.5 è reperibile all'indirizzo internet `http://creativecommons.org/licenses/by-nc-nd/3.0/deed.it`.

- Si è liberi di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire e recitare quest'opera alle seguenti condizioni:

  Attribuzione Bisogna attribuire la paternità dell'opera nei modi indicati dall'autore o da colui al quale è stata data quest'opera in licenza; in questo caso si tratta del Gruppo degli Utilizzatori Italiani di TEX, GUIT.

  Non commerciale Non si può usare quest'opera per fini commerciali.

  Non opere derivate Non si può alterare o trasformare quest'opera, né usarla per crearne un'altra.

- Ogni volta che si usa o si distribuisce quest'opera, lo si deve fare secondo i termini di questa licenza, che va comunicata con chiarezza.

- In ogni caso si possono concordare con il titolare dei diritti d'autore (il GUIT, in questo caso) usi di quest'opera in deroga da questa licenza.

I nomi commerciali, i loghi, i trademark appartengono ai rispettivi proprietari.

La foto 2.2 della pagina 48 e le foto delle pagine 51, 272, e 499 — 501 sono di Claudio Beccari.
L'immagine 2.1 della pagina 48 è stata ricavata da un indirizzo Internet non più reperibile; l'autore sembra essere Hermann Zapf.

Coordinatore: Claudio Beccari

Hanno collaborato a questo testo in modo diretto o indiretto: Alessandro Andretta, Luciano Battaia, Francesco Biccari, Paolo Biffis, Riccardo Campana, David Carlisle, Antonio Cervone, Gustavo Cevolani, Agostino De Marco, Roberto Giacomelli, Tommaso Gordini, Enrico Gregorio, Orlando Iovino, Maurizio Himmelmann, Jerónimo Leal, Antonio Macrì, Giuseppe Molteni, Federico Morchio, Lorenzo Pantieri, Gianluca Pignalberi, Ottavio Rizzo, Salvatore Schirone, Luigi Scarso, Andrea Tonelli, Ivan Valbusa, Filippo Vomiero, Emanuele Zannarini.

Un grazie particolare a Enrico Gregorio.

Presentazione

Questo testo è rivolto a coloro che amano la bella composizione tipografica e trovano che i testi composti con LaTeX siano molto professionali. Questo implica che il lettore tipo abbia già un certa dimestichezza con LaTeX e abbia letto, almeno in parte, o il testo introduttivo [52] o, meglio ancora, l'ottimo manuale di Lorenzo Pantieri e Tommaso Gordini *L'arte di scrivere con LaTeX*, [54].

Alcune parti di questo testo possono risultare un po' ostiche a chi è completamente digiuno di programmazione, ma si è fatto ogni sforzo per rendere comprensibile queste parti anche a chi ha poca dimestichezza con il calcolatore. Certo c'è un compromesso fra chiarezza e concisione, ma si spera che il lettore vorrà ugualmente comprendere l'impegno del gruppo di persone che ha dato vita a questo testo.

Questo testo sulla composizione tipografica mediante LaTeX è stato infatti predisposto da diversi membri del Gruppo degli Utilizzatori Italiani di TeX, GuIt, il cui obiettivo è proprio quello di far conoscere il sistema di composizione sviluppato dal matematico Donald E. Knuth più di trenta anni fa. Non si tratta di un programma di interesse archeologico, perché esso è vivo e vegeto, ha dato origine a un buon numero di discendenti e alcune sue parti sono usate all'interno di altri programmi di elaborazione di testi, senza che gli utenti di questi programmi lo sappiano.

Il sistema TeX è stato uno dei primi esempi di software libero; questa sua qualità ne ha permesso il contributo creativo e/o critico di una moltitudine di utenti, come succede sempre con il software libero, per cui si è arricchito nel tempo di una moltitudine di estensioni che gli permettono di comporre praticamente qualsiasi cosa, tranne, forse, certi tipi di pieghevoli pubblicitari.

Si tratta di un programma di composizione tipografica, non di un impaginatore; quest'ultimo tipo di programmi consente di agire sul materiale da impaginare come se fosse una figura da modificare o da adattare, anche se svolge in parte le funzioni di compositore.

LaTeX, una specie di linguaggio di programmazione del sistema TeX, è, appunto, un linguaggio che va interpretato da appositi programmi del sistema TeX; la sinergia del linguaggio e dei programmi interpreti consente di comporre tipograficamente dei testi contenenti testo corrente, sia in prosa sia in poesia, scritti in qualunque alfabeto, per esempio latino, greco, cirillico, dall'andamento diretto (da sinistra a destra), oppure ebraico, arabo, dall'andamento inverso (da destra a sinistra), oppure cinese, giapponese, coreano, dall'andamento anche verticale.

Questa sinergia può gestire font di ogni genere, sia quelli a matrici di punti, sia i font PostScript, sia i font TrueType, OpenType, eccetera. Tali font possono essere codificati in varie maniere, ma alcuni 'figli' di TEX gestiscono anche i font codificati secondo la norma UNICODE.

La caratteristica che più differenzia il sistema TEX dagli altri elaboratori di composizione è il fatto che per comporre un documento con questo sistema bisogna agire in tempi diversi per introdurre il testo e per comporlo; in questo non è molto diverso da certi procedimenti professionali di impaginazione, dove il testo da comporre viene introdotto in un file di solo testo che poi viene in un secondo tempo fatto fluire dentro il programma di impaginazione, assemblandolo insieme alle figure e all'altro materiale non testuale per generare il documento 'finito', pronto da inviare alla fotoincisione e alla stampa.

Invece questo modo di comporre è molto diverso da quello dei word processor, dove il compositore vede direttamente sullo schermo del suo elaboratore il testo già composto, così da poter esaminare immediatamente il frutto del suo lavoro; l'analisi di questa differenza nel modo di procedere verrà svolta nel primo capitolo.

Il secondo capitolo darà al lettore alcune nozioni di tipografia, se non altro per abituarlo ad alcune parole che ricorrono spesso nella descrizione delle varie operazioni compositive. Chi avesse già queste nozioni può saltare la lettura di questo capitolo, ma se, nonostante tutto, decidesse di leggerlo, potrebbe constatare che alcune nozioni e/o alcuni vocaboli in questa guida sono usati per indicare cose leggermente diverse da quelle che conosceva.

Il terzo capitolo è dedicato all'ortografia tipografica; oltre alla definizione, che ovviamente non contrasta con l'ortografia della lingua, vengono impostate anche le buone regole per l'uso della cesura in fin di riga, delle forme dei caratteri, dell'uso della punteggiatura, dell'elisione e del troncamento, sottolineando che la grammatica consente numerose varianti, ma la buona tipografia predilige l'omogeneità e l'adesione a consuetudini ormai acquisite; la buona tipografia aiuta il lettore a leggere senza affaticarsi; usare convenzioni diverse dalle 'solite' affatica il lettore che deve abituarsi a comprendere uno scritto dall'aspetto per lui insolito.

Il quarto capitolo esaminerà le procedure da seguire per procurarsi il software del sistema TEX e per installare i programmi e i file accessori; non si vuole mettere in ombra la centralità dei programmi di composizione del sistema TEX, ma bisogna attrarre l'attenzione sui programmi accessori, per esempio gli *shell editor* la cui scelta può risultare determinante nella facilità con cui si può ottenere il meglio dal sistema TEX.

I capitoli cinque, sette ed otto richiamano i primi rudimenti della programmazione LATEX, che in modo asincrono attraverso i programmi interpreti diventa il testo composto. Si parlerà essenzialmente di composizione testuale, di liste e di tabelle. Queste ultime già possono dare qualche pensiero anche al lettore con una modesta esperienza alle spalle.

La programmazione della composizione mediante LATEX si distingue da altri procedimenti di composizione per consentire al compositore, oltre che allo scrittore, di concentrarsi sul messaggio da trasmettere al lettore, invece che sulla forma da dargli, cioè sulla sua estetica. A seconda del documento da comporre lo stile di composizione può essere molto diverso; anche la semplice impaginazione può assumere aspetti grafici diversi; LATEX consente di scegliere in modo globale il tipo di documento e di apportare piccole modifiche stilistiche durante la composizione,

senza che il compositore debba preoccuparsi della nerezza dei caratteri, oppure della distanza del numero indicativo di un paragrafo dall'inizio del suo titolino. A tutte queste cose pensa la scelta iniziale del tipo di documento e specificando opportune opzioni. Il capitolo sei si occupa appunto di queste cose.

I capitoli nove e dieci si dedicano alla creazione, manipolazione e inclusione di figure.

Passando ad argomenti più specializzati, i capitoli undici e dodici si occupano della preparazione della bibliografia e della composizione di indici analitici e glossari.

I capitoli tredici e quattordici si dedicano invece alla composizione della matematica; se il lettore ha familiarità con manuali di tipografia, avrà notato che l'argomento della composizione della matematica è praticamente assente da quei manuali, perché si tratta di un tipo di composizione abbastanza specializzato e che interviene raramente nei libri pubblicati. Ma un documento non è necessariamente un libro di narrativa; può essere un rapporto tecnico, una tesi di laurea o di dottorato, un manuale o un prontuario tecnico, un articolo scientifico, insomma, uno scritto dove la matematica compare, spesso, anche in forma avanzata.

Alla composizione di testi letterari e di filologia è dedicato il capitolo quindici; questi testi verosimilmente non contengono una sola formula matematica, ma richiedono un tipo di composizione spesso bidimensionale, che, come la matematica, richiede di comprendere a fondo i meccanismi compositivi bidimensionali. Alle necessità compositive della filologia si provvede mediante l'uso di moduli esterni, chiamati *file* o *pacchetti di estensione*, che fanno parte integrante del sistema TEX.

Il capitolo sedici si riferisce alla preparazione delle presentazioni, cioè di quei documenti che verranno verosimilmente usati per essere proiettati mentre se ne espone il contenuto a voce durante una conferenza, una lezione, o simili.

Il capitolo diciassette tratta dell'arte della composizione tipografica con LATEX; questo è proprio il capitolo che giustifica il titolo di questo manuale; vi si parlerà delle tecniche di cui LATEX o, più ancora, i programmi di composizione del sistema TEX dispongono per eseguire la composizione tipografica come e meglio della totalità dei word processor e meglio di molti programmi di impaginazione; si tratta della microgiustificazione, una tecnica a cui ricorrevano fin dall'inizio i prototipografi, a cominciare da Gutenberg, ma di cui si è un po' persa l'abitudine nel corso dei secoli successivi, sia per i costi, sia per i vincoli imposti dai caratteri metallici; per fortuna la composizione elettronica è svincolata da queste limitazioni, ma ovviamente bisogna che il programma di composizione sia stato predisposto per la microgiustificazione.

Il capitolo diciotto si riferisce ai caratteri da stampa e ne descrive le particolarità, i comandi per gestirli, i modi per caricare altri caratteri; fornisce qualche nozione di tipografia, ma dedica ampio spazio ai font del sistema TEX, il quale, è bene ribadirlo, non è limitato ai font disponibili in prima installazione, che sono già molto numerosi, ma può usare virtualmente qualsiasi font disponibile sia gratuito, sia commerciale.

Il capitolo diciannove è dedicato all'arte di definire nuovi comandi compositivi per agevolare il compito del compositore.

Il capitolo venti tratta invece del progetto grafico delle pagine e delle strutture che compongono il documento; normalmente queste sono decisioni e scelte operate dal grafico editoriale; con LATEX spesso siamo noi stessi i grafici editoriali, i

compositori, talvolta anche gli editori dei nostri documenti; è bene avere anche una buona formazione in questo senso, almeno fino al livello di conoscere i problemi, senza avere la presunzione di saperli risolvere come un professionista. Conoscendo i problemi, però, può nascere il desiderio di conoscere più a fondo l'arte tipografica, che ci porterà non solo ad apprezzare maggiormente i prodotti tipografici ben riusciti, ma anche ad arricchire la nostra cultura in un settore spesso trascurato o considerato erroneamente troppo tecnico per meritare il nome di 'arte'. Ricordiamo invece che il nome del sistema TEX è ottenuto dalle prime tre lettere maiuscole della parola greca τέχνη[1], che, come ricorda Knuth stesso, vuol dire *arte*.

Questo testo non è un manuale; per LATEX ce ne sono di liberi e di commerciali assai validi; a questo argomento è dedicato il capitolo ventuno. Nello stesso tempo, dicendo che questo non è un manuale si vorrebbe sottolineare che le indicazioni qui esposte servono per affrontare la composizione con LATEX con un approccio che guarda di più alla sostanza, alla composizione professionale, che non all'aspetto grafico del singolo dettaglio man mano che questo si presenta, come succede spesso, invece, con i vari word processor.

Per esempio in questo testo si parla di 'scatole' proprio di sfuggita, mentre in ogni manuale questo argomento richiede almeno una sezione espressamente dedicata loro. Così si parla poco o nulla di contatori o di lunghezze rigide o elastiche; sono argomenti importanti, ma che servono maggiormente per scrivere i programmi, le macroistruzioni, contenute nei file che specificano lo stile compositivo o nei file che raccolgono le macro personali. Durante la composizione non bisognerebbe mai perdersi in questi dettagli.

Piuttosto la programmazione in linguaggio 'nativo' TEX può diventare essenziale per l'utente che deve comporre testi dallo stile insolito o che abbia bisogno di strutture compositive particolari.

Qui, però, tranne una esposizione sommaria nel capitolo 19, non si parlerà di queste cose, ma si rinvia direttamente il lettore a manuali che trattano questo aspetto con maggiore dettaglio, (capitolo 21). Si veda piuttosto il capitolo 29, di tipo più manualistico, dove sono raccolte, commentate, modificate le informazioni che Leslie Lamport ha scritto nel suo manuale [39].

Il capitolo ventidue è dedicato ad un aspetto piuttosto moderno e fortemente legato alla tipografia elettronica, vale a dire si occupa del problema dell'archiviabilità dei documenti elettronici; questa caratteristica richiede strumenti ed accorgimenti particolari che si possono raggiungere con l'uso del sistema TEX, ma che richiedono comunque una speciale attenzione.

Il capitolo ventitré riguarda invece un aspetto produttivo che coinvolge la tipografia elettronica; si tratta del lavoro di assemblaggio dei file da comporre con il sistema TEX, quando i file iniziali non hanno il mark-up di LATEX. Si può eseguire l'elaborazione manuale, ma è meglio disporre di adeguati strumenti per la conversione automatica, almeno per ottenere un primo approccio al mark-up di LATEX, ma con la trasformazione delle codifiche dei vari caratteri in modo da non alterare il messaggio del testo da comporre e, specialmente, per non introdurre errori durante il processo di conversione. L'editoria moderna deve poter usare

[1] la parola τέχνη in Italia viene pronunciata 'tècne'; in Grecia 'téchni'; in ogni caso quella che sembra una X nel nome di TEX e di LATEX è una χ maiuscola e va pronunciata come una 'k' o come i toscani pronunciano la 'c' iniziale di 'casa', non come una 'x'.

questi strumenti con cognizione di causa, sia per il lavoro commerciale sia per quello che gradatamente si va estendendo, cioè la documentazione libera, la produzione di testi liberamente scaricabili dalla rete, ma composti con ogni cura al massimo livello qualitativo.

Il capitolo 24 è dedicato alla simbologia e alla nomenclatura delle grandezze fisiche, nonché ai simboli codificati dalle norme ISO per l'uso nella matematica usata dai fisici, dai tecnologi e in generale dagli scienziati che misurano grandezze; vi sono anche non pochi riferimenti al Sistema Internazionale delle Unità di Misura e alla loro 'ortografia'.

Il capitolo 25 spiega invece in ogni dettaglio l'algoritmo che viene usato dai programmi del sistema TEX per dividere le parole in fin di riga; questo è un procedimento che in italiano funziona impeccabilmente nella totalità dei casi, ma è bene conoscerne i dettagli per intervenire in quei pochi casi in cui la giustificazione non viene eseguita perfettamente oppure quando si scrive in una lingua straniera dalle regole più rigide di quelle che valgono per l'italiano; oppure per ricorrere ad una sillabazione italiana diversa da quella prescritta dalla norma UNI, ma consentita in casi particolari.

Il capitolo 26 vorrebbe riempire un vuoto di informazione che per usare LATEX va assolutamente colmato. Si tratta della distinzione fra la codifica di entrata e quella di uscita; fra la codifica che l'utente, tramite i suoi programmi, usa per scrivere un testo usando la tastiera, e quella che il programma di composizione usa in relazione ai font impiegati. Bisogna conoscere queste due codifiche, distinguerle e usarle con cognizione di causa.

Il capitolo 27, molto tecnico, cerca di spiegare i dettagli di come ogni motore di composizione del sistema TEX costruisca i capoversi e costruisca le pagine; si tratta di un meccanismo abbastanza complesso, ma la sua comprensione aiuta a capire perché la composizione talvolta sia diversa da come ce la aspetteremmo. Normalmente ha ragione il motore di composizione a fare quello che fa, ma talvolta può essere necessaria una piccola dose di aiuto consapevole.

Il capitolo 29, lungi dal rappresentare una semplice traduzione della corrispondente appendice del manuale [39] di Leslie Lamport, il creatore di LATEX, pur mantenendone l'impostazione, rappresenta, si spera, una fonte comoda e pratica per rivedere in modo ragionato ma non troppo prolisso la sintassi dei vari comandi del mark-up di LATEX. In questa parte, per la verità, si sono ridotte al minimo rispetto al manuale di Lamport, intere parti che oggi sono considerate superate; invece alcune altre parti sono state estese considerevolmente.

Il capitolo **??** tratta dei continui aggiornamenti di LATEX. Questo testo è andato formandosi nell'arco di diversi anni, con i contributi di molte persone, che hanno dedicato tempo e esperienza for fornire informazioni esatte e aggiornate. Tuttavia dal 2018 gli aggiornamenti del sistema TEX sono continui e molto importanti; mentre il curatore di questo testo cerca di apportare le modifiche al contenuto degli altri capitoli, in questo particolare riporta le informazioni pubblicate dal LATEX Project Team attraverso le *news letters* che quel gruppo di lavoro del TUG (TEX Users Group) pubblica in seguito all'introduzione di correzioni, modifiche, estensioni che esso apporta in continuazione. Il curatore cerca di selezionare le notizie di queste *news letters* in modo da riportare nel capitolo **??** quelle che possono interessare l'utente finale; il lettore attento, specialmente se desidera usare le novità introdotte nel costruire i suoi file di classe o i suoi pacchetti di macro personali, può leggere ogni *news letter* usando la

funzionalità della sua installazione consistente nel dare da terminale il comando `texdoc ltnews`⟨*numero*⟩ in modo da leggere l'intera *news letter* identificata da quel ⟨*numero*⟩.

Il lettore non si scoraggi se vengono usate molte parole inglesi; in un certo senso la cosa è voluta, perché i comandi del *mark-up* di LaTeX sono quasi tutti formati da parole inglesi o da loro abbreviazioni; in ogni punto in cui questi comandi vengono descritti si è sempre cercato di associare le parole inglesi alla spiegazione italiana; tuttavia non si è rifuggiti da alcune similitudini insolite, come 'flottante' (che esiste in italiano con un significato simile) corrispondente a *floating*, ritenuto preferibile rispetto all'italianissima parola 'mobile' che esprime lo stesso concetto, ma che forse è espresso con una parola slegata dai messaggi che LaTeX espone sullo schermo durante la sua esecuzione. Così si è frequentemente usata la locuzione *di default*, piuttosto che usare espressioni italiane come 'predefinito', 'da usarsi in mancanza d'altro', 'se non si è specificato diversamente', e simili locuzioni quasi equivalenti l'una all'altra.

Inoltre, sebbene non si sia sicuri di essere stati sempre consistenti, è stata usata la congiunzione oppositiva 'o' per indicare il concetto esclusivo di 'o questo o quello ma non entrambi', e la congiunzione 'oppure' per indicare il concetto inclusivo di 'o questo o quello o entrambi'. La lingua parlata e quella scritte sono flessibili e talvolta non è necessario essere così precisi, ma si cerchi di tenere presente questo particolare uso delle congiunzioni oppositive.

Di fianco a molte figure vi è una piccola legenda, spesso in verticale, con il nome dell'autore e il nome del programma, quasi sempre del sistema TeX, che l'autore ha usato. Questa indicazione non è messa vicino alle fotografie oppure alle immagini tratte da Internet; la paternità di queste immagini è indicata, ove possibile, nel retrofrontespizio. Per le altre figure, oltre a indicare l'autore del disegno mostrato, si è ritenuto utile indicare anche il programma usato per realizzare il disegno; quasi tutti i disegni sono stati realizzati usando comandi e mark-up forniti da pacchetti di estensione del programma di composizione; questi pacchetti consentono di eseguire egregiamente molti tipi di disegni, senza bisogno di ricorrere a programmi esterni; hanno anche il vantaggio di essere stilisticamente compatibili con la tipografia eseguita con LaTeX, cosa che spesso non è possibile ottenere con i normali word processor o con altri programmi di disegno assistito dal calcolatore.

Il materiale di questa guida è stato riordinato rispetto alle prime edizioni; ovviamente è anche stato aggiornato e l'aggiornamento continuo cerca di non restare indietro rispetto all'evoluzione continua del sistema TeX; non è possibile garantire che questo testo sia sempre aggiornato all'ultima distribuzione del sistema TeX, ma si fa il possibile per farlo. Si veda più avanti anche quanto scritto nell'introduzione del capitolo **??**.

Questa guida può essere composta in formato A4 (210 mm per 297 mm) o in formato B5 (176 mm per 250 mm); quest'ultimo formato può venire stampato e rifilato in modo che il volume brossurato a colla o legato possa trovare facilmente spazio nei normali scaffali; può venire anche stampato in tre tomi distinti in formato B5, dove però le pagine iniziali (dal frontespizio fino agli indici) e le pagine finali (dalla bibliografia in poi) siano comuni a tutti i tomi; i riferimenti degli indici generale e analitici saranno quindi sempre corretti anche in riferimento a tomi diversi.

Indice

Presentazione **9**

I Volume **37**

1 Composizione sincrona e asincrona **39**
1.1 Introduzione 39
1.2 Il mark-up di LaTeX 40
1.3 Scribus: un impaginatore con licenza GPL 41
1.4 Conclusione 45

2 Nozioni elementari di tipografia **47**
2.1 Tipografia e dattilografia 47
2.2 Unità di misura tipografiche 49
2.3 Misure tipografiche 51
2.4 Le particolarità dei caratteri 54
2.5 I contrografismi 56
2.6 Le parti di alcuni documenti a stampa 56
2.7 Osservazioni finali 60

3 Ortografia tipografica **61**
3.1 Ortografia testuale 61
3.1.1 Ortografie alternative 62
3.1.2 La 'd' eufonica, la 'i' prostetica e gli apostrofi 63
3.2 Accenti 64
3.3 Sillabazione 66
3.4 Punteggiatura 67
3.5 Abbreviazioni 73
3.6 Appellativi e maiuscole 75
3.7 Uso dei font 76
3.8 Le note 78
3.9 Conclusioni 78

4 Installare il sistema TEX **81**
4.1 Installazione su macchine Windows da XP in poi 82
4.1.1 Installare TEX Live 82
4.1.2 Installare MiKTEX 83
4.1.3 La visualizzazione dell'estensione dei file 84
4.2 Installazione su Linux 85
4.3 Installazione su Macintosh con MacOS X 87
4.4 Gli alberi di cartelle del sistema TEX 89
4.4.1 Gli alberi di cartelle di TEX Live 91
4.4.2 Gli alberi di cartelle di MiKTEX 93
4.4.3 L'aggiornamento dei database dei nomi dei file 94
4.5 I programmi accessori 95
4.6 Le tastiere 97
4.6.1 Le tastiere delle macchine Windows 97
4.6.2 Tastiere Linux 101
4.6.3 Le tastiere sulle macchine Macintosh 102
4.6.4 Le tastiere virtuali 103
4.7 Gli shell editor 104
4.7.1 Shell editor multipiattaforma 106
4.7.2 Shell editor per le macchine Windows 109
4.7.3 Shell editor per le macchine Linux 112
4.7.4 Shell editor per le macchine Macintosh 115
4.8 Editor quasi WYSIWYG 119
4.9 LATEX e i programmi di composizione del sistema TEX 120
4.10 Altri programmi del sistema TEX 124
4.10.1 Plain TEX 124
4.10.2 I programmi estesi 125
4.10.3 Il mark-up ConTEXt 126
4.10.4 I programmi Omega e Lambda, Aleph e Lamed 127
4.10.5 Il programma XETEX 128
4.10.6 Il programma LuaTEX 129
4.11 Il sistema TEX 130

5 LATEX: prime nozioni **135**
5.1 Introduzione 135
5.2 L'inizio del file sorgente 135
5.3 Il documento 138
5.4 La fine del documento 139
5.5 Un semplice esercizio 139
5.6 I caratteri speciali 141
5.7 Testo strutturato 142
5.8 Organizzazione dei file sorgente 144
5.9 Gestione degli errori 151

6 LaTeX: i vari tipi di documenti e stili di composizione 153
6.1 Introduzione . . . 153
6.2 Classi standard . . . 153
6.3 La creazione di nuove classi . . . 155
6.4 Alcune classi non standard . . . 156
6.4.1 Le classi Komascript . . . 156
6.4.2 La classe *memoir* . . . 158
6.4.3 La classe *ncc* . . . 159
6.4.4 Le classi per le tesi di laurea . . . 160
6.4.5 L'estensione `layaureo` . . . 162
6.5 I pacchetti di estensione . . . 164
6.5.1 Come invocare i file di estensione . . . 164
6.5.2 I vari pacchetti e gli archivi internazionali . . . 165
6.6 Come scrivere nuovi pacchetti . . . 168
6.7 Non modificare i pacchetti esistenti . . . 170

7 LaTeX: testi speciali 173
7.1 Che cosa sono i testi in display . . . 173
7.2 Le citazioni . . . 173
7.2.1 Le citazioni brevi . . . 174
7.2.2 Le citazioni lunghe . . . 176
7.2.3 I versi . . . 176
7.2.4 Brani in linguaggi speciali . . . 178
7.3 Gli elenchi . . . 178
7.3.1 Le elencazioni in linea . . . 178
7.3.2 Le enumerazioni . . . 179
7.3.3 Le elencazioni semplici . . . 180
7.3.4 Alcune osservazioni relative alle elencazioni . . . 181
7.4 Le descrizioni . . . 181
7.5 Le liste bibliografiche . . . 182
7.6 I riferimenti incrociati . . . 184
7.7 Altri testi in display . . . 185
7.8 Le note . . . 187
7.8.1 Le note in calce . . . 188
7.8.2 Le note marginali . . . 189
7.9 Un esempio specifico di testi speciali . . . 191

8 LaTeX: tabelle 195
8.1 Introduzione . . . 195
8.2 Come far flottare una tabella . . . 196
8.3 Le didascalie . . . 196
8.4 Come comporre la tabella vera e propria . . . 198
8.4.1 I descrittori delle colonne . . . 198
8.4.2 Il raggruppamento delle celle . . . 200
8.4.3 I separatori verticali . . . 201

8.4.4 Come rendere le tabelle un poco più aperte 202
8.5 Le tabelle di larghezza specificata 204
8.6 Problemi compositivi delle tabelle 207
8.6.1 Tabelle troppo larghe 207
8.7 Tabelle troppo lunghe 209
8.8 Pacchetti di estensione per le tabelle 211

9 LaTeX: figure 217
9.1 Le figure e le immagini 217
9.2 L'ambiente `figure` 217
9.2.1 Controllo dei grandi oggetti flottanti 218
9.2.2 Modifica degli ambienti flottanti 218
9.3 L'ambiente `picture` 219
9.4 Il pacchetto **pgf** 229
9.5 Vantaggi dei programmi nativi del sistema TeX 232
9.6 METAPOST 233
9.7 Usi insoliti dell'ambiente *picture* 235
9.8 Linee guida per la grafica 237
9.8.1 Preliminari 238
9.8.2 Il tempo necessario per la creazione della grafica 239
9.8.3 Piano di lavoro per creare un grafico 240
9.8.4 Collegamento fra testo e grafico 241
9.8.5 Coerenza fra testo e grafica 242
9.8.6 Legende nei grafici 243
9.8.7 Diagrammi di vario genere 244
9.8.8 Attenzione e distrazione 249
9.8.9 Commenti 251

10 LaTeX: l'importazione di figure esterne 253
10.1 Introduzione 253
10.2 I formati grafici 253
10.2.1 I formati vettoriali 253
10.2.2 I formati diversi da quelli vettoriali 254
10.3 I formati accettabili 256
10.3.1 I formati accettabili da latex 258
10.3.2 I formati accettabili da pdflatex, lualatex e xelatex 260
10.4 Conversione dei formati 262
10.5 Scontornare le immagini 264
10.6 L'importazione delle immagini 265
10.6.1 Organizzare le immagini 266
10.6.2 Includere le immagini 269

11 La bibliografia **275**
11.1 Introduzione 275
11.2 I database bibliografici 276
11.3 Programmi e pacchetti per la bibliografia 279
11.4 BibTeX 279
11.4.1 Come comporre la bibliografia 279
11.4.2 Come specificare lo stile bibliografico 280
11.4.3 Chiavi e citazioni 280
11.5 Il pacchetto biblatex 282
11.6 Quanti database bibliografici? 284

12 LaTeX: indici e glossari **287**
12.1 Introduzione 287
12.2 L'indice analitico 287
12.2.1 Il programma makeindex 288
12.2.2 La composizione effettiva dell'indice analitico 289
12.3 Il glossario 290
12.4 Modifica dell'indice analitico 291
12.5 Indicizzazione sincrona 291
12.6 Composizione automatica dell'indice analitico 293
12.7 Conclusione 296

II Volume **297**

13 LaTeX: la matematica semplice **299**
13.1 Introduzione 299
13.2 I modi matematici 299
13.3 Alcune annotazioni sulle lettere greche 302
13.4 Alcune osservazioni sugli operatori funzionali 303
13.5 Alcune osservazioni sui grandi operatori 308
13.6 I grandi delimitatori 309
13.7 Accenti e segni diacritici matematici 310
13.8 Gli ambienti matematici 311
13.9 Le unità di misura 313

14 LaTeX: la matematica avanzata **317**
14.1 I simboli di *amsmath* 317
14.2 Le estensioni dei font matematici 320
14.3 I sistemi di equazioni 320
14.4 Gli ambienti di composizione di *amsmath* 321
14.4.1 L'ambiente *equation* 322
14.4.2 L'ambiente *aligned* 322
14.4.3 L'ambiente *split* 323
14.4.4 L'ambiente *multline* 324

14.4.5 L'ambiente *gather* . . . 325
14.4.6 L'ambiente *align* . . . 325
14.4.7 L'ambiente *flalign* . . . 326
14.4.8 L'ambiente *alignat* . . . 327
14.4.9 L'ambiente *subequations* . . . 327
14.4.10 Gli ambienti spezzati . . . 328
14.5 Altri comandi e ambienti . . . 329
14.5.1 Definizione di operatori funzionali . . . 329
14.5.2 Le frazioni in generale e le frazioni continue . . . 330
14.5.2.1 Le frazioni e gli altri costrutti simili . . . 330
14.5.2.2 Le frazioni continue . . . 330
14.5.3 Il testo intercalato alle equazioni . . . 330
14.5.4 Le frecce estensibili . . . 332
14.5.5 Gli indici incolonnati . . . 332
14.5.6 Gli integrali multipli . . . 332
14.5.7 L'operatore differenziale . . . 333
14.5.8 I simboli corsivi matematici in nero . . . 334
14.5.9 Le espressioni matematiche riquadrate . . . 335
14.6 Le matrici e i determinanti . . . 335
14.7 I diagrammi commutativi . . . 338
14.8 La punteggiatura in matematica . . . 339
14.9 Conclusioni . . . 343

15 LaTeX: composizione di testi letterari e filologici 345
15.1 La composizione di testi letterari . . . 345
15.2 Scrivere in greco . . . 349
15.3 Font adatti ai testi classici . . . 353
15.4 La composizione del tedesco classico . . . 354
15.5 La composizione di testi in lingue classiche . . . 356
15.5.1 Scrivere in latino . . . 356
15.5.2 Scrivere in greco . . . 358
15.6 La composizione di testi filologici . . . 361
15.7 Un esempio di composizione con XeLaTeX . . . 363
15.7.1 Esempio di composizione in greco . . . 364
15.7.2 XeLaTeX, i font OpenType e i font Type 1 . . . 366
15.8 Conclusione . . . 368

16 LaTeX: le presentazioni 369
16.1 Introduzione . . . 369
16.2 Le classi per le presentazioni . . . 369
16.3 Altre classi per le presentazioni . . . 370
16.4 La classe `beamer` . . . 371
16.5 La documentazione . . . 372
16.6 Una breve presentazione . . . 373
16.7 Creare un nuovo stile . . . 379

16.8 Osservazioni 380

17 LaTeX: la microgiustificazione **381**
17.1 Introduzione 381
17.2 La composizione dei capoversi 382
17.3 Metodi per migliorare la giustificazione 384
17.4 La microgiustificazione 385
17.5 Come funziona la microtipografia 387

18 LaTeX: i caratteri e i font **391**
18.1 Introduzione 391
18.2 Terminologia relativa ai caratteri 391
18.3 I comandi per la scelta dei font 400
18.3.1 La scelta del corpo e dello scartamento 400
18.3.2 La scelta delle altre caratteristiche 403
18.4 Altri font diversi da quelli di default 407
18.5 I font per la matematica 408
18.6 Il Text Companion Font 415
18.7 Gli alfabeti diversi da quello latino 416
18.7.1 Caratteri cirillici 416
18.7.2 Caratteri greci 418
18.7.3 Scrivere con altri alfabeti 421
18.8 La gestione dei font 423
18.8.1 Font OpenType 425
18.8.2 Font Type 1 432
18.8.2.1 Altri font già disponibili 432
18.8.2.2 Installazione di altri font vettoriali 435
18.8.2.3 Operazioni preliminari 436
18.8.3 Installare un font Type 1 creato con FontForge 438
18.8.4 Installare un font Type 1 creato con METAFONT 439
18.8.5 Installare font Type 1 440
18.8.6 Installare font TrueType 441
18.8.7 Installare font OpenType 441
18.8.8 Aggiornamento dei file generali di mappa 441
18.8.8.1 TeX Live 442
18.8.8.2 MiKTeX 444
18.8.9 I file di descrizione dei font 444
18.8.10 Ricapitolazione 449
18.8.11 Font e sistema TeX 450
18.9 Conclusioni 453

19 LaTeX: nuovi comandi **455**
19.1 Introduzione . . . 455
19.2 Le definizioni di comandi nuovi . . . 456
19.2.1 Definizione del logo di MacTeX . . . 457
19.2.2 Definizione del comando `\cs` . . . 457
19.2.3 Il comando `\\` . . . 458
19.2.4 i comandi di definizione asteriscati . . . 459
19.2.5 I capilettera . . . 459
19.2.6 Un titolo di sezione da trattare in modo speciale . . . 461
19.2.7 Scalare il corpo . . . 463
19.3 Ridefinizione di comandi già esistenti . . . 464
19.3.1 Ridefinizione della parole fisse . . . 465
19.3.2 Ridefinizione del comando `\XeLaTeX` . . . 466
19.4 Ridefinizioni di comandi di sistema . . . 467
19.4.1 I numeri romani maiuscoletti . . . 467
19.4.2 La virgola intelligente . . . 472
19.4.2.1 La virgola intelligente che riconosce lo spazio . 473
19.4.2.2 La virgola intelligente che riconosce se è seguita da una cifra . . . 474
19.4.2.3 La virgola intelligente che riconosce il codice di categoria del carattere successivo . . . 474
19.4.2.4 La virgola intelligente che riconosce se è seguita da una cifra ASCII . . . 475
19.4.3 Comandi da eseguire solo per determinate lingue . . . 477
19.4.4 Approfondimenti . . . 477
19.4.5 Il numero nelle liste delle tabelle e delle figure . . . 478
19.5 Esiste già o non esiste ancora il comando? . . . 479
19.6 Definizione di comandi robusti . . . 479
19.7 Definizione di un nuovo ambiente . . . 480
19.7.1 Gli ambienti *medaglione* e *sintassi* . . . 481
19.7.2 Definizione di un ambiente per due figure . . . 482
19.7.3 Commenti sugli ambienti . . . 484
19.8 La ridefinizione di ambienti esistenti . . . 484
19.8.1 Ridefinizione dell'ambiente *theindex* . . . 485
19.8.2 Ridefinizione dell'ambiente *thebibliography* . . . 489
19.9 Situazioni particolari . . . 491
19.9.1 Le linee guida . . . 491
19.9.2 Controllo della posizione di grandi oggetti . . . 494
19.9.3 Immagini, tabelle e scatole . . . 495

20 LaTeX: la geometria delle pagine **503**
20.1 Introduzione . . . 503
20.2 La geometria della pagina . . . 505
20.2.1 Il formato delle pagine . . . 505
20.2.2 Le segnature e le imposizioni . . . 507

20.2.3 I crocini 509
20.2.4 Dimensioni della gabbia del testo 511
20.2.4.1 Testatine e piedini; il pacchetto *fancyhdr* . . . 513
20.2.4.2 Le proporzioni della gabbia di testo 514
20.2.4.3 I margini 515
20.3 Lo scartamento e i contrografismi verticali 521
20.4 I capoversi 524
20.5 Testatine e piedini 526
20.6 I pacchetti di personalizzazione 527
20.6.1 Il pacchetto *geometry* 527
20.6.2 I pacchetti *titlesec*, *fancyhdr*, *sectsty* e *tocloft* 530
20.6.3 Testatine 532
20.7 La pagina del titolo 535
20.8 Le inserzioni flottanti e non flottanti 536
20.8.1 Gli oggetti flottanti 536
20.8.2 Gli oggetti non flottanti 537
20.9 Conclusioni 539

21 Dove documentarsi **541**
21.1 La documentazione essenziale 541
21.2 Documentazione sulla tipografia 542
21.3 Documentazione su LaTeX 545
21.4 Documentazione sulla grafica 546
21.5 Documentazione sui singoli pacchetti 547
21.6 Documentazione su TeX 547
21.7 Documentazione sui simboli di LaTeX 548
21.8 Composizione della matematica 548
21.9 L'archiviazione dei documenti 551

III Volume **553**

22 Il formato PDF archiviabile **555**
22.1 Preliminari 557
22.2 Le immagini 558
22.3 I font 559
22.4 Gli hyperlink 563
22.5 Generazione di un file PDF archiviabile 563
22.5.1 La strada maestra 564
22.5.2 Trasformazione mediante il file *pdfpages* 566
22.5.3 Trasformazione di un file PDF o di un file PS 567
22.5.4 La produzione mediante lualatex 567
22.6 Verifica della conformità 568
22.7 Commenti 568

23 Comporre documenti di molti autori **571**
23.1 Conversione manuale . . . 572
23.1.1 Copia e incolla . . . 573
23.2 Conversione automatica . . . 579
23.2.1 Documenti in formato `.docx` . . . 579
23.2.2 Documenti in formato PDF . . . 583
23.2.3 Documenti in formato XML . . . 584
23.3 Documenti in collaborazione . . . 585
23.3.1 GIT . . . 587
23.3.2 Organizzazione minimale . . . 587
23.3.2.1 File `txt` . . . 588
23.3.2.2 File MMD e MD . . . 588
23.3.3 Andare oltre: Scrivener . . . 589
23.3.3.1 Compilazione del documento finale in PDF . . 591
23.3.3.2 Compilazione del documento finale in file `tex` . 593
23.3.4 Commenti in stile *html* . . . 593
23.4 Conclusioni . . . 597

24 Simbologia matematica e fisica **599**
24.1 Unità di misura del Sistema Internazionale . . . 599
24.2 Simboli matematici nelle scienze . . . 604
24.3 Nomenclatura . . . 604

25 Divisione in sillabe **627**
25.1 Quando viene eseguita la cesura . . . 628
25.2 La sillabazione fonetica oppure etimologica . . . 631
25.3 Come fa il programma a dividere in sillabe . . . 633
25.4 Bruttezza residua . . . 636
25.5 I pattern per la lingua italiana . . . 637
25.6 Cosa fare con le righe troppo lunghe . . . 639
25.7 I file di pattern . . . 640

26 Codifica in entrata e codifica in uscita **645**
26.1 Introduzione . . . 645
26.2 Le tre distinte codifiche di TEX . . . 648
26.2.1 Situazioni di default . . . 656
26.2.2 La codifica di ingresso . . . 656
26.2.3 La codifica di uscita . . . 658
26.3 Specificare la codifica giusta . . . 661
26.3.1 Scoprire la codifica di input . . . 663
26.3.2 Cambiamento della codifica . . . 664
26.4 Collage di contributi diversi . . . 668
26.5 Considerazioni riassuntive . . . 669

27 Come fa TEX a comporre le pagine **673**
27.1 Divisione dei capoversi 674
27.1.1 La microgiustificazione 674
27.1.2 Il meccanismo generale 674
27.2 Divisione della pagine 677
27.3 Cosa fare se... 679
27.3.1 Collocazione degli oggetti flottanti 679
27.3.2 Le equazioni ingombranti 680
27.3.3 I sezionamenti 681
27.4 Conclusioni 682

28 Trattamento degli errori **683**
28.1 Errori ortografici nei nomi dei comandi 683
28.2 Errore nella ricerca dei file 683
28.3 Ciclo infinito 685
28.4 Gruppi aperti 686
28.5 Mancata apertura di un gruppo 687
28.6 Interferenza con le definizioni 687
28.7 File personali che provocano conflitti 688
28.8 Tracciare l'operato del programma 690
28.8.1 I comandi primitivi di tracciamento 690
28.8.2 Il pacchetto `trace` 691
28.9 Costruzione dei capoversi e delle pagine 694
28.10 Conclusioni 697

29 Riepilogo della sintassi di LATEX **699**
29.1 La struttura del documento 703
29.2 Periodi e capoversi 703
29.2.1 Periodi 703
29.2.2 Capoversi 705
29.2.3 Note in calce 706
29.2.4 Note marginali 706
29.2.5 Accenti e simboli speciali 707
29.3 Suddivisione del testo e indici 708
29.3.1 Comandi di sezionamento 708
29.4 Classi, pacchetti e stili delle pagine 711
29.4.1 Classe del documento 711
29.4.2 Pacchetti 713
29.4.3 Stili delle pagine 715
29.4.4 Il frontespizio 716
29.5 Testi in display 718
29.5.1 Citazioni e poemi 718
29.5.2 Liste 718
29.5.3 Testo composto verbatim 723
29.6 Formule matematiche 724

29.6.1 Formule . . . 724
29.6.2 Simboli, accenti, delimitatori e grandi operatori . . . 727
29.6.3 Impilare gli oggetti matematici . . . 727
29.6.4 Spaziatura matematica . . . 727
29.6.5 Font matematici . . . 728
29.6.6 Stili di composizione matematica . . . 729
29.7 Definizioni, numeri e programmazione . . . 729
29.7.1 Comandi di definizione . . . 729
29.7.2 Comandi per la definizione di ambienti . . . 730
29.7.3 Teoremi . . . 730
29.8 Numeri, lunghezze e spazi . . . 731
29.8.1 Numeri . . . 731
29.8.2 Lunghezze . . . 734
29.8.3 Operazioni fra numeri e grandezze . . . 736
29.8.4 Il pacchetti *ifthen* e *etoolbox* . . . 740
29.8.4.1 Il pacchetto *ifthen* . . . 740
29.8.4.2 Il pacchetto *etoolbox* . . . 743
29.9 Spaziature . . . 746
29.10 Figure, tabelle ed altri oggetti flottanti . . . 747
29.10.1 Figure e tabelle . . . 747
29.10.2 Note marginali . . . 753
29.11 Incolonnamenti . . . 754
29.11.1 L'ambiente tabbing . . . 754
29.11.2 Gli ambienti array e tabular . . . 755
29.12 I file ausiliari e i loro comandi . . . 758
29.12.1 I file del sistema TEX . . . 758
29.12.2 I riferimenti incrociati . . . 759
29.12.3 Bibliografia e citazioni . . . 760
29.12.4 Suddivisione del file sorgente . . . 761
29.13 Indice analitico e glossario . . . 762
29.13.1 Indice analitico . . . 762
29.13.2 Glossario . . . 763
29.14 Compilazione interattiva . . . 763
29.15 Interruzione di riga e di pagina . . . 764
29.15.1 Interruzione di riga . . . 764
29.15.2 Interruzione di pagina . . . 765
29.16 Scatole . . . 767
29.16.1 Scatole di uso immediato . . . 767
29.16.2 Scatole per conservare del testo . . . 768
29.16.3 Ambienti e comandi per scatole particolari . . . 769
29.16.4 I comandi per le scatole con il linguaggio di TEX . . . 771
29.16.5 Operazioni di misura sulle scatole . . . 773
29.17 Disegni e colori . . . 774
29.17.1 Disegni . . . 774
29.17.2 Colori e grafica . . . 775

29.18 Selezione dei caratteri 778
29.18.1 Scegliere famiglia, forma e serie 778
29.18.2 Scegliere il corpo 778
29.18.3 Corpi testuali e matematici 778
29.18.4 Simboli speciali 779

Bibliografia **781**

Indice analitico **786**

Indice degli ambienti **804**

Indice delle classi **806**

Indice dei file **807**

Indice dei pacchetti **810**

Indice dei programmi e delle distribuzioni **814**

Elenco delle tabelle

2.1 Componenti testuali di un libro 57

4.1 Legame fra alcuni comandi di sistema, gli interpreti e i formati . 123

8.1 Descrittori delle colonne per le tabelle 199
8.2 Tabella la cui colonna di destra è specificata con `p{80mm}` . . . 200
8.3 Tabella adattata alla giustezza 205
8.4 Tabella male adattata alla giustezza 205
8.5 Tabella bene adattata alla giustezza 206
8.6 Tabella composta con le estensioni del pacchetto *array* 213

9.1 Tabella insolita 236
9.2 Esempio di valutazione di un seminario 245

10.1 Dimensioni di `texbook.pdf` con vari metodi 259

13.1 Le lettere greche 303
13.2 Gli operatori di relazione 304
13.3 Gli operatori binari 304
13.4 Gli operatori funzionali 305
13.5 I grandi operatori 305
13.6 I grandi delimitatori 306
13.7 Accenti e diacritici matematici 307
13.8 Altri simboli 307

14.1 Prima serie di simboli accessibili con il pacchetto *amsmath* . . . 318
14.2 Seconda serie di simboli accessibili con il pacchetto *amsmath* . 319
14.3 Gli ambienti di allineamento di *amsmath* 322

15.1 Formati britannici della pagina 346
15.2 I caratteri della tastiera latina e i segni dell'alfabeto greco . . . 350

18.1 Istruzioni per la scelta del corpo dei caratteri 401
18.2 I corpi ottici a confronto col corpo unico 402

18.3 Il font latino a 128 caratteri con codifica OT1 404
18.4 Il font cirillico a 128 caratteri con codifica OT2 404
18.5 Il font latino a 256 caratteri con codifica T1 405
18.6 Varie combinazioni di serie e forma per i caratteri standard . . . 406
18.7 Il corsivo matematico con codifica OML 410
18.8 La polizza dei simboli matematici con codifica OMS 410
18.9 I segni matematici estensibili con codifica OMX 411
18.10 Il Text Companion Font con codifica TS1 415
18.11 Il font cirillico esteso con la codifica *X2* 417
18.12 Il font greco a 256 caratteri con codifica LGR 420
18.13 Il font latino con codifica LY1 435

19.1 Allineamento scorretto in una tabella 499
19.2 Allineamento corretto in una tabella 499
19.3 Tabella con allineamento corretto e spaziatura corretta 499
19.4 Allineamento aggiustato in modo accurato 501

20.1 Formati delle pagine a seconda della segnatura 506
20.2 Giustezze determinate mediante la lunghezza dell'alfabeto . . . 513

24.1 Unità fondamentali 599
24.2 Prefissi decimali 601
24.3 Prefissi binari 601
24.4 Unità logaritmiche 601
24.5 Unità derivate 602
24.6 Unità di misura legalmente ammesse 603
24.7 Unità di misura temporaneamente accettate 604
24.8 Simboli matematici 605
24.9 Nomenclatura, simboli e unità di misura 619

25.1 Pattern usati per dividere in sillabe *dell'istruzione* 634
25.2 Pattern usati per dividere in sillabe *discinesia* 635

26.1 I 95 caratteri ASCII stampabili 648
26.2 Alcune codifiche di ingresso per i caratteri latini 653

29.1 Gli accenti testuali e i simboli speciali 708
29.2 Numeri e nomi associati al livello di sezionamento 709
29.3 Parametri nella classe `book` per gestire gli oggetti flottanti . . . 752
29.4 Dichiarazioni per la scelta di famiglia, serie e forma 779
29.5 Corrispondenza fra comandi e dichiarazioni 779
29.6 Dichiarazioni di corpo 779

Elenco delle figure

1.1 Semplice esempio di composizione con LaTeX. 42

2.1 Dedica a Donald Knuth 48
2.2 Auguri di TUG (TeX Users Group) per il 2003 48
2.3 Confronto relativo fra le varie scale di lunghezza 50
2.4 Caratteristiche dei caratteri mobili 51
2.5 Cassa dei caratteri di una tipografia italiana 52

4.1 Schermata di LaTeXiT 88
4.2 Schema della tastiera italiana sulle piattaforme Windows 98
4.3 Schema della tastiera configurata con il driver EurKey 99
4.4 Schema della tastiera svizzera sulle piattaforme Windows 101
4.5 Schema della tastiera italiana su Linux senza personalizzazioni . 101
4.6 Schema della tastiera USA estesa sul MacBook Pro 103

6.1 Tre geometrie di pagina a confronto 163
6.2 Proporzioni ISO e aurea della pagina 164

7.1 Il comando `\centering` e l'ambiente *center* 188

9.1 Relazione fra le dimensioni della tela e l'origine degli assi 221
9.2 Le potenzialità dell'ambiente *picture* secondo Lamport 225
9.3 Curve di Bézier di secondo e terzo grado nell'ambiente *picture* . 225
9.4 Moto uniformemente accelerato 226
9.5 Giunzioni e terminazioni delle linee spesse 227
9.6 Disegno ottenuto con il pacchetto *pgf* e l'ambiente *tikzpicture* . 230
9.7 Disegno nell'ambiente *tikzpicture* usando il pacchetto *circuitikz* 231
9.8 Una paginetta con l'angolo ripiegato 233
9.9 Un esempio di figura geometrica composta con METAPOST . . . 234
9.10 Istogramma con il risultato della valutazione di un seminario . . 245
9.11 Diagramma con il risultato della valutazione di un seminario . . 247
9.12 Diagramma a torta con i tipi di produzione dell'energia elettrica 248

10.1 Confronto sugli effetti dell'ingrandimento su due figure simili . . 257

10.2 Inclusione e conversione di un file eps 261
10.3 Due foto trattate con diverse chiavi 272

12.1 Le fasi per la produzione dell'indice analitico 289

13.1 Due esempi di strafalcioni giornalistici 315

15.1 Sette contro Tebe 348
15.2 Esempio di composizione in greco 351
15.3 Alcuni alfabeti gotici che includono anche i capilettera ornati . 355
15.4 Completamento del file sorgente 365
15.5 Risultato della composizione con XeLaTeX 367

16.1 Otto slide per una presentazione di cinque frame 374

18.1 Il carattere metallico della 'm' 392
18.2 Il corpo, l'interlinea e l'avanzamento di riga 392
18.3 Composizione normale e sterlineata 393
18.4 Legature antiche 395
18.5 Il simbolo di *maschio* fortemente ingrandito 399
18.6 Esempi di testi composti con i nuovi font Times e Palatino estesi 409
18.7 Il Credo di Nicea in tre lingue che fanno uso di alfabeti diversi . 422
18.8 I *dingbats* creati da Hermann Zapf 433
18.9 Schema grafico del processo di composizione 451
18.10 Presentazione all'esterno dei risultati della composizione 451
18.11 Processi di conversione dei vari formati di uscita 452
18.12 Trasformazioni di formato dei font e generazione dei file ausiliari 453
18.13 Gestione delle mappe dei font 454

19.1 La nuova cucina 485
19.2 Mansarda con un letto a due piazze 485
19.3 Alcune proprietà delle scatole 496

20.1 Imposizioni per diverse segnature 510
20.2 Esempio di crocini predisposti per una tipografia italiana 511
20.3 Disegno della pagina secondo Gutenberg 516
20.4 Disegno della pagina con il metodo delle strisce 517
20.5 Confronto fra tre gabbie auree 519

23.1 La finestra di compilazione di Scrivener 592
23.2 La compilazione da Scrivener 594
23.3 La compilazione da Scrivener in MMD o LaTeX 595

Avvertenze

In questo testo si parla di quello che si può fare con la ricca dotazione di programmi e di pacchetti presenti in ogni distribuzione del sistema TEX. Le distribuzioni del sistema TEX gratuite sono essenzialmente due, TEX Live e MiKTEX. In commercio ci sono diverse altre distribuzioni.

Qui ci si riferisce principalmente alla distribuzione del sistema eseguita mediante il DVD prodotto dalla Associazione Internazionale degli Utenti di TEX (TEX Users Group, TUG) e ridistribuito anche ai membri dell'Associazione Italiana, GUIT (Gruppo degli Utenti Italiani di TEX e LATEX). Volendo, il DVD può venire acquistato ad un costo nominale dal sito `http://www.tug.org"/store/`, ma nel sito di TUG si dice che l'immagine ISO del disco può essere scaricata gratuitamente. L'immagine ISO (che con i sistemi operativi recenti può essere usata come un vero DVD) può poi essere masterizzata su un DVD vergine e usata tranquillamente alla pari del DVD originale. La distribuzione scaricata dal sito TUG menzionato è completa e permette l'installazione sulle macchine Windows, Linux e Macintosh.

Anche la distribuzione MiKTEX (valida solo per le macchine Windows) può venire scaricata gratuitamente dal sito `http://miktex.org/2.9/setup`. Più precisamente da questo sito è possibile scaricare dei semplici programmi di installazione, uno molto piccolo per installare una versione di base e uno di dimensioni maggiori per scaricare la distribuzione completa. Benché la distribuzione di base possa essere arricchita in modo praticamente automatico via via che l'utente richiama funzionalità ancora non installate, si consiglia vivamente di eseguire l'installazione completa.

Nel seguito si farà riferimento specialmente alla distribuzione TEX Live, perché con minime varianti è identica per tutte le piattaforme di elaborazione più comuni. Anzi, anche per chi lavora su macchine Windows, sebbene esista l'eccellente distribuzione MiKTEX, espressamente prodotta per lavorare con i vari sistemi operativi della Microsoft, si consiglia di installare la versione TEX Live. Certe personalizzazioni necessarie per la composizione tipografica di alta qualità diventano relativamente difficili da fare con la distribuzione MiKTEX, prevista per utenti ai quali, lavorando con un sistema operativo "proprietario", non sono concesse troppe personalizzazioni.

Nel seguito quando si scriverà CTAN ci si riferirà al Comprehensive TEX Archive Network (la rete degli archivi completi del sistema TEX); questa rete è formata da tre server principali dislocati in varie parti del mondo e da un numero sterminato di "mirror" che ne rispecchiano il contenuto. I primi sono aggiornati in modo sincrono ogni notte; i secondi sono aggiornati in modo dilazionato dipendente dalla loro politica amministrativa, ma generalmente sono molto aggiornati anche i mirror.

Nel seguito quando si userà l'indicazione `$TEXMF` ci si riferirà alla cartella o folder che funziona da "radice" per gli "alberi" di cartelle del sistema TEX. Questo sistema, infatti, ha i suoi file collocati in molte cartelle gerarchicamente collegate fra di loro in strutture ad albero; la radice di ciascuno di questi alberi è

posta nel disco fisso dell'elaboratore in posizioni diverse a seconda della macchina e del sistema operativo. Ogni utente conosce la sua macchina e sa dove trovare le cartelle radice dei suoi programmi preferiti.

Tuttavia, tranne dove indicato in modo esplicito, qui si userà solo il segno di barra diritta '/' come separatore fra i nomi delle cartelle; questo è il separatore normale per le macchine che lavorano con i sistemi operativi di tipo UNIX, incluso il sistema operativo delle macchine Macintosh recenti, etichettato con la sigla Mac OS X. Nelle macchine Windows il separatore è notoriamente la barra rovescia '\'; ma anche questo, all'interno dei file che contengono i testi da comporre con i programmi del sistema TEX va sostituito con la barra normale, poiché la barra rovescia per il sistema TEX ha un significato speciale.

I lettori ci vorranno perdonare se abbiamo dato la preferenza sia a TEX Live, sia alla barra diritta; questo è anche legato al fatto che come il sistema TEX fa parte del software libero, così abbiamo dato la preferenza ai sistemi operativi liberi o quasi. Ovviamente non c'è niente di male a servirsi di prodotti commerciali, che talvolta offrono qualcosa di più dei prodotti liberi. Anche nel campo delle distribuzioni del sistema TEX ci sono diverse distribuzioni commerciali che offrono qualcosa in più. Tutti i prodotti commerciali, però, danno qualcosa in meno, nel senso che l'utente è vincolato al prodotto e deve talvolta rinunciare alla possibilità di trasferire agevolmente programmi e documenti da una macchina ad un'altra con un diverso sistema operativo o priva dello stesso prodotto commerciale.

Una certa conoscenza preliminare del sistema TEX è utile per comprendere a fondo questo testo; in ogni caso si dà per scontato che l'utente conosca bene come lavorare sulla propria macchina di elaborazione e abbia una certa familiarità con l'uso della finestra chiamata "Terminale" o "Prompt dei comandi" nei vari sistemi operativi. Talvolta il sistema TEX richiede necessariamente di dare dei comandi in linea, scritti cioè in una di queste finestre specificatamente destinate ad interagire con la macchina mediante comandi scritti, senza usare il mouse o altri comandi "scorciatoia".

Si raccomanda infine di non confondere i tre elementi seguenti

shell editor Uno *shell editor* è uno strumento utilissimo per lavorare con il sistema TEX, ma non è il sistema TEX; è semplicemente un editor di testi che l'utente usa per scrivere il suo testo inframmezzandolo con istruzioni di composizione del linguaggio LATEX; l'editor è poi capace di inviare il file prodotto, detto *file sorgente*, ad un programma interprete del sistema TEX affinché ne esegua la composizione tipografica.

LATEX È un linguaggio di programmazione (non l'unico) per la composizione tipografica di documenti con i programmi del sistema TEX, ma non è il programma di composizione.

Programma di composizione Questo è uno dei vari programmi di composizione facenti parte del sistema TEX che interpreta le istruzioni di composizione inserite dall'utente nel file sorgente del suo documento assieme al testo che vuole comporre; il programma, via via interpreta le istruzioni e acquisisce il testo e li trasforma in testo composto salvando il risultato

complessivo in un file di uscita che quasi sempre ha il formato PDF; PDF è l'acronimo che sta per *Portable Document Format*, il formato raccomandato dalle norme ISO per ogni uso successivo del suo contenuto: lettura a schermo, stampa, distribuzione in Internet, archiviazione, eccetera.

Confondere questi tre concetti vuol dire rendersi il lavoro con il sistema TEX molto complicato.

Tuttavia nel seguito si userà la scrittura LATEX per indicare sia il linguaggio sia i programmi che sanno interpretarlo per eseguire la composizione del documento, ma solo quando non è importante distinguere di quale programma si sta parlando. Quando invece è importante distinguere il programma dal linguaggio, allora manterremo la scrittura LATEX per designare il linguaggio, ma useremo esplicitamente i nomi dei programmi scrivendoli con un font particolare: pdflatex, lualatex e xelatex.

II

Volume

Capitolo 13

LaTeX: la matematica semplice

13.1 Introduzione

La forza di LaTeX, che ne ha determinato la grandissima diffusione iniziale in ambito accademico e tecnico, prima di diffondersi in ambito letterario e umanistico, è proprio la professionalità con la quale compone la matematica; in questo primo capitolo ci soffermeremo sulle strutture matematiche più semplici e nel prossimo capitolo scenderemo nei dettagli per strutture matematiche più complesse, avvalendoci del pacchetto di estensione `amsmath`.

Non è sempre necessario ricorrere al pacchetto `amsmath`; dipende dal tipo di matematica che si vuole comporre. In questo capitolo, quindi, ci familiarizziamo solamente con la composizione della matematica, ma, anche se LaTeX, così com'è appena installato, è già in grado di comporre strutture matematiche piuttosto avanzate, rimandiamo al prossimo capitolo ciò che, pur essendo componibile con LaTeX senza estensioni, riesce molto più comodo da comporre con le estensioni fornite dal pacchetto `amsmath`.

13.2 I modi matematici

LaTeX compone la matematica sostanzialmente in quattro modi; se si andasse per il sottile si scoprirebbe che i modi sono otto, ma questo il lettore interessato lo può studiare nel TeXbook. I modi che interessano la quasi totalità degli utenti sono i seguenti:

1. Il modo testuale. LaTeX compone in questo modo quando scrive una espressione matematica in linea con il testo, come quando si specifica che la sezione aurea è $\varphi = (\sqrt{5}-1)/2$ e che questo numero gode della proprietà che $1/\varphi = 1 + \varphi$. Se si vuole comporre in questo modo anche quando

LaTeX di default comporrebbe in modo diverso bisogna dare l'istruzione `\textstyle`.

2. Il modo display. LaTeX compone la matematica in display quando compone le espressioni in linee a sé stanti, staccate dal testo precedente e seguente mediante spazi bianchi di ampiezza adeguata per 'mettere in mostra' l'espressione; per esempio[1]:

$$\varphi = \frac{\sqrt{5}-1}{2} \qquad \text{e} \qquad \frac{1}{\varphi} = 1 + \varphi \qquad \text{(sezione aurea)}$$

   Se si vuole comporre in questo modo anche quando LaTeX di default comporrebbe in modo diverso bisogna dare l'istruzione `\displaystyle`.

3. Il modo degli indici primi. LaTeX compone gli apici e i pedici (gli esponenti e i deponenti) di primo ordine in questo modo; usa un carattere di corpo più piccolo rispetto a quello delle variabili principali e li rialza per gli apici o li abbassa per i pedici di una quantità costante e prestabilita dalle caratteristiche dei font usati; per esempio in display si hanno le soluzioni di una equazione di secondo grado nella forma:

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

   Se si vuole comporre in questo modo anche quando LaTeX di default comporrebbe in modo diverso bisogna dare l'istruzione `\scriptstyle`.

4. Il modo degli indici secondi. LaTeX compone gli apici e i pedici (gli esponenti e i deponenti) di secondo ordine, cioè quelli applicati agli apici e ai pedici di primo livello, con un carattere ancora più piccolo e adeguatamente rialzato o ribassato rispetto alle variabili a cui si riferiscono, che sono già per loro conto composte con il font degli indici di primo livello; per esempio:

$$V_{R_\mathrm{e}} = R_\mathrm{e} I_\mathrm{e}$$

   Se si vuole comporre in questo modo anche quando LaTeX di default comporrebbe in modo diverso bisogna dare l'istruzione `\scriptscriptstyle`.

Di solito è piuttosto raro che si debba specificare lo stile, ovvero il modo di composizione matematica; è molto meglio lasciare fare a LaTeX che sa quasi sempre qual è il modo giusto da usare per la composizione.

Per formule così semplici come quelle esposte nell'enumerazione precedente basta che il compositore scriva il testo sorgente in lettere nello stesso modo come se lo dettasse al telefono (in inglese). Per esempio:

[1] Spesso, specialmente nella letteratura anglosassone, con ϕ si intende il numero aureo $(\sqrt{5}+1)/2$. Si riesce benissimo a distinguere dal contesto del discorso a quale dei due significati sia associato il simbolo; qui si userà φ per la sezione aurea e ϕ per il numero aureo.

```
\begin{displaymath}
   x_{1,2}= \frac{-b\pm\sqrt{b^2-4ac}}{2a}
\end{displaymath}
```

L'ambiente[2] *displaymath* dice "metti in display l'espressione matematica che racchiudo". Il segno `_` serve per dire "metti in posizione di deponente quanto contenuto nel prossimo gruppo"; il gruppo è delimitato dalle parentesi graffe, come si è già avuto modo di osservare; `\frac` sta per *fraction*; è una abbreviazione di una parola inglese, ma è una parola quasi completa; essa dà l'ordine di comporre una frazione; quindi bisogna specificare il numeratore e il denominatore mediante le rispettive espressioni racchiuse nei rispettivi gruppi; all'interno del numeratore compare il comando `\pm` che vuol dire "metti un segno 'più o meno' "; compare anche il comando `\sqrt`, sigla inglese che sta per *square root*, radice quadrata, ed esso comanda di mettere sotto radice il contenuto del gruppo che lo segue; il segno `^` vuol dire "metti ad esponente il contenuto del prossimo gruppo", ma in questo caso il gruppo non è evidenziato con le graffe, perché è costituito da un solo token.

Come si vede la scrittura di una breve espressione è semplicissima e non richiede una memoria particolare per ricordare i comandi da usare. Quando ci sono da scrivere delle lettere greche esse si scrivono con i loro nomi per disteso (in inglese) e le maiuscole si scrivono con l'iniziale maiuscola; per non confondere le sequenze di lettere che formano i nomi delle lettere greche come semplici variabili matematiche scritte in lettere latine, i nomi delle lettere greche sono preceduti dal segno '\': `\alpha`, `\beta`, `\gamma`, ..., `\Gamma`, `\Delta`, eccetera.

Per gli operatori matematici più comuni bastano i tasti della tastiera di un qualunque PC o laptop. Se il PC o il laptop è dotato di una tastiera estesa o avanzata, è possibile comporre da tastiera anche altri segni che non sono serigrafati sui tasti; bisogna solamente premere contemporaneamente uno o più tasti di controllo (Alt, Shift, Ctrl, ecc.) e almeno un tasto 'ordinario'. In ogni caso nelle prossime pagine appaiono un certo numero di tabelle che contengono la lista completa dei comandi matematici di LaTeX che si possono introdurre senza disporre di nessuna tastiera speciale.

Gli spazi che in matematica compaiono nel file sorgente non hanno nessuna rilevanza; LaTeX mette gli spazi matematici dove sono necessari e, anche se offre dei comandi per inserire esplicitamente alcuni tipi di spazi matematici, l'utente è vivamente invitato a non farne uso! LaTeX compone benissimo da solo, senza che l'utente debba perdere tempo per rendere il 99% delle volte il frutto della sua composizione meno bello e meno professionale di quello che LaTeX avrebbe fatto da solo.

Esistono alcuni pochi casi ben noti in cui l'utente può inserire degli spazi matematici; essi riguardano tutti gli operatori obliqui quando gli oggetti che li precedono o li seguono hanno forme particolari, oppure quando l'utente deve lavorare con integrali multipli e non vuole usare il pacchetto di estensione

[2] I comandi `\[` e `\]` sostituiscono efficacemente i comandi di apertura e di chiusura dell'ambiente *displaymath*.

amsmath, di cui si parlerà nel prossimo capitolo. Quindi il compositore sarà autorizzato a correggere gli spazi usati da LaTeX solo dopo aver corretto le bozze e se vuol aggiungere quei rarissimi tocchi di qualità che LaTeX non è riuscito a mettere da solo.

Per aiutare l'utente ad avere sottomano tutti i simboli che LaTeX mette a disposizione (e sono moltissimi) sono state preparate alcune tabelle di segni accostati ai comandi che li producono; si vedano allora le tabelle 13.1–13.8

Va detto subito che nelle tabelle 13.2, 13.3 e 13.8 i segni con uno sfondo grigio non sono usabili se non si usa il pacchetto di estensione *amsfonts* oppure il pacchetto di estensione *latexsym*; in realtà questo secondo pacchetto rende accessibili alcuni segni che con il vecchio LaTeX 2.09 erano direttamente accessibili senza estensioni; perciò questo pacchetto è da considerare come una maniera per compilare vecchi file sorgente predisposti per il vecchio LaTeX. Con il nuovo LaTeX 2_ε questi segni sono tutti accessibili tramite il pacchetto *amsfonts*. Siccome il pacchetto fa parte della collezione a cui appartiene anche *amsmath*, e questo è utilissimo per la composizione di testi che fanno uso di molta matematica, questi due pacchetti vengono solitamente caricati comunque, quindi non è un gran problema disporre di quei simboli. Tuttavia bisogna ancora segnalare che alcuni, come per esempio $\mho$, non dovrebbero mai venire usati perché contrari alle norme ISO, quindi non è una gran perdita se non se ne fa uso.

13.3 Alcune annotazioni sulle lettere greche

Per quanto riguarda le lettere greche, si può notare che alcune di esse hanno delle forme varianti i cui nomi cominciano con l'abbreviazione `var`. Esse sono delle varianti per quel che riguarda gli Stati Uniti, ma, tolto ϖ che non si usa mai, forse nemmeno negli Stati Uniti, gli altri segni sono quelli usati normalmente in Europa e in Italia; in alcuni file di classe che lo scrivente ha creato, egli ha scambiato i nomi varianti con quelli ordinari per rendere normali quelli che vengono usati di solito in Europa.

Inoltre in tutto il mondo è consuetudine scrivere le lettere greche maiuscole con caratteri diritti e non inclinati, nonostante le norme ISO prescrivano l'uso delle lettere inclinate per qualsiasi variabile matematica o fisica.

Se si desidera scrivere le lettere greche maiuscole con caratteri inclinati bisogna fare esattamente come è suggerito nella tabella 13.1. Oppure, se si sta facendo uso del pacchetto *amsmath*, si ottengono le lettere inclinate usando i comandi varianti (non visibili in nessuna delle tabelle citate) dove il nome delle lettere greche con iniziale maiuscola è preceduto da `var`. Scrivendo quindi `\varGamma`, `\varDelta`, `\varTheta`, eccetera, si ottengono i simboli $\varGamma$, $\varDelta$, $\varTheta$, ...

Minuscole					
\alpha	α	\iota	ι	\rho	ρ
\beta	β	\kappa	κ	\sigma	σ
\gamma	γ	\lambda	λ	\tau	τ
\delta	δ	\mu	μ	\upsilon	υ
\epsilon	ϵ	\nu	ν	\phi	ϕ
\zeta	ζ	\xi	ξ	\chi	χ
\eta	η	o	o	\psi	ψ
\theta	θ	\pi	π	\omega	ω
Varianti delle minuscole					
\varepsilon	ε	\varpi	ϖ	\varsigma	ς
\vartheta	ϑ	\varrho	ϱ	\varphi	φ
Maiuscole					
\Gamma	Γ	\Xi	Ξ	\Phi	Φ
\Delta	Δ	\Pi	Π	\Psi	Ψ
\Theta	Θ	\Sigma	Σ	\Omega	Ω
\Lambda	Λ	\Upsilon	Υ		
Maiuscole corsive					
\mathit{\Gamma}	$\mathit{\Gamma}$	\mathit{\Xi}	$\mathit{\Xi}$	\mathit{\Phi}	$\mathit{\Phi}$
\mathit{\Delta}	$\mathit{\Delta}$	\mathit{\Pi}	$\mathit{\Pi}$	\mathit{\Psi}	$\mathit{\Psi}$
\mathit{\Theta}	$\mathit{\Theta}$	\mathit{\Sigma}	$\mathit{\Sigma}$	\mathit{\Omega}	$\mathit{\Omega}$
\mathit{\Lambda}	$\mathit{\Lambda}$	\mathit{\Upsilon}	$\mathit{\Upsilon}$		

Tabella 13.1: Le lettere greche. La "omicron" di fatto è la lettera latina, che d'altra parte non si usa mai in matematica.

13.4 Alcune osservazioni sugli operatori funzionali

Gli operatori funzionali come sin, arctan, eccetera sono presentati nella tabella 13.4; essi *devono* essere usati esattamente con quei comandi per due importanti motivi.

1. I nomi che vengono scritti nel documento durante l'esecuzione di quei comandi sono quelli prescritti dalle norme internazionali ISO e, siccome essi sono indicati anche dalle norme nazionali UNI e queste in Italia hanno valore di legge, non è consentito usare nomi diversi, anche se si è abituati a scrivere 'tg' al posto di 'tan' oppure 'sen' al posto di 'sin'. In realtà le norme ISO prescrivono nomi che costituiscono abbreviazioni di parole o locuzioni latine, quindi *Ubi maior, minor cessat*!

2. Gli operatori, in quanto tali, richiedono certi spazi alla loro destra e alla loro sinistra; questi spazi dipendono dalla natura degli oggetti matematici che precedono o seguono gli operatori; solo usando quei comandi LaTeX sa che sta usando degli operatori e sa quali spazi usare.

\approx	≈	\asymp	≍	\bowtie	⋈
\cong	≅	\dashv	⊣	\doteq	≐
\downarrow	↓	\Downarrow	⇓	\equiv	≡
\frown	⌢	\ge	≥	\geq	≥
\gets	←	\gg	≫	\hookleftarrow	↩
\hookrightarrow	↪	\iff	⟺	\in	∈
\Join	⋈	\le	≤	\leadsto	⇝
\leftarrow	←	\Leftarrow	⇐	\leftharpoondown	↽
\leftharpoonup	↼	\leftrightarrow	↔	\Leftrightarrow	⇔
\leq	≤	\ll	≪	\longleftarrow	⟵
\Longleftarrow	⟸	\longleftrightarrow	⟷	\Longleftrightarrow	⟺
\longmapsto	⟼	\longrightarrow	⟶	\Longrightarrow	⟹
\mapsto	↦	\mid	∣	\models	⊨
\ne	≠	\nearrow	↗	\neq	≠
\ni	∋	\not=	≠	\nwarrow	↖
\parallel	∥	\perp	⊥	\prec	≺
\preceq	⪯	\propto	∝	\rightarrow	→
\Rightarrow	⇒	\rightharpoondown	⇁	\rightharpoonup	⇀
\rightleftharpoons	⇌	\searrow	↘	\sim	∼
\simeq	≃	\smile	⌣	\sqsubset	⊏
\sqsubseteq	⊑	\sqsupset	⊐	\sqsupseteq	⊒
\subset	⊂	\subseteq	⊆	\succ	≻
\succeq	⪰	\supset	⊃	\supseteq	⊇
\swarrow	↙	\to	→	\uparrow	↑
\Uparrow	⇑	\vdash	⊢		

Tabella 13.2: Gli operatori di relazione

\amalg	∐	\ast	∗	\bigcirc	◯
\bigtriangledown	▽	\bigtriangleup	△	\bullet	•
\cap	∩	\cdot	·	\circ	∘
\cup	∪	\dagger	†	\ddagger	‡
\diamond	⋄	\div	÷	\lhd	◁
\mp	∓	\odot	⊙	\ominus	⊖
\oplus	⊕	\oslash	⊘	\otimes	⊗
\pm	±	\rhd	▷	\setminus	\
\sqcap	⊓	\sqcup	⊔	\star	⋆
\times	×	\unlhd	⊴	\unrhd	⊵
\uplus	⊎	\vee	∨	\wedge	∧
\wr	≀				

Tabella 13.3: Gli operatori binari

La tabella 13.4 suddivide gli operatori in due categorie: (*a*) quelli chiamati 'operatori senza limiti' nei quali le indicazioni di esponente o di deponente vengono usate normalmente come tali, e (*b*) quelli chiamati 'operatori con limiti' per i quali le indicazioni di esponente e di deponente vengono usati come nei

Operatori senza limiti				
\arccos	\arcsin	\arctan	\arg	\cos
\cosh	\cot	\coth	\csc	\deg
\dim	\exp	\hom	\ker	\lg
\ln	\log	\sec	\sin	\sinh
\tan	\tanh			
Operatori con limiti				
\det	\gcd	\inf	\lim	\liminf
\limsup	\max	\min	\Pr	\sup

Tabella 13.4: Gli operatori funzionali

\bigcap	$\bigcap$	$\cap$	\bigvee	$\bigvee$	$\vee$
\bigcup	$\bigcup$	$\cup$	\bigwedge	$\bigwedge$	$\wedge$
\bigodot	$\bigodot$	$\odot$	\coprod	$\coprod$	$\amalg$
\bigoplus	$\bigoplus$	$\oplus$	\int	$\int$	$\int$
\bigotimes	$\bigotimes$	$\otimes$	\oint	$\oint$	$\oint$
\bigsqcup	$\bigsqcup$	$\sqcup$	\prod	$\prod$	$\prod$
\biguplus	$\biguplus$	$\uplus$	\sum	$\sum$	$\sum$

\sqrt	$\sqrt{}$	$\sqrt{}$	$\sqrt{}$	$\sqrt{}$	$\sqrt{}$

Tabella 13.5: I grandi operatori

simboli di sommatoria; si noti la differenza nell'equazione 13.1:

$$\lim_{x\to 0}\sum_{k=0}^{\infty}\sin^2 2kx = 0 \tag{13.1}$$

scritta facendo uso dei segni _ e ^ che normalmente indicano rispettivamente l'inserzione di un deponente o di un esponente:

```
\lim_{x\to 0}\sum_{k=0}^\infty \sin^2 2kx = 0
```

Si noti che *in modo testo* LaTeX compone i 'limiti' degli operatori con limiti, delle sommatorie, degli integrali, e simili, come se fossero dei deponenti o degli esponenti; esso si comporta in questo modo per evitare di far diventare troppo grandi (alte e profonde) le righe con gli operatori con limiti che obbligherebbero

(	/	\langle	\rangle
)	\lfloor	\rfloor	\lgroup
[	\lceil	\rceil	\rgroup
]	\backslash	\vert	\bracevert
|	\Vert	\updownarrow	\Updownarrow
|	\uparrow	\Uparrow	\downarrow
}	\arrowvert	\Arrowvert	\Downarrow
{	\lmoustache	\rmoustache	

Tabella 13.6: I grandi delimitatori

a inserire di una generosa interlinea; in modo display, invece tutto procede come esemplificato nell'equazione 13.1.

Se si desidera definire nuovi operatori funzionali oltre a quelli che compaiono nella tabella 13.4, il pacchetto *amsmath* mette a disposizione due comandi con la seguente sintassi:

```
\DeclareMathOperator{⟨operatore⟩}{⟨definizione⟩}
\DeclareMathOperator*{⟨operatore⟩}{⟨definizione⟩}
```

dove ⟨*operatore*⟩ è il comando con il quale si vuole invocare l'operatore mentre la ⟨*definizione*⟩ contiene il modo di scrivere il nome dell'operatore. La dichiarazione asteriscata definisce il comando per un operatore con limiti, mentre quella non asteriscata lo definisce per un operatore senza limiti.

\hat{x}	$\hat{x}$	\check{x}	$\check{x}$
\breve{x}	$\breve{x}$	\acute{x}	$\acute{x}$
\grave{x}	$\grave{x}$	\tilde{x}	$\tilde{x}$
\bar{x}	$\bar{x}$	\vec{x}	$\vec{x}$
\dot{x}	$\dot{x}$	\ddot{x}	$\ddot{x}$
\overline{x+y}	$\overline{x+y}$	\underline{x+y}	$\underline{x+y}$
\overbrace{x\cdots y}	$\overbrace{x\cdots y}$	\underbrace{x\cdots y}	$\underbrace{x\cdots y}$
\overrightarrow{x-y}	$\overrightarrow{x-y}$	\overleftarrow{x-y}	$\overleftarrow{x-y}$
\widehat{x-y}	$\widehat{x-y}$	\widetilde{x-y}	$\widetilde{x-y}$
\mathring{x}	$\mathring{x}$		

Tabella 13.7: Accenti e diacritici matematici

\aleph	$\aleph$	\angle	$\angle$	\backslash	$\backslash$
\bot	$\bot$	\Box	$\Box$	\clubsuit	$\clubsuit$
\Diamond	$\Diamond$	\diamondsuit	$\diamondsuit$	\ell	ℓ
\emptyset	$\emptyset$	\exists	$\exists$	\flat	$\flat$
\forall	$\forall$	\hbar	$\hbar$	\heartsuit	$\heartsuit$
\Im	$\Im$	\imath	$\imath$	\infty	∞
\jmath	$\jmath$	\mho	$\mho$	\nabla	∇
\natural	$\natural$	\neg	$\neg$	\partial	∂
\prime	$\prime$	\Re	$\Re$	\sharp	$\sharp$
\spadesuit	$\spadesuit$	\surd	$\surd$	\top	$\top$
\triangle	$\triangle$	\wp	$\wp$	\\|	$\|$

Tabella 13.8: Altri simboli

Per esempio, l'operatore per separare la parte reale di un numero complesso è `\Re` e produce il simbolo $\Re$; volendolo ridefinire in modo che esso sia scritto in caratteri non gotici, ma sia in tondo, si potrebbe usare la dichiarazione[3]:

```
\DeclareMathOperator{\Re}{Re}
```

In questo modo lo stesso comando `\Re` non produce più $\Re z = \Re(x + \mathrm{i}\, y) = x$ ma $\mathrm{Re}\, z = \mathrm{Re}(x + \mathrm{i}\, y) = x$ e tratta il nuovo simbolo come un operatore senza limiti lasciandovi attorno gli stessi spazi che lascia normalmente attorno agli operatori. Si noti che anche l'unità immaginaria 'i' è un operatore (vettoriale) ed è stato stampato facendo uso del comando `\uimm` la cui definizione un po' particolare verrà discussa nella pagina 328.

[3]Il comando `\Re` è già definito e i vari comandi `\Declare...` non producono messaggi d'errore se si ridefiniscono comandi esistenti; controllano però l'esistenza di un comando pre-esistente e, nel caso, scrivono un avvertimento nel file `.log`.

Si noti che non è necessario specificare il font di forma tonda, perché `\DeclareOperatorName` usa il font predefinito per gli operatori, ché è generalmente il tondo chiaro. Certo se uno desiderasse l'uso di un font diverso, fra quelli che si possono usare in matematica, lo deve specificare espressamente; se per esempio si volesse scrivere l'operatore `\Re` in tondo nero bisognerebbe dare la definizione:

```
\DeclareMathOperator{\Re}{\mathbf{Re}}
```

Di solito, però, non è una buona idea cambiare forma o serie rispetto alle forme o alle serie predefinite, perché queste in matematica hanno significati speciali ed è meglio evitare situazioni in cui si possano manifestare degli equivoci.

13.5 Alcune osservazioni sui grandi operatori

I grandi operatori hanno almeno due dimensioni, una 'normale' adatta al modo testo e una più grande adatta al modo display; la radice quadrata, invece ha numerose varianti adatte alla grandezza del radicando.

Così in modo testo si potrà scrivere con l'operatore piccolo (scelto automaticamente da LaTeX senza che il compositore se ne debba preoccupare) $\sum_{k=0}^{\infty} x^k = 1/(1-x)$ mentre in display sarà:

$$\sum_{k=0}^{\infty} x^k = \frac{1}{1-x} \tag{13.2}$$

Bisogna fare attenzione ad un dettaglio non trascurabile. Mentre la maggior parte delle volte LaTeX usa correttamente (almeno nel senso delle nostre aspettative) i grandi operatori scegliendo l'operatore grande oppure quello piccolo, talvolta ci sembra che si sbagli; si osservi l'equazione 13.3:

$$\frac{1}{\sum_{k=0}^{\infty} x^k} = 1 - x \tag{13.3}$$

Si nota che, nonostante l'equazione sia in display, LaTeX ha usato l'operatore del modo testo. A noi sembra scorretto, ma LaTeX ha agito correttamente, perché, nonostante si trovi a comporre una formula in display, se usasse la forma grande dell'operatore, l'intera formula risulterebbe squilibrata:

$$\frac{1}{\displaystyle\sum_{k=0}^{\infty} x^k} = 1 - x \tag{13.4}$$

come si vede bene nell'equazione 13.4.

LaTeX si comporta in questo modo tutte le volte che cerca di mantenere più o meno costante l'altezza delle righe nelle espressioni matematiche; esso infatti passa al modo testo anche per scrivere gli elementi delle matrici; bisogna

ricordarsene e, se non ci piacesse, sarebbe nostra cura modificare il modo di scrivere le formule in modo da ottenere una forma grafica gradevole e nello stesso tempo corretta; si veda a questo proposito l'equazione 13.6.

13.6 I grandi delimitatori

Nella tabella 13.6 sono rappresentati i grandi delimitatori, o meglio i delimitatori estensibili; questi si possono estendere in altezza in modo virtualmente illimitato e possono racchiudere espressioni di qualunque grandezza; si pensi anche solo ai delimitatori che differenziano una matrice da un determinante. Ma i delimitatori possono essere resi estensibili solo premettendo al delimitatore di sinistra il comando `\left` e al delimitatore di destra il comando `\right`; questi due comandi devono essere sempre appaiati nella stessa espressione matematica, quindi se si deve mettere un solo delimitatore, l'altro lo si rende invisibile scrivendovi un 'punto' invece del segno di una parentesi; si veda l'equazione 13.5:

$$x_{1,2} = \begin{cases} \dfrac{-b \pm \sqrt{b^2 - 4ac}}{2a} & \text{se} \quad b^2 - 4ac > 0 \\ -\dfrac{b}{2a} & \text{se} \quad b^2 - 4ac = 0 \\ \dfrac{-b \pm \mathrm{i}\sqrt{4ac - b^2}}{2a} & \text{se} \quad b^2 - 4ac < 0 \end{cases} \tag{13.5}$$

composta scrivendo

```
x_{1,2} = \left\{ ... \right.
```

Per comporre la struttura dell'equazione 13.5, già più complessa di quelle usate come esempio fino ad ora, si può usare l'ambiente *cases* fornito dal pacchetto *amsmath*; con questo ambiente non è il caso di preoccuparsi di usare delimitatori invisibili perché ci pensa lui; anzi, questa osservazione ci fa capire perché è meglio servirsi di ambienti di composizione appositamente predisposti dal file di classe o dai file di estensione o da definizioni create dall'utente, piuttosto che ripetere ogni volta l'intero assemblaggio di comandi e di istruzioni, col rischio di dimenticarne qualcuno o di non essere coerenti ogni volta che si deve comporre lo stesso genere di struttura nella stessa maniera.

Ma `\left` e `\right` sono comodi anche per scrivere correttamente in modo display una formula come l'equazione 13.4 che diventa:

$$\left(\sum_{k=0}^{\infty} x^k \right)^{-1} = 1 - x \tag{13.6}$$

Bisogna prestare una particolare attenzione quando si deve spezzare una equazione fra più righe; quando si è in modo testo si suppone che le espressioni stampate siano molto semplici e non troppo lunghe; se l'espressione cade in fin

di riga LaTeX (con molta riluttanza) provvede a dividere l'espressione dopo un operatore di relazione o dopo un operatore binario. Tuttavia sta al compositore evitare che si debba andare a capo mentre si è all'interno di una parentesi. Quando invece si è in display le espressioni possono essere anche molto lunghe e potrebbe essere necessario andare a capo all'interno di una parentesi. Solo chi ha scritto l'equazione ne conosce il ritmo di lettura e solo lui sa dove è meno grave andare a capo. Tuttavia resta la regola principale che si può andare a capo dopo gli operatori di relazione e, subordinatamente, dopo gli operatori binari, mai dopo gli operatori funzionali o i grandi operatori o immediatamente dopo i delimitatori di apertura. Ecco perché sono utili le tabelle 13.3–13.6, così anche chi non è matematico sa di che cosa si sta parlando.

Nella disgraziata ipotesi che fosse necessario andare a capo all'interno di una lunga espressione fra parentesi, allora potrebbero risultare utili i seguenti comandi, che, come `\left` e `\right` ingrandiscono i delimitatori, ma di una quantità fissa, indipendentemente da che cosa essi delimitino; non è difficile ottenere:

$$\Biggl(\biggl(\Bigl(\bigl(()\bigr)\Bigr)\biggr)\Biggr)$$

con i comandi

```
\Biggl(\biggl(\Bigl(\bigl(()\bigr)\Bigr)\biggr)\Biggr)
```

La lettera `l` o `r` alla fine di ogni comando specifica se il segno che segue è un delimitatore di sinistra (*left*) o di destra (*right*); esiste anche un suffisso `m` per i separatori che talvolta si trovano a metà di certe espressioni matematiche.

13.7 Accenti e segni diacritici matematici

Per segnare con 'diacritici' matematici i vari simboli che si usano, esistono i comandi riportati nella tabella 13.7; sarebbe un grave errore usare quegli accenti per inserire diacritici in modo testo. Per questi si veda la tabella 29.1. Per mettere accenti sulle lettere 'i' e 'j' in matematica bisogna ricorrere a due particolari versioni di questi segni privi del puntino: `\imath` e `\jmath`. Usando questi segni si può scrivere per esempio $\vec{a} = a_x\vec{\imath} + a_y\vec{\jmath}$ con il codice[4]:

```
\( \vec a = a_x \vec\imath + a_y \vec\jmath \)
```

Ma fra i segni che si possono mettere sopra o sotto singoli simboli o addirittura gruppi di simboli o espressioni matematiche, si possono notare in particolare `\overbrace` e `\underbrace`, che hanno proprietà diverse dagli altri diacritici: essi non servono solamente per mettere sopra o sotto una espressione matematica

[4] I comandi `\imath` e `\jmath` per l'interprete sono ciascuno un solo token, anche se a noi umani sembrano sequenze di segni e di lettere e quindi ci appaiono ciascuno come una moltitudine di token.

una graffa orizzontale, ma servono anche per applicare una annotazione all'espressione, una annotazione che di solito è costituita da un'altra breve espressione matematica; infatti la loro sintassi è rispettivamente:

```
\overbrace{⟨espressione matematica⟩}^{⟨annotazione⟩}
\underbrace{⟨espressione matematica⟩}_{⟨annotazione⟩}
```

e l'⟨*annotazione*⟩ (matematica) viene collocata rispettivamente sopra o sotto l'⟨*espressione principale*⟩ nel corpo degli indici di primo livello.

Va da sé che se l'annotazione è testuale l'⟨*annotazione*⟩ deve essere scritta in ambiente testo; quindi, se si sta usando il modulo **amsmath**, si può scrivere la nota nella forma `\text{`⟨*annotazione*⟩`}`; se invece non si sta usando **amsmath** bisogna ricorrere a una "scatola", come `\mbox`, nella quale si deve anche specificare a mano il corpo del font usato per la nota, per esempio `\mbox{`⟨`\scriptsize` *annotazione*⟩`}`.

Sarebbe un errore usare il comando `\mathrm` per scrivere in tondo l'annotazione perché questa verrebbe scritta in tondo ma in modo matematico. Si confronti

- "valori efficaci", scritto con `\mbox{valori efficaci}`, con
- "valoriefficaci", scritto con `\mathrm{valori efficaci}`.

13.8 Gli ambienti matematici

Per inserire una espressione matematica in linea con il testo bisogna entrare in modo matematico; la maniera più semplice, ma sconsigliata, è quella di racchiudere i comandi matematici all'interno di un gruppo costituito da due segni di dollaro `$`. Così per scrivere che $\sin(\alpha + \beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta$, basta inserire nel testo sorgente

```
$\sin(\alpha+\beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta $
```

Ci si rende conto, però, che il segno di dollaro `$` funziona come un interruttore che cambia modo da testo a matematico o viceversa, senza preoccuparsi di controllare se l'operazione sia corretta; LaTeX offre il modo di eseguire questo controllo se si racchiude l'espressione matematica fra i comandi `\(` e `\)`, così:

```
\(
   \sin(\alpha+\beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta
\)
```

Non è necessario andare a capo dopo `\(` e prima di `\)`, ma il testo sorgente è più facilmente leggibile.

Il comando `\(` può essere dato solo mentre si è in modo testo e `\)` solo quando si è in modo matematico; se ciò non succedesse, LaTeX provvederebbe

ad emettere gli opportuni messaggi d'errore che consentirebbero di eseguire le necessarie correzioni con facilità.

Analogamente in modo display sarebbe possibile usare i comandi di basso livello costituiti da due coppie di segni di dollaro; per scrivere:

$$\sin(\alpha + \beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta$$

basterebbe scrivere nel file sorgente:

```
$$
   \sin(\alpha+\beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta
$$
```

ma di nuovo mancherebbe la diagnostica di LaTeX. Va notato che aprire e chiudere una espressione matematica fra due coppie di dollari è incompatibile con altri pacchetti basati sulla sintassi di LaTeX; fra questi il pacchetto ***amsmath*** che è estremamente importante per la composizione della matematica avanzata – ma, direi, anche per la matematica semplice perché offre comandi e ambienti utilissimi e non disponibili nel nucleo di LaTeX. La delimitazione delle espressioni matematiche in display mediante le coppie di dollari è anche incompatibile con le opzioni e i pacchetti che consentono di comporre le espressioni accostate a sinistra.

LaTeX, invece, offre due ambienti per scrivere espressioni senza la numerazione e un ambiente per scrivere espressioni numerate:

```
\[
   \sin(\alpha+\beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta
\]
```

con la sintassi:

```
\[ ⟨espressione matematica⟩\]
```

oppure con l'ambiente *displaymath*:

```
\begin{displaynmath}
   \sin(\alpha+\beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta
\end{displaymath}
```

che segue la sintassi:

```
\begin{displaymath}
⟨espressione matematica⟩
\end{displaymath}
```

Per le espressioni numerate si usa l'ambiente *equation* con la sintassi:

```
\begin{equation}
⟨espressione matematica⟩
\label{⟨etichetta⟩}
\end{equation}
```

Usando il pacchetto `amsmath` diventa disponibile l'ambiente *equation**:

```
\begin{equation*}
⟨espressione matematica⟩
\end{equation*}
```

del tutto equivalente all'ambiente *displaymath* solo che gestisce meglio gli spazi e accetta anche espressioni matematiche più complesse disponibili con quel pacchetto.

La sintassi è auto esplicativa, ma si noti che l'⟨*etichetta*⟩ può contenere un nome simbolico con il quale richiamare l'equazione per nome, invece che per numero, rendendo così automatico il processo dei riferimenti incrociati mediante i comandi `\label`, `\ref` e `\pageref`. Il pacchetto `amsmath` per citare le equazioni mette a disposizione anche il comando `\eqref` che provvede da solo a racchiuderne il riferimento vero fra parentesi tonde come si fa abitualmente (e come *non* si è fatto in questo testo). Quel pacchetto mette a disposizione anche un comando `\tag` per identificare le equazioni; esso consente di assegnare all'equazione un 'nome' letterale, invece che un identificativo numerico, ed è usabile anche all'interno dell'ambiente *equation**, benché questo si riferisca ad un ambiente non numerato; è proprio quello che si è fatto per l'equazione che definisce la sezione aurea nella pagina 300.

13.9 Le unità di misura

Quanto esposto nei paragrafi precedenti dovrebbe consentire di scrivere equazioni di una riga, numerate o non numerate e di poterle citare in modo corretto senza scambi di identità.

Prima di passare al capitolo successivo vale la pena di soffermarci su una questione molto importante, visto che la matematica non serve solo ai matematici puri, ma anche ai tecnologi, ai fisici, ai chimici, e a tanti altri tecnici e scienziati che scrivono equazioni fra grandezze misurabili con adeguati strumenti.

Le scienze sperimentali usano espressioni che coinvolgono grandezze misurabili e con queste la composizione della matematica deve soddisfare a ulteriori regole specificate dalle norme ISO–UNI.

1. Le grandezze vanno scritte in corsivo matematico; anche le costanti fisiche vanno scritte in corsivo matematico, a differenza delle costanti matematiche che vanno scritte in tondo;

2. quando si esplicita il valore numerico delle costanti fisiche o di qualunque grandezza, bisogna inserire anche le unità di misura.

3. Le unità di misura devono seguire scrupolosamente le norme ISO, sia per i loro nomi, sia per i loro simboli e i loro prefissi decimali; inoltre

   (a) vanno sempre scritte dopo (a destra) del valore numerico che indica la misura della granzza;

   (b) devono essere scritte in tondo anche nelle espressioni matematiche;

   (c) devono essere separate da uno spazio fine dal valore numerico;

   (d) questo spazio deve essere 'non separabile' nel senso che non si può avere un 'a capo' fra la misura e l'unità di misura;

   Tutte queste azioni si ottengono in un colpo solo se si usa il comando `\unit`[5] disponibile con il pacchetto *babel* quando si scrive in italiano. Bisogna solo scrivere il comando `\unit` con il suo argomento 'attaccato' alla misura, per esempio `3,9\unit{k\ohm}` che produce 3,9 kΩ.[6]

4. I pedici e gli apici numerici vanno scritti con carattere diritto, ma quelli letterali vanno scritti in corsivo matematico se rappresentano quantità variabili, o in tondo se rappresentano delle apposizioni della grandezza: quindi $V_{\max}$ e non V_{max}; invece bisogna scrivere V_T se T è, per esempio, una temperatura; V_i se i rappresenta l'indice dell'i-esimo elemento di un insieme numerabile, ma V_{i} se 'i' sta per 'iniziale'. Per questo scopo *babel* in italiano offre il comando `\ped` per scrivere pedici in tondo; e `\ap` per scrivere apici in tondo; entrambi i comandi funzionano anche in modo testo per inserire pedici o apici, ma in modo testo mantengono il font in uso, senza preoccuparsi se sia diritto o inclinato.

Una tipica equazione fisica potrebbe essere:

$$i(t) = \sqrt{2} I_{\mathrm{eff}}\, \mathbf{Re}\left[\mathrm{e}^{\mathrm{i}\,\omega t}\right]$$

dove 'eff' sta per 'efficace', 'e' è la costante matematica base dei logaritmi naturali da distinguere dalla 'carica elettrica elementare e', e 'i' è l'unità immaginaria da distinguere dalla grandezza fisica 'corrente elettrica i'.

La figura 13.1 riprende due strafalcioni giornalistici pubblicati su un quotidiano nazionale; nella figura di sinistra c'è l'errore frequentissimo di scrivere 'K', invece di 'k' per il prefisso 'kilo'; in effetti in quella figura si parla di kelvin metri

[5] Il comando `\unit` non è disponibile di default, ma, se si scrive in italiano, deve eventualmente essere reso disponibile dando nel preambolo il comando `\setISOcompliance`. Per contro, il modulo *italian* di *babel* si disinteressa di queste impostazioni se nel preambolo sono stati caricati i pacchetti che mettono a disposizioni dell'utente altri comandi per gestire correttamente le unità di misura in conformità con le norme ISO, per esempio il file di estensione *siunitx*.

[6] In modo matematico bisognerebbe scrivere `\Omega`, mentre in modo testo bisognerebbe scrivere `\textohm`. Non è difficile definire un comando `\ohm` che distingua che cosa si deve usare a seconda se si è in modo matematico o in modo testo, come è stato fatto in questo documento nella pagina 479.

Figura 13.1: Due esempi di strafalcioni giornalistici

all'ora... Nella figura a destra 'M/bit al secondo' doveva essere[7] M bit/s o forse Mi bit/s. Ammesso che si tratti solo di un errore di stampa, con una velocità di trasmissione di 75 Mibit/s per trasmettere il contenuto di un DVD completo (che generalmente contiene più di 4,5 Gibyte, cioè più di 36 Gibit) occorrono almeno 480 s cioè almeno 8 min, tutt'altro che pochi secondi...

Chi compone i suoi testi con LaTeX è tenuto ad astenersi da una simile ignoranza e sciatteria!

Egli, però, può trarre giovamento dall'uso dei pacchetti *units*, *SIunits* e *siunitx*. Il primo pacchetto definisce una versione del comando `\unit` (diversa da quella che è definita con l'opzione *italian* del pacchetto *babel*) che consente di specificare alcune opzioni di spaziatura fra la misura e l'unità di misura e consente di scrivere le frazioni mediante numeratore rialzato e denominatore ribassato, utili sia in linea con il testo, sia in matematica, affinché risulti ben chiaro chi sia il numeratore e chi il denominatore; si può quindi specificare una velocità nella forma $70\,^{\mathrm{km}}/_{\mathrm{h}}$; questo modo di scrivere è una pratica diffusa nei paesi dell'Europa del Nord, ma non è previsto dalle norme ISO.

Il secondo pacchetto contiene una bella e aggiornata descrizione della situazione relativa al Sistema Internazionale (SI) di unità di misura e richiama anche le norme tipografiche specificate dalla Conferenza Generale dei Pesi e delle Misure e dalle sue commissioni, nonché dalle norme ISO che regolano questa questione. Il pacchetto fornisce anche una diversa definizione del comando `\unit` che richiede però due argomenti e quindi è incompatibile con la definizione dell'opzione *italian* di *babel*; all'interno del secondo argomento è possibile specificare le unità di

[7] I prefissi stabiliti dalle norme ISO dal kilo in poi rappresentano potenze di 1000. In informatica è più conveniente fare riferimento alle potenze di 2; per indicare i multipli normalmente si usano gli stessi prefissi ISO (con k mutato in K) seguiti da una 'i'; essi rappresentano le potenze di $2^{10} = 1024$. Il prefisso 'Mi' indica quindi $(2^{10})^2$ e il prefisso 'Gi' indica il valore $(2^{10})^3$. Inoltre il simbolo né il bit né il byte sono codificati come unità di misura del SI; sarebbe opportuno scrivere queste grandezze per disteso, piuttosto che con 'b' e 'B', anche se gli informatici spesso usano queste abbreviazioni, che si prestano a essere fraintese; per esempio le compagnie telefoniche che concedono in uso le loro linee per collegamenti ADSL, le reclamizzano spesso con velocità strepitose, per esempio di 200 MB, o 200 mb, in entrambi i casi dimenticando di rispettare le norme; in realtà quelle velocità indicano il valore massimo di trasmissione di 200 Mi bit/s lasciando credere ai clienti che si tratti di 200 Mi byte/s, quando invece sono otto volte più basse.

misura a parole (in inglese britannico e al singolare) per cui è possibile indicare la velocità di 70 km/h con la scrittura:

```
\unit{70}{\kilo\metre\per\hour}
```

Invece con il comando definito con l'opzione *italian* di `babel` basta scrivere `70\unit{km/h}`.

Usando il pacchetto `SIunits` è possibile specificargli l'opzione *italian* cosicché si evita la collisione con la definizione di `\unit` in `babel`, perché la definizione del pacchetto `SIunits` diventa `\unita` (in italiano e senza accento) e i due comandi con le rispettive sintassi possono essere usati contemporaneamente. Viceversa, se questa soluzione non interessa, basta caricare il pacchetto `babel` *dopo* il pacchetto `SIunits` perché `babel` è predisposto affinché non ridefinisca il comando `\unit` di `SIunits`.

Lo stesso autore di `SIunits`, D.N.J. Els, ha preparato un secondo pacchetto, `SIstyle`, per comporre correttamente le indicazioni metriche delle grandezze fisiche; secondo questo pacchetto il comando per comporre la grandezza fisica è semplicemente:

`\SI{`⟨*misura*⟩`}{`⟨*unità di misura*⟩`}`

Il pregio del pacchetto, oltre alla semplicità d'uso, che fa dire ad Els di preferire egli stesso questo pacchetto al suo più esteso pacchetto `SIunits`, consiste nelle vaste possibilità di personalizzazione e di adeguamento a specifiche norme nazionali; serve, cioè, ad impostare lo stile di scrittura delle grandezze in modo da assicurare l'uniformità di composizione all'interno del documento.

Va infine segnalato anche il recentissimo quarto pacchetto `siunitx` che sostituisce tutti i precedenti e che offre molte possibilità in più per gestire le grandezze fisiche in termini numerici; dispone di un modulo di elaborazione dei numeri che consente anche scritture del tipo `30e3` e che le macro di scrittura trasformano in 30×10^3; questo permette di trarre i valori numerici da file scritti dagli stessi strumenti di misura moderni, dove i valori numerici sono espressi con la notazione informatica dei numeri a virgola mobile. Si può specificare il numero di cifre da scrivere nei valori numerici delle misure: il modulo di elaborazione dei numeri provvede ad arrotondare quel valore numerico al numero richiesto di decimali. Infatti se si specifica che si vogliono consistentemente 4 decimali e la virgola decimale, il numero `1.234567` viene stampato nella forma 1,2346 dove il numero di decimali è quello voluto ma l'ultima cifra tiene conto dell'arrotondamento in alto, visto che la parte scartata è maggiore della metà dell'unità corrispondente all'ultima cifra scritta; inoltre il punto decimale è consistentemente cambiato nella virgola, come richiesto. Si rinvia il lettore alla documentazione in `.../doc/latex/siunitx/siunitx.pdf`.

Il compositore stia bene attento alla preposizione 'per', che in italiano significa 'moltiplicato' e in inglese significa 'diviso'; una resistività di 'sette microohm per metro' si scrive $7\,\mu\Omega\,\mathrm{m}$ oppure $7\,\mu\Omega \cdot \mathrm{m}$, ma sarebbe sbagliatissimo scriverla $7\,\mu\Omega/\mathrm{m}$ interpretando la preposizione 'per' all'inglese.

Capitolo 14

LaTeX: la matematica avanzata

Questo capitolo è dedicato a chi ha bisogno di scrivere espressioni matematiche un po' più complesse di quelle che si possono comporre con i comandi base di LaTeX esposti nel capitolo precedente.

È comodo fare uso del pacchetto di estensione *amsmath*. Questa estensione è il frutto della collaborazione con l'American Mathematical Society (AMS) che, dai tempi del Plain TeX originale, anzi del TeX 78, la primissima versione del sistema TeX, ha sempre collaborato ed è sempre stata strenua sostenitrice di questo software, ne ha sponsorizzato il 'trade mark', ed è stata la prima società scientifica ad accettare i contributi alle proprie riviste scritti in Plain TeX, in $\mathcal{AMS}$ TeX e infine in $\mathcal{AMS}$ LaTeX; vale a dire con i programmi standard estesi con i pacchetti preparati dai programmatori della Società stessa e poi messi a disposizione di tutta la comunità degli utenti. L'American Mathematical Society ha anche messo a disposizione i file di classe specializzati per scrivere articoli destinati alle sue varie pubblicazioni; ora questa pratica è seguita da tutte le più importanti riviste scientifiche internazionali; anche gli atti di molti congressi sono scritti usando direttamente i file sorgente degli autori delle memorie presentate a quei congressi.

Il pacchetto principe per l'utente generico è il pacchetto *amsmath*. Questo pacchetto mette a disposizione una miriade di comandi nuovi, da quelli che servono per usare altri simboli, oltre a quelli già molto numerosi presenti normalmente in LaTeX, a quelli che servono per predisporre incolonnamenti particolari, a quelli che servono per comporre segni o strutture matematiche avanzate.

14.1 I simboli di *amsmath*

Il pacchetto *amsmath* consente di accedere ai simboli delle tabelle 14.1 e 14.2.

\ulcorner	⌜	\urcorner	⌝	\llcorner	⌞
\lrcorner	⌟	\dashrightarrow	⇢	\dashleftarrow	⇠
\widehat	^	\widetilde	~	\dasharrow	⇢
\rightleftharpoons	⇌	\leftrightharpoons	⇋	\angle	∠
\hbar	ℏ	\sqsubset	⊏	\sqsupset	⊐
\mho	℧	\square	□	\lozenge	◊
\vartriangleright	▷	\vartriangleleft	◁	\trianglerighteq	⊵
\triangleleftteq	⊴	\circleddash	⊝	\boxdot	⊡
\boxplus	⊞	\boxtimes	⊠	\boxminus	⊟
\blacksquare	■	\centerdot	·	\blacklozenge	⧫
\circlearrowright	↻	\circlearrowleft	↺	\Vdash	⊩
\Vvdash	⊪	\vDash	⊨	\twoheadrightarrow	↠
\twoheadleftarrow	↞	\leftleftarrows	⇇	\rightrightarrows	⇉
\upuparrows	⇈	\downdownarrows	⇊	\upharpoonright	↾
\downharpoonright	⇂	\upharpoonleft	↿	\downharpoonleft	⇃
\rightarrowtail	↣	\leftarrowtail	↢	\leftrightarrows	⇆
\rightleftarrows	⇄	\Lsh	↰	\Rsh	↱
\rightsquigarrow	⇝	\leftrightsquigarrow	↭	\looparrowleft	↫
\looparrowright	↬	\circeq	≗	\succsim	≿
\gtrsim	≳	\gtrapprox	⪆	\multimap	⊸
\therefore	∴	\because	∵	\doteqdot	≑
\triangleq	≜	\precsim	≾	\lesssim	≲
\lessapprox	⪅	\eqslantless	⪕	\eqslantgtr	⪖
\curlyeqprec	⋞	\curlyeqsucc	⋟	\preccurlyeq	≼
\leqq	≦	\leqslant	⩽	\lessgtr	≶
\backprime	‵	\risingdotseq	≓	\fallingdotseq	≒
\succcurlyeq	≽	\geqq	≧	\geqslant	⩾
\gtrless	≷	\vartriangleright	▷	\vartriangleleft	◁
\trianglerighteq	⊵	\triangleleftteq	⊴	\bigstar	★
\between	≬	\blacktriangledown	▼	\blacktriangleright	▶
\blacktriangleleft	◀	\vartriangle	△	\blacktriangle	▲
\triangledown	▽	\eqcirc	≖	\lesseqgtr	⋚
\gtreqless	⋛	\lesseqqgtr	⪋	\gtreqqless	⪌
\Rrightarrow	⇛	\Lleftarrow	⇚	\veebar	⊻
\barwedge	⊼	\doublebarwedge	⩞	\measuredangle	∡
\sphericalangle	∢	\varpropto	∝	\smallsmile	⌣
\smallfrown	⌢	\Subset	⋐	\Supset	⋑
\Cup	⋓	\Cap	⋒	\curlywedge	⋏
\curlyvee	⋎	\leftthreetimes	⋋	\rightthreetimes	⋌
\subseteqq	⫅	\supseteqq	⫆	\bumpeq	≏
\Bumpeq	≎	\lll	⋘	\ggg	⋙
\circledS	Ⓢ	\pitchfork	⋔	\dotplus	∔
\backsim	∽	\backsimeq	⋍	\complement	∁
\intercal	⊺	\circledcirc	⊚	\circledast	⊛

Tabella 14.1: Prima serie di simboli accessibili con il pacchetto *amsmath*

\lvertneqq	$\lvertneqq$	\gvertneqq	$\gvertneqq$	\nleq	$\nleq$
\ngeq	$\ngeq$	\nless	$\nless$	\ngtr	$\ngtr$
\nprec	$\nprec$	\nsucc	$\nsucc$	\lneqq	$\lneqq$
\gneqq	$\gneqq$	\nleqslant	$\nleqslant$	\ngeqslant	$\ngeqslant$
\lneq	$\lneq$	\gneq	$\gneq$	\npreceq	$\npreceq$
\nsucceq	$\nsucceq$	\precnsim	$\precnsim$	\succnsim	$\succnsim$
\lnsim	$\lnsim$	\gnsim	$\gnsim$	\nleqq	$\nleqq$
\ngeqq	$\ngeqq$	\precneqq	$\precneqq$	\succneqq	$\succneqq$
\precnapprox	$\precnapprox$	\succnapprox	$\succnapprox$	\lnapprox	$\lnapprox$
\gnapprox	$\gnapprox$	\nsim	$\nsim$	\ncong	$\ncong$
\diagup	$\diagup$	\diagdown	$\diagdown$	\varsubsetneq	$\varsubsetneq$
\varsupsetneq	$\varsupsetneq$	\nsubseteqq	$\nsubseteqq$	\nsupseteqq	$\nsupseteqq$
\subsetneqq	$\subsetneqq$	\supsetneqq	$\supsetneqq$	\varsubsetneqq	$\varsubsetneqq$
\varsupsetneqq	$\varsupsetneqq$	\subsetneq	$\subsetneq$	\supsetneq	$\supsetneq$
\nsubseteq	$\nsubseteq$	\nsupseteq	$\nsupseteq$	\nparallel	$\nparallel$
\nmid	$\nmid$	\nshortmid	$\nshortmid$	\nshortparallel	$\nshortparallel$
\nvdash	$\nvdash$	\nVdash	$\nVdash$	\nvDash	$\nvDash$
\nVDash	$\nVDash$	\ntrianglerighteq	$\ntrianglerighteq$	\ntrianglelefteq	$\ntrianglelefteq$
\ntriangleleft	$\ntriangleleft$	\ntriangleright	$\ntriangleright$	\nleftarrow	$\nleftarrow$
\nrightarrow	$\nrightarrow$	\nLeftarrow	$\nLeftarrow$	\nRightarrow	$\nRightarrow$
\nLeftrightarrow	$\nLeftrightarrow$	\nleftrightarrow	$\nleftrightarrow$	\divideontimes	$\divideontimes$
\varnothing	$\varnothing$	\nexists	$\nexists$	\Finv	$\Finv$
\Game	$\Game$	\eth	$\eth$	\eqsim	$\eqsim$
\beth	$\beth$	\gimel	$\gimel$	\daleth	$\daleth$
\lessdot	$\lessdot$	\gtrdot	$\gtrdot$	\ltimes	$\ltimes$
\rtimes	$\rtimes$	\shortmid	$\shortmid$	\shortparallel	$\shortparallel$
\smallsetminus	$\smallsetminus$	\thicksim	$\thicksim$	\thickapprox	$\thickapprox$
\approxeq	$\approxeq$	\succapprox	$\succapprox$	\precapprox	$\precapprox$
\curvearrowleft	$\curvearrowleft$	\curvearrowright	$\curvearrowright$	\digamma	$\digamma$
\varkappa	$\varkappa$	\Bbbk	$\Bbbk$	\hslash	$\hslash$
\backepsilon	$\backepsilon$				

Tabella 14.2: Seconda serie di simboli accessibili con il pacchetto *amsmath*

Se insieme a *amsmath* si invoca anche *amssymb* si hanno disponibili tra gli altri anche i vecchi segni di LaTeX 2.09, forniti anche dal pacchetto *ltxsymb* il cui uso, come si è avuto occasione di commentare, è oggi sconsigliato. D'altra parte *amssymb* invoca a sua volta *amsfonts* che mette a disposizione anche altri font, in particolare i font 'Euler fraktur' che in matematica possono talvolta risultare più efficaci dei caratteri calligrafici.

Invece, se non occorre tutto il macchinario di *amsmath* ma occorrono solo alcuni simboli, in particolare le lettere maiuscole ad aste raddoppiate, si può usare solo il pacchetto *amsfonts* che definisce il comando `\mathbb` per usarle; tuttavia si consiglia di caricare solo il pacchetto *amssymb* che a sua volta carica *amsfonts*. Qui, nelle tabelle citate, si è riportato tutto l'insieme dei simboli componibili per mezzo di tutti i pacchetti associati a *amsmath* in modo che

il lettore abbia la consapevolezza della potenza del sistema TeX per quanto riguarda la simbologia disponibile.

14.2 Le estensioni dei font matematici

La AMS ha messo a disposizione due altre polizze di font chiamate cripticamente `msam` e `msbm` i cui simboli sono direttamente accessibili con il pacchetto *`amssymb`*.

Nella tabella 14.2 si notano diversi operatori binari 'negati'; i font normali di LaTeX contengono anche il qualificatore `\not` che serve per creare per sovrapposizione[1] la negazione di qualsiasi operatore binario.

In linea di principio sarebbe desiderabile usare sempre i segni negati della tabella 14.2; talvolta però bisogna fare di necessità virtù e ricorrere a `\not`; per esempio non c'è preconfezionato il simbolo di 'non appartenenza', `\notin`; questo si può ottenere negando l'operatore `\in`: $a \notin b$, e il risultato appare pienamente accettabile. In altri casi, come per esempio per l'operatore 'non esiste', si può negare l'operatore 'esiste',`\exists`, ma se si fa così si ottiene: $\not\exists x$. Se invece si usa direttamente l'operatore `\nexists` il risultato è molto migliore: $\nexists x$, anzi, è perfetto.

Un'altra cosa a cui bisogna fare attenzione sono gli spazi che precedono e seguono gli operatori di vario genere a seconda di ciò che li precede o li segue; il discorso può risultare abbastanza complesso, ma è completamente descritto nel TeXbook, che riporta anche la tabella completa per tutte le combinazioni di accostamento dei vari operatori e dei vari simboli matematici. Generalmente lo spazio viene azzerato quando due operatori dello stesso tipo si susseguono; per esempio, per dire che l'ordinamento implicato dall'operatore $\leq$ viene esteso dall'ordinamento implicato da $\preceq$, non è corretto scrivere `\( \leq \subseteq \preceq \)`, ottenendo $\leq\subseteq\preceq$, perché gli spazi sono sbagliati; bisogna invece scrivere `\( {\leq} \subseteq {\preceq} \)`, perché le parentesi di gruppo nascondono all'operatore intermedio la natura del loro contenuto; infatti si ottiene: $\leq \subseteq \preceq$.

Questi sono punti delicati, forse troppo tecnici per la natura di questo testo, ma ricordiamoci: la tipografia è un'arte e la composizione tipografica della matematica ne costituisce un livello superiore. LaTeX può fare di tutto da solo, ma qualche volta ha bisogno di un piccolo aiuto da parte dell'utente.

14.3 I sistemi di equazioni

I sistemi di equazioni sono formati da un certo numero di equazioni incolonnate una sopra l'altra, di solito allineate in modo che i segni di uguaglianza (o qualunque altro segno di relazione) siano incolonnati verticalmente.

[1] Si veda il capitolo 22 dove si spiega che il comando `\not` va modificato in modo da renderlo compatibile con il formato archiviabile; un altro segno che va modificato è `\mapsto`.

LaTeX è nato con un ambiente *eqnarray* (anche asteriscato) che consente di eseguire questo incolonnamento; per esempio:

$$\begin{aligned} 3x + 7y - 2z &= 13 && (14.1)\\ 2y + 3z &= 12 && (14.2)\\ 5x - 3y + 4z &= 6 && (14.3) \end{aligned}$$

in cui le equazioni sono numerate da 14.1 a 14.3 e vi si può fare riferimento singolarmente usando il comando `\label` alla fine di ciascuna riga. Il sistema esposto è stato scritto con il codice

```
\begin{eqnarray}
  3x + 7y - 2z &=& 13 \label{equ:sistema1}\\
       2y + 3z &=& 12 \label{equ:sistema2}\\
  5x - 3y + 4z &=& 6  \label{equ:sistema3}
\end{eqnarray}
```

Se invece di usare l'ambiente *eqnarray* si fosse usato l'ambiente *eqnarray** le equazioni non sarebbero state numerate.

Purtroppo questo ambiente è nato con un 'peccato originale'; se il lettore osserva bene, constata che lo spazio a destra e a sinistra dei segni di uguaglianza nelle equazioni 14.1–14.3 è maggiore che non nelle equazioni del capitolo precedente. Non si sa se questo fosse voluto o se sia successo per caso, ma quando dal vecchio LaTeX 2.09 si è passati al nuovo LaTeX 2_ε, questo difetto è stato conservato, apparentemente per mantenere una compatibilità con i file predisposti per essere composti con la vecchia versione.

Sebbene non sarebbe difficile correggere la definizione del comando di apertura dell'ambiente *eqnarray* per disporre della spaziatura giusta ricorrendo alle funzionalità del pacchetto *xpatch*, oggi gli ambienti *eqnarray* e *eqnarray** non dovrebbero venire più usati, ma si dovrebbero usare sempre solo gli ambienti predisposti dal pacchetto *amsmath*.

14.4 Gli ambienti di composizione di *amsmath*

Il pacchetto *amsmath* mette a disposizione una serie di ambienti per comporre e/o incolonnare equazioni o per raccoglierle in modo ordinato, ovvero per scriverle su più righe in modo professionale, anche se l'intervento dell'autore per conoscere il ritmo delle espressioni matematiche è fondamentale.

Gli ambienti disponibili sono raggruppati nella tabella 14.3.

Tutti gli ambienti hanno la versione asteriscata che differisce da quella non asteriscata per la mancanza delle numerazione delle equazioni; solo *split* e *aligned* ne sono privi perché devono essere usati all'interno degli altri ambienti, non possono essere usati da soli. *subequations* invece ne è privo perché serve per numerare le equazioni in un modo particolare.

equation	*equation**	*align*	*align**
gather	*gather**	*flalign*	*flalign**
multline	*multline**	*alignat*	*alignat**
aligned	*split*	*subequations*	

Tabella 14.3: Gli ambienti di allineamento di `amsmath`

La numerazione può anche essere esclusa inserendo il comando `\notag` prima del segno di 'a capo' di ogni riga dell'allineamento; alternativamente il comando `\tag` consente di inserire una etichetta a propria scelta come per esempio nell'equazione:

$$a^2 + b^2 = c^2 \qquad \text{(Teorema di Pitagora)}$$

ottenuto scrivendo:

```
\begin{equation}
a^2 +b^2 = c^2 \tag{Teorema di Pitagora}
\end{equation}
```

Tutti gli ambienti di incolonnamento usano il segno `&` per passare da una colonna all'altra e non vogliono mai due segni a cavallo degli operatori di relazione, come succede con *eqnarray*, ma ne vogliono uno solo *prima* dell'operatore.

14.4.1 L'ambiente *equation*

Di *equation* si è già detto quanto basta; il suo compagno asteriscato, a parte la numerazione, consente di contenere tutti gli altri ambienti che possono essere inclusi negli ambienti numerabili, in particolare *split* per scrivere lunghe equazioni su più righe. La presenza di *equation** consente di evitare di usare *displaymath*; quest'ultimo infatti non garantisce le spaziature corrette e omogenee con quelle degli altri ambienti matematici. La definizione mediante l'asterisco è anche più coerente con l'uso dell'asterisco negli altri ambienti numerabili, ma di cui si desidera eliminare la numerazione.

14.4.2 L'ambiente *aligned*

L'ambiente *aligned* serve per comporre un allineamento di equazioni o di espressioni matematiche all'interno di un altro ambiente che sia in grado, eventualmente, di assegnare un numero a questo allineamento. Per esempio si può scrivere:

$$\begin{aligned}\operatorname{Ker}_n(x) = &-[\log(x/2)+\gamma] + \tfrac{1}{4}\pi \operatorname{Bei}_n(x) \\ &+ \frac{1}{2}\sum_{k=0}^{n-1} \frac{(n-k-1)!(x/2)^{2k-n}}{k!} \cos\frac{(3n+2k)\pi}{4} \\ &+ \frac{1}{2}\sum_{k=0}^{\infty} \frac{(x/2)^{n+2k}}{k!(n+k)!}[\Phi(k)+\Phi(n+k)] \cos\frac{(3n+2k)\pi}{4}\end{aligned} \tag{14.5}$$

Questo ambiente produce risultati molto simili a quelli dell'ambiente *split* a cui si rinvia per i commenti.

14.4.3 L'ambiente *split*

L'ambiente *split* serve per dividere una lunga espressione matematica fra più righe.

L'equazione su più righe 14.5, resa con *aligned*, con *split* diventa:

$$\begin{aligned}\mathrm{Ker}_n(x) = &-[\log(x/2)+\gamma] + \tfrac{1}{4}\pi\,\mathrm{Bei}_n(x)\\ &+\frac{1}{2}\sum_{k=0}^{n-1}\frac{(n-k-1)!(x/2)^{2k-n}}{k!}\cos\frac{(3n+2k)\pi}{4}\\ &+\frac{1}{2}\sum_{k=0}^{\infty}\frac{(x/2)^{n+2k}}{k!(n+k)!}[\Phi(k)+\Phi(n+k)]\cos\frac{(3n+2k)\pi}{4}\end{aligned} \tag{14.6}$$

Il risultato è stato ottenuto con

```
\begin{equation}
  \begin{split}
    \Ker_n(x) & = -[\log(x/2)+\gamma]
                          +{\textstyle\frac{1}{4}}\pi\Bei_n(x)\\
              & \quad+\frac{1}{2}\sum_{k=0}^{n-1}
                        \frac{(n-k-1)!(x/2)^{2k-n}}{k!}
                                        \cos\frac{(3n+2k)\pi}{4}\\
              & \qquad+\frac{1}{2}\sum_{k=0}^\infty
                \frac{(x/2)^{n+2k}}{k!(n+k)!}[\Phi(k)+\Phi(n+k)]
                                         \cos\frac{(3n+2k)\pi}{4}
  \end{split}
  \label{equ:Ker2}
\end{equation}
```

La differenza fra *split* e *aligned* è che il primo ambiente, oltre a consentire allineamenti verticali, come fa anche il secondo ambiente, produce un risultato finale che, a parte il suo contenuto, ha esattamente la larghezza della riga in cui compare, mentre *aligned* produce il suo risultato finale sotto forma di un oggetto che è largo quanto il suo contenuto. In tal modo questo oggetto può costituire un elemento di costruzione di un oggetto più grande. Questi due ambienti, inoltre, differiscono dall'ambiente *multline* perché consentono degli allineamenti verticali, mentre *multline* non lo consente, come si vede ne paragrafo seguente.

14.4.4 L'ambiente *multline*

La stessa equazione su tre righe composta con l'ambiente *multline* diventa:

$$\begin{aligned}\mathrm{Ker}_n(x) = &-[\log(x/2)+\gamma] + \tfrac{1}{4}\pi\,\mathrm{Bei}_n(x)\\ &+\frac{1}{2}\sum_{k=0}^{n-1}\frac{(n-k-1)!(x/2)^{2k-n}}{k!}\cos\frac{(3n+2k)\pi}{4}\\ &+\frac{1}{2}\sum_{k=0}^{\infty}\frac{(x/2)^{n+2k}}{k!(n+k)!}[\Phi(k)+\Phi(n+k)]\cos\frac{(3n+2k)\pi}{4}\end{aligned}\tag{14.7}$$

con il codice sorgente:

```
\begin{multline}
  \Ker_n(x)  = -[\log(x/2)+\gamma]
                       +{\textstyle\frac{1}{4}}\pi\Bei_n(x)\\
  +\frac{1}{2}\sum_{k=0}^{n-1}
               \frac{(n-k-1)!(x/2)^{2k-n}}{k!}
                                    \cos\frac{(3n+2k)\pi}{4}\\
  +\frac{1}{2}\sum_{k=0}^\infty
       \frac{(x/2)^{n+2k}}{k!(n+k)!}[\Phi(k)+\Phi(n+k)]
                                      \cos\frac{(3n+2k)\pi}{4}
\end{multline}
```

Si noti in particolare la mancanza dei segni di incolonnamento `&` e la mancanza dei comandi di spaziatura `\quad` e `\qquad` presenti negli esempi precedenti; infatti `multline` compone la prima riga allineata con il margine sinistro delle equazioni; allinea l'ultima espressione con il margine destro tenendo conto del numero dell'equazione; infine allinea tutte le altre righe intermedie in modo da centrarle nello spazio disponibile. Se il numero dell'equazione è a destra, è in linea con l'ultima riga, oppure sotto all'ultima riga, se non ci fosse sufficiente spazio; se si decide di comporre con i numeri delle equazioni a sinistra, il numero è allineato con la prima riga oppure è collocato prima della prima riga; in entrambi i casi non c'è mai sovrapposizione fra il numero e le espressioni.

Questa gestione intelligente della collocazione del numero dell'equazione, in modo da non sovrapporsi mai all'equazione stessa, è comune a tutti gli ambienti matematici numerabili prodotti da *amsmath*; questa gestione intelligente già da sola basterebbe a giustificare l'uso sistematico di *amsmath* e dei suoi compagni ogni qual volta si debbano scrivere testi con equazioni anche semplici (prive di segni insoliti) ma lunghe.

È necessario richiamare l'attenzione su come si dividano le espressioni matematiche fra più righe; il punto di cesura va messo *prima* di un operatore di relazione o di un operatore binario, senza ripeterlo alla fine della riga precedente, come è già stato detto. Ma se da un lato non sarebbe opportuno spezzare una espressione di "grandi" dimensioni racchiusa fra parentesi, talvolta è necessario farlo. In questo modo diventa difficile usare correttamente i comandi

`\left` e `\right` per rendere automatico il dimensionamento dei delimitatori, proprio a causa dell'espressione spezzata fra più righe. È quindi necessario rinunciare agli automatismi offerti da LaTeX e ricorrere ai comandi manuali `\bigl`, `\bigr`, eccetera, come mostrato nella pagina 310. Anche se gli ambienti di *amsmath* aiutano moltissimo per comporre espressioni matematiche spezzate, sta all'autore oltre che al "tastierista" trovare il giusto ritmo di divisione di queste lunghe espressioni e di racchiuderle fra le parentesi di dimensioni corrette.

14.4.5 L'ambiente *gather*

L'ambiente *gather* serve per raccogliere ordinatamente da due a più equazioni una per riga, senza nessun incolonnamento particolare ma centrando le varie equazioni; in un certo senso potrebbe essere visto come una sequenza di ambienti *equation*; con questo ambiente si possono raccogliere le equazioni di Maxwell, per esempio, nella forma:

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \tag{14.8}$$

$$\nabla \times \vec{H} = \frac{\partial \vec{D}}{\partial t} + \vec{J} \tag{14.9}$$

$$\nabla \cdot \vec{D} = \rho \tag{14.10}$$

$$\nabla \cdot \vec{B} = 0 \tag{14.11}$$

ottenuta scrivendo

```
\begin{gather}
  \nabla\times\vec{E}=-\frac{\partial\vec{B}}{\partial t}\\
  \nabla\times\vec{H}=
                     \frac{\partial\vec{D}}{\partial t}+\vec{J}\\
  \nabla \cdot\vec{D}= \rho\\
  \nabla \cdot\vec{B}= 0
\end{gather}
```

14.4.6 L'ambiente *align*

Al contrario, usando l'ambiente *align* le stesse equazioni di Maxwell diventano:

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \tag{14.12}$$

$$\nabla \times \vec{H} = \frac{\partial \vec{D}}{\partial t} + \vec{J} \tag{14.13}$$

$$\nabla \cdot \vec{D} = \rho \tag{14.14}$$

$$\nabla \cdot \vec{B} = 0 \tag{14.15}$$

ottenute scrivendo:

```
\begin{align}
  \nabla\times\vec{E}&= -\frac{\partial\vec{B}}{\partial t}\\
  \nabla\times\vec{H}&=
                    \frac{\partial\vec{D}}{\partial t}+\vec{J}\\
  \nabla \cdot\vec{D}&= \rho\\
  \nabla \cdot\vec{B}&= 0
\end{align}
```

Si nota che l'unica differenza consiste nel comando di incolonnamento `&` inserito prima di ciascun segno di uguaglianza.

All'interno di un ambiente *align* possono comparire diversi incolonnamenti, non uno solo come nell'esempio precedente; per esempio le equazioni di Maxwell potrebbero venire riorganizzate così:

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \qquad \nabla \cdot \vec{D} = \rho \tag{14.16}$$

$$\nabla \times \vec{H} = \frac{\partial \vec{D}}{\partial t} + \vec{J} \qquad \nabla \cdot \vec{B} = 0 \tag{14.17}$$

usando la scrittura:

```
\begin{align}
  \nabla\times\vec{E}&=-\frac{\partial\vec{B}}{\partial t}
                    &\nabla \cdot\vec{D}&=\rho\\
  \nabla\times\vec{H}&=\frac{\partial\vec{D}}{\partial t}+\vec{J}
                    &\nabla \cdot\vec{B}&=0
\end{align}
```

14.4.7 L'ambiente *flalign*

L'ambiente *flalign* differisce dall'ambiente *align* solo per il fatto che le espressioni allineate vengono spostate al massimo all'esterno della riga dove compaiono:

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \qquad \nabla \cdot \vec{D} = \rho \tag{14.18}$$

$$\nabla \times \vec{H} = \vec{J} + \frac{\partial \vec{D}}{\partial t} \qquad \nabla \cdot \vec{B} = 0 \tag{14.19}$$

e si vede chiaramente che questo ambiente va meglio quando le equazioni non sono numerate e quando le espressioni sono sufficientemente lunghe; nelle altre circostanze è meglio usare *align*. Se le espressioni sono più di due in ogni riga lo spazio viene usato meglio, ma resta il fatto che i numeri delle equazioni diventano ancora più 'superflui'.

14.4.8 L'ambiente *alignat*

L'ambiente *alignat* è un po' particolare, perché accetta un numero come argomento del comando di apertura che specifica quanti punti di allineamento compaiono in ogni riga; la regola dice: conta il massimo numero di segni `&` che compaiono in ogni riga, aggiungi 1 e dividi per 2. Così l'esempio del manuale diventa:

$$\begin{aligned} x &= y_1 - y_2 + y_3 - y_5 + y_8 - \dots && \text{by eq. (3.21)} && (14.20)\\ &= y' \circ y^* && \text{by eq. (4.1)} && (14.21)\\ &= y(0)y' && \text{by Axiom 1} && (14.22) \end{aligned}$$

prodotto con il codice:

```
\begin{alignat}{2}
  x&= y_1 - y_2 + y_3 -y_5 +y_8 -\dots
                        &\quad & \text{by eq.~(3.21)}\\
   &=y'\circ y^*        &      & \text{by eq.~(4.1)}\\
   &= y(0)y'            &      & \text{by Axiom 1}
\end{alignat}
```

e nel quale si osserva la presenza del comando `\text`; questo comando serve per scrivere del testo all'interno di espressioni matematiche; se questo testo contiene una espressione matematica in linea, questa va esplicitamente racchiusa fra i delimitatori `\(` e `\)`.

14.4.9 L'ambiente *subequations*

L'ambiente *subequations* è particolare perché le equazioni all'interno di questo ambiente vengono numerate con un contatore subalterno così da produrre numeri del tipo (10.22a), (10.22b) eccetera. Per esempio, il sistema di proprietà condizionate:

$$\begin{aligned} \mathbf{Im}[F(\sigma + \mathrm{i}\,0)] &= 0 && \forall \sigma > 0 && (14.23a)\\ F(p) &\neq \infty && \forall \sigma > 0 && (14.23b)\\ \lim_{p \to \mathrm{i}\,\omega_0} (p - \mathrm{i}\,\omega_0) F(p) &= k && \forall \omega_0 \text{ e con } 0 \leq k < \infty && (14.23c)\\ \lim_{p \to \infty} F(p)/p &= h && \text{con } 0 \leq h < \infty && (14.23d)\\ \mathbf{Re}[F(0 + \mathrm{i}\,\omega)] &\geq 0 && \forall \omega && (14.23e) \end{aligned}$$

è ottenuto con il codice seguente:

```
\DeclareMathOperator{\uimm}{\mathrm{i}}          % nel preambolo
...
\begin{subequations}
                                                 \label{equ:sube}
  \begin {align}
```

```
    \Im[F(\sigma+\uimm 0)]  &= 0     &\forall\sigma>0\\
    F(p)                    &\not=\infty&\forall\sigma>0\\
    \lim_{p\to\uimm\omega_0}(p-\uimm\omega_0)F(p)&=k
                            & \forall\omega_0\text{ e con }
                                        0\le k<\infty\\
    \lim_{p\to\infty} F(p)/p&=h      &\text{con }0\le h<\infty\\
    \Re[F(0+\uimm\omega)]   &\ge 0   &\forall\omega
                                          \label{equ:sube5}
 \end{align}
\end{subequations}
```

Con il comando `\label` inserito prima dell'ambiente *align*, ma dentro all'ambiente *subequations* si recupera il numero dell'intero sistema di condizioni 14.23; invece con il comando `\label` inserito all'interno dell'ambiente *align* si recupera il 'numero' assegnato a ciascuna condizione, per esempio all'ultima condizione 14.23e si fa riferimento mediante `\ref{equ:sube5}`.

La definizione dell'unità immaginaria `\uimm` è un po' particolare nel senso che essa è definita come un operatore, invece che come una semplice variabile. Si noti che se si usasse il comando primitivo `\mathop`, la definizione dovrebbe contenere oltre alla 'i' in tondo anche uno strut o dello spazio matematico al fine di centrare la 'i' rispetto al suo asse matematico. È questa una ragione in più per ricorrere ai comandi di definizione disponibili con il pacchetto *amsmath*, che permettono di concentrarsi sul significato intrinseco di ciò che si sta definendo, senza badare ai dettagli compositivi a livello di base.

Esercizio 14.1 Il lettore provi a ripetere la composizione di questo sistema di condizioni 14.23 definendo il comando `\uimm` con `\newcommand` ed usando solamente `\mathop`. Che cosa succede?

14.4.10 Gli ambienti spezzati

Di default il pacchetto *amsmath* fa sì che gli ambienti di incolonnamento precedentemente descritti non vengano spezzati da eventuali salti di pagina, ma restino composti interamente in una sola pagina. Questo in generale è preferibile rispetto a quando si è obbligati a voltare pagina per completare la lettura di un gruppo di equazioni raccolte assieme in un uno di questi ambenti.

Tuttavia talvolta è impossibile per vari motivi mantenere tutte le espressioni di un ambiente nella stessa pagina; è sicuramente obbligatorio quando si usano gli ambienti *split*, *aligned*, *gathered*, *alignedat* dove andare ad una nuova pagina è impossibile, mentre con *multline* sarebbe possibile, ma sarebbe anche "sconveniente". Negli altri casi si può essere più tolleranti. Per questo scopo il pacchetto *amsmath* mette a disposizione due comandi:

```
\allowdisplaybreaks[⟨codice⟩]
\displaybreak[⟨codice⟩]
```

per consentire di eseguire un fine pagina fra una espressione e l'altra.

Precisamente `\allowdisplaybreaks` può essere messo nel preambolo e vale per tutti gli incolonnamenti matematici del documento; se all'interno di uno di questi ambienti *non* si vuole eseguire un cambio di pagina, invece di usare il comune comando `\\` si può usare la sua variante asteriscata `\\*` che, appunto impedisce il salto di pagina.

Invece il comando `\displaybreak` va usato solamente prima del comando `\\` dopo il quale si desidera consentire un salto di pagina.

Il ⟨*codice*⟩ per entrambi i comandi funziona alla stessa maniera dei comandi simili `\pagebreak` da usarsi nel corpo del testo. Si tratta di una cifra nell'intervallo `0`–`4`[2] che dà un peso alla "licenza" di eseguire un cambio di pagina: `0` è un blando consenso, mentre `4` è un ordine perentorio.

Vale la pena di osservare qui che il comando di fine espressione `\\` ammette la stessa sintassi che esso usa quando viene usato nel testo (vedi la sintassi esposta nella pagina 701), sia per la versione normale, sia per quella asteriscata:

```
\\[⟨spazio⟩]
\\*[⟨spazio⟩]
```

cosicché, usando il valore ⟨*spazio*⟩, si può controllare anche la spaziatura fra le equazioni.

14.5 Altri comandi e ambienti

Il pacchetto *amsmath* offre numerosi altri comandi e ambienti che consentono di comporre la matematica in modo estremamente professionale.

14.5.1 Definizione di operatori funzionali

Qui si cita nuovamente il comando per definire i nomi di nuovi operatori funzionali:

```
\DeclareMathOperator{⟨operatore⟩}{⟨definizione⟩}
```

così che se si volesse definire l'operatore per l'arcoseno iperbolico 'arsinh' (che non è definito né da LaTeX standard, né da `amsmath`) basterebbe dichiarare nel preambolo:

```
\DeclareMathOperator{\arsinh}{arsinh}
```

Il comando asteriscato serve per definire operatori funzionali con i limiti.

[2] Solo il comando `\allowdisplaybreak` non ammette il valore `0`.

14.5.2 Le frazioni in generale e le frazioni continue

14.5.2.1 Le frazioni e gli altri costrutti simili

Il pacchetto *amsmath* consente di scrivere le frazioni e gli altri costrutti simili alle frazioni in tre varianti distinte e quindi con tre comandi distinti: il comando 'normale' che compone in uno stile diverso in modo testo rispetto al modo display, il nome testuale per comporre sempre e comunque in modo testo, e il modo display per scrivere sempre e comunque in modo display; per esempio le frazioni normalmente vengono composte con il comando `\frac` ma il modo testuale viene composto con `\tfrac` e il modo display con `\dfrac`. L'operatore binomiale, per esempio si compone con `\binom`:

$$\binom{n}{k} = \frac{n(n-1)(n-2)\cdots(n-k+1)}{1\cdot 2\cdot 3\cdots k}$$

Se proprio si volesse indicare il coefficiente binomiale in modo display all'interno del testo, e si usasse `\dbinom`, si otterrebbe $\dbinom{n}{k}$; come si vede, questo è un tipo di scrittura da evitare assolutamente quando si compone matematica in linea con il testo, perché altera vistosamente lo scartamento.

14.5.2.2 Le frazioni continue

Fra i comandi utilissimi conviene citare quello per la composizione delle frazioni continue `\cfrac` con il quale si possono comporre facilmente espressioni del tipo:

$$\tanh z = \cfrac{z}{1+\cfrac{z^2}{3+\cfrac{z^2}{5+\cfrac{z^2}{7+\cfrac{z^2}{\cdots}}}}} \tag{14.24}$$

composta con i comandi:

```
\begin{equation}
  \tanh z = \cfrac{z}{1+\cfrac{z^2}{3+\cfrac{z^2}{5+
                       \cfrac{z^2}{7+\cfrac{z^2}{\cdots}}}}}
\end{equation}
```

14.5.3 Il testo intercalato alle equazioni

Un altro comando molto interessante è `\intertext`; esso consente di inserire un (breve) testo che interrompe un incolonnamento di equazioni senza perdere i

parametri dell'incolonnamento, così che le successive equazioni mantengono lo stesso incolonnamento; per esempio:

$$a_4x^4 + a_2x^2 + a_0 = 0 \tag{14.25}$$

diventa una semplice equazione di secondo grado:

$$a_4t^2 + a_2t + a_0 = 0 \tag{14.26}$$

se si pone:

$$t = x^2 \tag{14.27}$$

Simile a `\intertext` è il comando `\text`, già visto; esso serve per inserire un po' di testo dentro una formula, specialmente negli enunciati delle condizioni, ma non solo; esso serve anche e tipicamente all'interno dell'ambiente *cases*; l'equazione 13.5 usata per mostrare la composizione dei delimitatori estensibili e del delimitatore invisibile, può essere scritta mediante questo ambiente e questo comando nella maniera seguente:

$$x_{1,2} = \begin{cases} \dfrac{-b \pm \sqrt{b^2 - 4ac}}{2a} & \text{se } b^2 - 4ac > 0 \\ -\dfrac{b}{2a} & \text{se } b^2 - 4ac = 0 \\ \dfrac{-b \pm \mathrm{i}\sqrt{4ac - b^2}}{2a} & \text{se } b^2 - 4ac < 0 \end{cases} \tag{14.28}$$

scrivendo nel file sorgente:

```
  \begin{equation}
  x_{1,2}=
  \begin{cases}
    \dfrac{-b\pm\sqrt{b^2-4ac}}{2a}
                              &\text{se \(b^2-4ac>0\)}\\[1.5ex]
    -\dfrac{b}{2a}            &\text{se \(b^2-4ac=0\)}\\[1.5ex]
    \dfrac{-b\pm\uimm\sqrt{4ac-b^2}}{2a}
                              &\text{se \(b^2-4ac<0\)}
  \end{cases}
\end{equation}
```

Si noti che nell'ambito dell'argomento di `\text` la condizione è nuovamente composta come matematica in linea, quindi è necessario usare gli appositi comandi `\(` e `\)` come si fa normalmente quando si compone all'interno dei capoversi. Si noti anche l'uso del comando `\dfrac` per obbligare LaTeX a comporre le frazioni in modo display, nonostante che spontaneamente esso comporrebbe quelle frazioni in modo testo; l'ulteriore spaziatura mediante l'argomento facoltativo `\\[1.5ex]` serve per compensare l'effetto prodotto dall'uso delle frazioni in display.

Esercizio 14.2 Che cosa succederebbe se nell'equazione 14.28 non si usasse `\dfrac`?

Esercizio 14.3 Che cosa succederebbe nell'equazione 14.28 se si usasse `\\` senza il suo argomento facoltativo?

14.5.4 Le frecce estensibili

Le frecce estensibili diventano lunghe quanto i loro argomenti; si tratta dei comandi `\xleftarrow` e `\xrightarrow` che accettano due argomenti, uno obbligatorio che sarà collocato sopra la freccia e uno facoltativo che verrà collocato sotto la freccia:

$$A \xleftarrow{n+\mu-1} B \xrightarrow[T]{n\pm i-1} C \tag{14.29}$$

composto con

```
A \xleftarrow{n+\mu-1} B \xrightarrow[T]{n\pm i-1} C
```

14.5.5 Gli indici incolonnati

Talvolta può essere necessario indicare gli indici di somma o di prodotto con diverse indicazioni o condizioni, come in

$$\sum_{\substack{0\le i\le m\\ 0<j<n}} P(i,j) \tag{14.30}$$

Questo risultato si ottiene mediante `\substack` come si vede dall'esempio 14.30, composto con

```
\begin{equation}
\sum_{\substack{0\le i\le m\\
                0<j<n
               }
     } P(i, j)
\end{equation}
```

14.5.6 Gli integrali multipli

Nel caso di integrali multipli LaTeX senza estensioni necessiterebbe di un aggiustamento della spaziatura; si osservi:

$$\int\int\int\int \dots \,\mathrm{d}w\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z \qquad \iiiint \dots \,\mathrm{d}w\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z \tag{14.31}$$

Nel primo caso si è usato quattro volte il segno di integrale, mentre nel secondo caso si è usato il comando per l'integrale quadruplo disponibile con il pacchetto *amsmath* che mette a disposizione anche i comandi per l'integrale doppio e per quello triplo; in totale i comandi disponibili sono `\int` per l'integrale semplice, `\iint` per l'integrale doppio, `\iiint` per l'integrale triplo e `\iiiint` per l'integrale quadruplo; infine c'è anche `\idotsint` per l'integrale multiplo.

14.5.7 L'operatore differenziale

Si noti nell'equazione 14.31 il segno di differenziale usato come un operatore speciale che lascia il suo normale spazio alla sua sinistra ma non lascia spazio alla sua destra; inoltre esso è scritto in tondo come prescrivono le norme ISO valide per le espressioni matematiche per la fisica e le altre scienze sperimentali. La matematica non considera il segno di differenziale come un operatore. Qui lo chiamiamo operatore speciale per giustificare perché usiamo il comando `\mathop` per definirlo. È chiaro che non si tratta di un normale operatore definibile con il comando già descritto `\DeclareMathOperator`, perché con questa definizione il differenziale avrebbe lo spazio anche alla sua destra. Nel preambolo è stato invece definito la macro per il differenziale `\diff` servendosi invece di un operatore 'vuoto', così:

```
\newcommand*{\diff}{\mathop{}\!\mathrm{d}}
```

I dettagli di `\newcommand` verranno visti nel capitolo 19; qui limitiamo la nostra attenzione sull'operatore `\mathop` il cui argomento viene definito come operatore matematico, salvo che in questo caso il suo argomento è vuoto; nonostante ciò gli spazi che attorniano gli operatori sono rispettati, anche a destra; ma questo spazio a destra non ci vuole, quindi con il comando `\!` si arretra della stessa quantità, dopo di che si inserisce il segno del differenziale in tondo. Ecco, questa è una delle rare volte in cui si è usato un comando di spaziatura matematica (arretramento di uno spazio fine), ma lo si è fatto a ragion veduta per definire un comando per un simbolo speciale come il differenziale.

Come già detto, i matematici affermano che il simbolo del differenziale non rappresenta un operatore; qui non si discute la sua intima essenza, ma si sottolinea il fatto che le norme ISO prescrivono la scrittura in tondo solo per le espressioni matematiche della fisica e delle altre scienze sperimentali, dove le variabili devono essere scritte in corsivo e dove la maggior parte delle variabili ha significati standardizzati. Nel caso specifico d può voler dire: diametro, densità, distanza dei piani reticolari, coefficiente piezoelettrico, eccetera, come si deduce dalla tabella 24.9 del paragrafo 24.2. La possibile confusione del segno corsivo del differenziale con uno di quei simboli standardizzati giustifica la norma ISO, almeno agli occhi dei fisici e dei tecnologi.

Vale la pena di sottolineare che gli spazi del differenziale sono necessari per scrivere gli integrali, mentre servono di meno per scrivere le equazioni differenziali, ma ci pensa LaTeX a tenerli nella debita considerazione, come nella seguente equazione:

$$a\frac{\mathrm{d}^2x}{\mathrm{d}t^2} + b\frac{\mathrm{d}x}{\mathrm{d}t} + c = f(t)$$

scritta usando il codice:

```
\[
a\frac{\diff^2x}{\diff t^2} + b\frac{\diff x}{\diff t} + c = f(t)
\]
```

14.5.8 I simboli corsivi matematici in nero

Una cosa che capita spesso in matematica è quella di usare il carattere nero all'interno di una formula composta con carattere medio. LaTeX senza estensioni non consente di scegliere altro che una formula tutta in carattere medio oppure tutta in carattere nero, non consente di mescolare questi due casi, a meno di non fare acrobazie di tipo prescrittivo, piuttosto che di mark-up. Inoltre se si desidera comporre una formula tutta nera bisogna dichiararlo prima di entrare in modo matematico, e di conseguenza occorre proteggere la formula 'nera' dentro un gruppo, oppure è necessario cancellare l'effetto di annerimento mediante l'apposito comando:

```
\boldmath
⟨ambiente matematico nero⟩
\unboldmath
```

All'interno di una espressione è possibile usare il carattere nero, ma solo per il tondo, non per il corsivo matematico. Il comando sarebbe `\mathbf` del tutto analogo al corrispondente comando per il testo `\textbf`; ma per il corsivo matematico non c'è nulla.

L'estensione fornita da *amsmath* consente di usare il comando `\boldsymbol` il cui argomento può essere qualunque simbolo letterale o operazionale matematico:

```
\boldsymbol{⟨simbolo matematico⟩}
```

a cui si aggiunge una scorciatoia (anche qualitativa) del *poor man bold*, `\pmb`, che simula il font nero replicando un segno chiaro tre volte con un leggero spostamento a destra e in alto (e questo rende meno definito il segno; ecco perché qualitativamente si tratta di una soluzione da pover'uomo). La sintassi è la stessa:

```
\pmb{⟨simbolo matematico⟩}
```

In generale si sconsiglia di usare `\pmb`; con i font di serie con LaTeX non dovrebbe essere necessario usarlo altro che per i grandi operatori e per i delimitatori estensibili, ma non sembra una operazione molto frequente da fare.

Con `\boldsymbol` è possibile, per esempio, usare il corsivo matematico per indicare i vettori:

$$\boldsymbol{a} = a_x\boldsymbol{i} + a_y\boldsymbol{j} + a_z\boldsymbol{k}$$

mentre con `\pmb` si ottiene:

$$\pmb{a} = a_x\pmb{i} + a_y\pmb{j} + a_z\pmb{k}$$

e la differenza si vede ad occhio nudo.

È disponibile sempre anche il pacchetto *bm* che produce lo stesso effetto ma in modo più efficiente; per compatibilità con *amsmath* ridefinisce `\boldsymbol` in modo che sia completamente equivalente al suo comando specifico `\bm`. Questo pacchetto, però, è indipendente da *amsmath* e può essere usato con o senza il secondo.

14.5.9 Le espressioni matematiche riquadrate

Secondo lo scrivente per mettere in risalto una espressione matematica si può usare il meccanismo fornito da LaTeX, cioè mettendola in display; talvolta, però, potrebbe essere necessario ricorrere ad un'enfasi maggiore e per questo scopo basta racchiudere l'espressione dentro ad una cornice; *amsmath* mette a disposizione il comando `\boxed` come nell'esempio seguente:

$$\boxed{x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}}$$

composto con

```
\[
\boxed{x_{1,2}= \frac{-b\pm\sqrt{b^2-4ac}}{2a}}
\]
```

14.6 Le matrici e i determinanti

Le matrici e i determinanti sono particolari tabelle che contengono solo, o prevalentemente, espressioni matematiche; *amsmath* mette a disposizione un certo numero di ambienti per comporre matrici variamente delimitate e perciò con significati matematici diversi.

L'ambiente base è ovviamente *matrix* e produce una matrice avente al massimo 10 colonne, ma sprovvista di delimitatori; anche gli altri ambienti consentono un massimo di 10 colonne; questo numero può venire cambiato se si imposta il contatore `MaxMatrixCols` con un valore diverso:

`\setcounter{MaxMatrixCols}{`⟨*numero delle colonne*⟩`}`

Gli altri ambienti per costruire matrici o determinanti sono i seguenti: *pmatrix*, *bmatrix*, *Bmatrix*, *vmatrix* e *Vmatrix*; tutti questi racchiudono la matrice vera e propria all'interno di una coppia di delimitatori estensibili; a seconda del delimitatore la matrice ha un significato diverso. Gli ambienti *pmatrix*, *bmatrix* e *Bmatrix* racchiudono la matrice all'interno di una coppia di parentesi tonde, oppure quadre, oppure graffe; l'ambiente *vmatrix* serve per comporre un determinante racchiudendo la matrice fra due aste verticali, mentre l'ambiente *Vmatrix* serve per comporre una norma racchiudendo la matrice fra due coppie di barre verticali.

Grazie a questi ambienti non è difficile comporre una espressione del tipo:

$$\begin{vmatrix} 1 & 1 & 0 & 0 \\ -1 & 2 & 1 & 0 \\ 0 & -1 & 3 & 1 \\ 0 & 0 & -1 & 4 \end{vmatrix} = 30 \tag{14.32}$$

mediante il codice:

```
\begin{equation}
    \begin{vmatrix}
      1 & 1 & 0  & 0 \\
     -1 & 2 & 1  & 0 \\
      0 &-1 & 3  & 1 \\
      0 & 0 &-1  & 4
    \end{vmatrix}
    = 30
\label{equ:determinante}
\end{equation}
```

La scrittura nel file sorgente del codice per la composizione della matematica è più facilmente leggibile se si usa una scrittura strutturata; ci si rende più facilmente conto se ogni segno di apertura è in corrispondenza con il suo compagno di chiusura; usando questa pratica, nei limiti del possibile si evitano moltissimi errori e si compone più speditamente.

Una equazione matriciale diventa abbastanza semplice da comporre, specialmente se si ha l'accortezza di scrivere in modo strutturato nel file sorgente tutto quello che va ordinatamente inserito nelle matrici:

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}$$

Il codice sorgente è:

```
\[
  \begin{pmatrix}
    a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\
    a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\
    a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\
    a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4}
  \end{pmatrix}
  \begin{pmatrix}
    x_1\\x_2\\x_3\\x_4
  \end{pmatrix}
  =
  \begin{pmatrix}
    y_1\\y_2\\y_3\\y_4
  \end{pmatrix}
\]
```

Tuttavia esistono certe necessità della matematica che non possono essere soddisfatte con i mezzi messi a disposizione dal pacchetto *amsmath*, come per esempio quella di ripartire una matrice in sottomatrici mediante linee tratteggiate,

come nell'equazione 14.33 ottenuta usando il pacchetto *tabu* e il suo ambiente *tabu* in ambiente matematico, quindi in sostituzione dell'ambiente *array*.

$$\left(\begin{array}{ccc:ccc} a_{1,1} & \cdots & a_{1,n} & b_{1,n+1} & \cdots & b_{1,n+m+1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & \cdots & a_{n,n} & b_{n,n+1} & \cdots & b_{n,n+m+1} \\ \hdashline c_{n+1,1} & \cdots & c_{n+1,n} & d_{n+1,n+1} & \cdots & d_{n+1,n+m+1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ c_{n+m+1,1} & \cdots & c_{n+m+1,n} & d_{n+m+1,n+1} & \cdots & d_{n+m+1,n+m+1} \end{array}\right) \tag{14.33}$$

Prima di comporre l'equazione 14.33, possibilmente dentro un gruppo che contenga anche l'equazione, si definisce lo stile delle linee tratteggiate orizzontali e verticali con il comando `\tabulinestyle`:

```
\tabulinestyle{on 2pt off 3pt}
```

dove la stringa `on` indica la lunghezza dei tratti "pieni" e la stringa `off` quella dei tratti "vuoti"; in realtà la sintassi di `\tabulinestyle` è più complessa e consente di specificare anche lo spessore della linea tratteggiata e il colore dei tratti pieni e vuoti; si veda la documentazione del pacchetto *tabu*. Il comando `\tabucline` può ricevere l'indicazione delle colonne che deve sottolineare (un po' come il comando `\cline` del nucleo di LaTeX), ma con la scrittura `\tabucline-` sottolinea tutte le colonne presenti nella tabella o nella matrice. Le linee verticali sono invece indicati dai soliti indicatori nell'argomento del comando di apertura dell'ambiente *tabu*. Ciò premesso, diventa chiaro il codice con il quale si è composta l'equazione 14.33:

```
{\tabulinestyle{on2pt off3pt}
\tabulinesep=1.5mm
\begin{equation}
\left(
\begin{tabu}{@{}*3c|*3c@{}}
a_{1,1}    & \cdots  & a_{1,n}   & b_{1,n+1}     & \cdots & b_{1,n+m+1}     \\
\vdots     & \ddots  & \vdots    &  \vdots       & \ddots & \vdots          \\
a_{n,1}    & \cdots  & a_{n,n}   & b_{n,n+1}     & \cdots & b_{n,n+m+1}     \\
\tabucline-
c_{n+1,1}  &  \cdots & c_{n+1,n} & d_{n+1,n+1}   & \cdots & d_{n+1,n+m+1}   \\
\vdots     &  \ddots & \vdots    &  \vdots       & \ddots & \vdots          \\
c_{n+m+1,1}&  \cdots &c_{n+m+1,n}& d_{n+m+1,n+1} & \cdots & d_{n+m+1,n+m+1} \\
\end{tabu}
\right)
\end{equation}}
```

Si sottolinea il fatto che che i "filetti" di separazione fra righe e/o colonne della tabella o della matrice sono solitamente poco professionali in tipografia; l'uso qui fatto per separare la matrice in sottomatrici invece è perfettamente giustificato.

14.7 I diagrammi commutativi

I diagrammi commutativi sono particolari espressioni matematiche a metà strada fra un insieme di espressioni matematiche e un disegno che mette in relazione le une con le altre.

Il pacchetto *amscd* serve per estendere *amsmath* definendo un ambiente *CD* con il quale è possibile disegnare diagrammi commutativi piani e senza frecce diagonali. Per diagrammi più complessi il lettore deve rivolgersi a pacchetti di estensione più specializzati come *xypic* oppure *kuvio*, senza trascurare l'onnipotente pacchetto *tikz*.

Il diagramma commutativo:

$$\begin{array}{ccc} S^{\mathcal{W}_\Lambda}\otimes T & \xrightarrow{\ j\ } & T \\ \big\downarrow & & \big\downarrow{\scriptstyle \text{End } P} \\ (S\otimes T)/I & = & (Z\otimes T)/J \end{array}$$

è stato composto con il codice:

```
\[
  \begin{CD}
    S^{\mathcal{W}_\Lambda}\otimes T @>j>> T \\
    @VVV                             @VV{\mathrm{End}\:P}V\\
    (S\otimes T)/I                   @=(Z\otimes T)/J
  \end{CD}
\]
```

Oltre ai vari comandi per inserire le frecce o le linee di collegamento fra le varie espressioni, si noti l'uso di `\mathcal` per scrivere caratteri 'calligrafici' e l'uso del comando di spaziatura `\:` per inserire uno spazio circa pari a quello interparola; il comando `\:` in matematica si comporta diversamente dai comandi analoghi per il modo testo `\enspace` o `\thinspace`, perché esso cambia valore a seconda che l'espressione matematica che si sta componendo sia in modo testo, in modo display, in modo degli indici del primo ordine o di quelli del secondo ordine.

Si prenda nota anche del fatto che l'ambiente *CD* deve essere realizzato all'interno di un ambiente matematico come quello prodotto da `\[...\]` oppure uno dei vari altri ambienti visti in questo capitolo.

A proposito di *xypic* conviene sottolineare che esso usa una sintassi molto particolare dove si fa uso di caratteri speciali per eseguire certe formattazioni; uno di questi caratteri è il segno `"` che con l'opzione *italian* di *babel*[3] è attivo e svolge altre funzioni diverse da quelle che esso è chiamato a svolgere quando si

[3]Dalla versione 1.3k, il modulo per l'italiano di *babel* definisce le doppie virgolette attive se e solo se dopo aver caricato *babel* e prima di `\begin{document}` si dà il comando `\setactivedoublequote`. Una volta attivate, le doppie virgolette diritte possono essere disattivate e riattivate solo mediante i comando`\shorthandoff` e `\shorthandon`.

usa *xypic*. Per questo conviene ridefinire il comando di apertura dell'ambiente *xy* nel modo seguente:

```
\toks0=\expandafter{\xy}
\edef\xy{\noexpand\shorthandoff{\noexpand"}\the\toks0 }
```

Si tratta di programmazione con il linguaggio primitivo di TEX, salvo per il comando `\shorthandoff` che, insieme al suo compagno `\shorthandon`, serve per disattivare o, rispettivamente, per riattivare uno o più caratteri attivi; è un comando definito da *babel*. In questo modo, se si usa il pacchetto *xypic*, basta dare quelle definizioni subito dopo la chiamata del pacchetto e poi si è sicuri di poter usare tranquillamente l'ambiente *xy* senza i problemi che il carattere attivo potrebbe dare. Si noti che questo trucco *deve* venire usato ogni volta che si compone in una lingua diversa dall'inglese, perché in tutte le altre lingue europee il segno " viene sempre definito attivo.

14.8 La punteggiatura in matematica

La punteggiatura in matematica è tipograficamente un problema aperto perché non esistono norme che regolino la questione; è più una questione di consuetudini tipografiche ed editoriali. Anzi, ogni casa editrice si premura spesso di fornire ai suoi potenziali autori una breve guida in cui sono indicate le tradizioni compositive della casa editrice. Gli argomenti spaziano attraverso tutto lo scibile che può essere scritto, e vanno dai titoli correnti alle testatine e ai piedini, fino alle virgole del testo e la punteggiatura in matematica; naturalmente se la casa editrice riceve un contributo da pubblicare in forma direttamente stampabile, generalmente fornisce i fogli di stile nel linguaggio di videocomposizione che l'autore userà. Non sono rari i casi di case editrici che forniscono direttamente i file di classe e/o le eventuali estensioni per testi da pubblicare dopo la composizione con LATEX.

Tuttavia per quel che riguarda la punteggiatura della matematica qualche volta le informazioni o le prescrizioni sono lacunose, quindi l'autore deve supplire con la sua esperienza e la sua sensibilità.

Non dovrebbero sussistere problemi per le espressioni matematiche (brevi) che compaiono in linea con il testo, dove la punteggiatura dominante è quella testuale.

Direi che non dovrebbero esserci problemi nemmeno per la punteggiatura *interna* a ogni espressione matematica; questa punteggiatura specifica interna alle espressioni matematiche ha un suo particolare significato matematico che fa parte della sua 'grammatica'. I problemi maggiori nascono per la punteggiatura *esterna* alle espressioni matematiche in display.

Tuttavia cominciamo con i problemi minori di punteggiatura interna. In matematica si usano essenzialmente gli stessi segni di interpunzione che si usano nel testo, eccetto il punto interrogativo, salvo che alcuni di questi segni hanno un significato particolare; per esempio il punto esclamativo che segue un numero

intero o un simbolo che rappresenta un numero intero, significa che di quel numero si considera il 'fattoriale':

$$n! = n(n-1)(n-2)\cdots 2 \cdot 1$$

Lo stesso punto esclamativo può servire per cambiare il significato di un 'quantificatore' logico; per esempio $\exists!$ da taluni è usato nel senso di 'esiste ed è unico'.

I due punti hanno un significato particolare in espressioni logiche, così come (in Italia e secondo le ultime versioni delle norme ISO 80000) possono indicare l'operazione di divisione, quindi non vengono usati come segno di interpunzione.

Il punto e virgola ha il solo significato di segno di interpunzione, mentre la virgola e il punto sono 'anfoteri', possono servire ora con il significato di segno di interpunzione, ora con il significato di separatore decimale.

Scrivendo in inglese, il punto serve come separatore decimale e come punto fermo, mentre la virgola è un semplice segno di interpunzione, usato, come punteggiatura interna, per separare gli elementi di una lista.

Al contrario, in qualunque lingua diversa dall'inglese le norme ISO prescrivono la virgola nella funzione di separatore decimale, e quindi la virgola viene ad assumere due ruoli distinti; il punto, invece sembrerebbe declassato a svolgere la sola funzione di segno di interpunzione (raro nell'interpunzione interna), ma poi rientra come separatore decimale nelle costanti numeriche 'reali' di tutti i linguaggi di programmazione. Nel capitolo 19 si vedrà come fare per convincere LaTeX a trattare correttamente le due funzioni della virgola in matematica.

La punteggiatura esterna è quella che pone i principali problemi. Mi spiego meglio: per *punteggiatura esterna* intendo sia la punteggiatura che compare in una espressione che contenga al suo interno anche delle parti testuali, sia la punteggiatura finale.

Non esistono regole 'grammaticali' codificate per gestire la punteggiatura esterna. Scrivendo per una data casa editrice, come già detto, è opportuno riferirsi alle sue regole editoriali. Scrivendo per proprio conto e, specialmente, quando si è editori di se stessi, bisogna decidere autonomamente e scegliere uno stile di punteggiatura esterna per la matematica.

Sul forum del GUIT alcuni argomenti sono stati dibattuti ampiamente e uno di questi argomenti trattava appunto della punteggiatura esterna. Sono emerse due scuole di pensiero fra loro antitetiche, ma ciascuna con pregi e difetti; l'unica cosa sulla quale le due scuole di pensiero concordano è che scelto uno stile di punteggiatura, quello deve essere usato costantemente e coerentemente; il lettore non deve essere confuso da un uso irregolare e non uniforme della punteggiatura. Le due scuole di pensiero dicono sostanzialmente quanto segue.

La punteggiatura esterna è necessaria in matematica come lo è per il testo Secondo questo orientamento la punteggiatura separa le varie parti di una espressione che contiene anche dei frammenti testuali e va inserita anche alla fine delle espressioni in display, sia sotto forma di virgole, di punti e virgola

o di punti finali; in altre parole l'espressione matematica, anche se è in display, fa parte del testo e la punteggiatura ne esprime il ritmo esattamente come lo fa quando si compone del testo non matematico. Bisogna solo stare attenti a che i segni di punteggiatura non interferiscano con la parte matematica in senso stretto al punto da renderne difficile la lettura o, peggio ancora, al punto di indurre il lettore a farne una lettura ambigua o errata.

Siccome la punteggiatura non va staccata da quanto viene prima (a differenza del francese, nel quale almeno il punto e virgola va spaziato con uno spazio di almeno un terzo di em da quanto viene prima), talvolta in italiano si potrebbero spaziare virgole e punti, suscettibili di introdurre ambiguità nella lettura delle espressioni, mediante uno spazio fine, che in matematica si rende semplicemente con il comando `\,`; si confronti, per esempio,

$$w = \frac{az+b}{cz+d},\ \text{con } a,b,c,d \in \mathbb{R};\ z,w \in \mathbb{C}$$

con

$$w = \frac{az+b}{cz+d}\,,\ \text{con } a,b,c,d \in \mathbb{R};\ z,w \in \mathbb{C}.$$

Si vede benissimo che nel secondo caso la virgola non può essere confusa con un apice della variabile d.

La matematica si scrive senza punteggiatura esterna Come si vede in questo stesso capoverso, LaTeX sostituisce la punteggiatura finale del titolino (composto con il comando `\paragraph`) con uno spazio interparola maggiore. La scuola di pensiero che non usa la punteggiatura esterna è in linea con la pratica tipografica di usare spazi adeguati al posto della punteggiatura. La punteggiatura esterna nelle espressioni matematiche viene sostituita con degli spazi o degli incolonnamenti: se si deve separare un frammento testuale all'interno di una espressione, lo si fa usando spazi di almeno un quadrato, ma meglio ancora di un quadratone, rispettivamente con i comandi `\quad` e `\qquad`; un incolonnamento eseguito per esempio con l'ambiente *align* permette di separare le colonne delle condizioni dalle colonne contenenti espressioni matematiche, come si è fatto, per esempio, con il sistema di condizioni 14.23. Proseguendo l'esempio precedente si esamini ancora

$$w = \frac{az+b}{cz+d} \qquad \text{con } a,b,c,d \in \mathbb{R} \quad z,w \in \mathbb{C}$$

Secondo questa scuola di pensiero la punteggiatura finale è ridondante, perché gli spazi prima e dopo una formula in display e il modo di scrivere la prima riga di testo che segue tale formula sono sufficienti a capire se la formula stessa chiude un capoverso, o se il periodo di cui fa parte finisce con la formula, oppure se continua dopo la formula; infatti se questa prima riga comincia con una lettera minuscola (e, ovviamente, non è rientrata) significa che il periodo continua dopo la formula; se questa prima riga non è rientrata ma comincia con una lettera maiuscola, la formula termina il periodo, e ne comincia uno nuovo subito dopo,

ma il capoverso non è terminato; se invece questa prima riga comincia con una lettera maiuscola ed è rientrata, vuol dire che dopo la formula, che chiude il periodo, comincia un nuovo capoverso. Quello che l'assenza di punteggiatura non permette di capire è il quarto caso, cioè se la formula termina una frase o se la frase continua dopo la formula; ma anche senza punteggiatura il senso del discorso è evidente anche in questo caso per la presenza di congiunzioni o altri connettivi che permettono di leggere l'espressione in display come ultimo elemento di una frase o come elemento interno ad una frase. In questi quattro casi l'altra scuola di pensiero avrebbe usato rispettivamente la virgola, il punto fermo, il punto e a capo (di fatto il punto alla fine della formula) oppure il punto e virgola.

Inoltre, con riferimento all'espressione 14.28, dove si sarebbe messa la virgola di chiusura dell'espressione? In casi analoghi si è vista la virgola (o altri segni di interpunzione) allineata con l'asse matematico[4] dell'espressione, oppure a chiusura dell'ultima riga, ma non è semplice prendere una decisione compositiva in situazioni del genere.

L'approccio seguito dalla scuola di pensiero dell'assenza di punteggiatura esterna è coerente con le regole formulate per le elencazioni semplici, dove è normale non usare la punteggiatura alla fine delle singole voci, ma è normale usare una scrittura *ellittica* nelle elencazioni semplici, mentre è da evitare assolutamente nelle enumerazioni. Quello delle elencazioni e delle enumerazioni ha a che fare solo con il testo, ma aver specificato che ogni elencazione semplice fa parte del capoverso che la precede, fa capire perché è consentita una forma di scrittura ellittica, che invece è vietata nelle enumerazioni, dove ogni voce è formata da almeno un capoverso completo. Mescolando testo e matematica non si possono ripetere gli stessi ragionamenti; tuttavia la forma ellittica per la matematica ha la sua giustificazione.

Uso simultaneo della punteggiatura e degli spazi in matematica Naturalmente nessuno vieta di usare sia la punteggiatura sia gli spazi; l'esempio riportato sopra diventerebbe

$$w = \frac{az + b}{cz + d}\,, \qquad \text{con } a, b, c, d \in \mathbb{R}; \quad z, w \in \mathbb{C}.$$

Come il lettore avrà notato, in questo testo si segue la seconda linea di pensiero; è una scelta del coordinatore, il quale si prende la responsabilità di questa scelta, ed è perfettamente conscio che i collaboratori seguaci dell'altra scuola di pensiero ne saranno un po' delusi. L'omogeneità dello stile e, quindi, la coerenza compositiva, richiedono di non cambiare la punteggiatura da un paragrafo all'altro, a seconda di chi l'ha scritto. Il lettore invece, quando compone

[4] L'asse matematico è la linea orizzontale (ideale, quindi invisibile) che passa a metà fra le due lineette che formano il segno =; gli operatori, le frazioni e tanti altri elementi delle espressioni matematiche dovrebbero avere un ingombro verticale simmetrico rispetto a questo asse.

i suoi testi, seguirà le sue convinzioni e sceglierà il tipo di punteggiatura che preferisce.

Nell'ultima versione de *L'Arte di scrivere con LaTeX*, [54], gli autori raccomandano di non usare la punteggiatura esterna.

Va infine segnalato il documento *Preparing Articles with LaTeX — Instructions to authors for preparing compuscripts* della casa editrice olandese Elsevier [22], ben nota in campo scientifico; si tratta delle istruzioni per gli autori e, per quel che riguarda la matematica, non è espressamente detto nulla sulla punteggiatura, ma gli esempi in display ne sono tutti privi.

Vale la pena di segnalare anche quanto viene scritto nel manuale *Style Manual* [2] della casa editrice Artech House. In questo testo, predisposto come aiuto agli autori dei testi scientifici che la casa editrice pubblica si afferma esplicitamente che lo stile dalla casa prescrive quanto segue.

1. Non si usa la punteggiatura esterna. In una versione più recente di quel manuale, scaricabile dal sito della Artech House, si afferma addirittura che le equazioni in display non vogliono nessuna punteggiatura.

2. Non si comincia mai un capoverso con una equazione né in linea con il testo né in display.

3. Il periodo o la frase che precede una equazione termina con i due punti se l'ultimo elemento della frase è un nome, un aggettivo, un avverbio, un numero di riferimento anche se fra parentesi o in un inciso; non si usa nessuna punteggiatura se la frase termina con un verbo, un articolo, una congiunzione o una preposizione[5].

Forse queste regole potrebbero essere usate anche per qualsiasi oggetto in display inserito nel corso del discorso, non per gli oggetti flottanti.

14.9 Conclusioni

Non è questo il luogo per ripetere quanto è reperibile sulla documentazione del pacchetto `amsmath`; né è questo il luogo per illustrare altri pacchetti e altre funzionalità che la fantasia degli utenti di LaTeX è riuscita a inventare.

Il lettore sappia però che la storia non è finita qui; egli è invitato a esplorare gli archivi internazionali CTAN e scoprirà una varietà infinita di estensioni per la composizione della matematica che non potrà fare a meno di stupirlo; sappia però discernere quanto è stato escogitato alla buona per risolvere un problema contingente, da quanto è stato programmato professionalmente per l'uso generale.

In particolare legga con attenzione il documento `Mathmode.pdf` facente parte della documentazione standard di TeX Live a partire dal 2010, [71]. In questo documento Herbert Voß ha raccolto una infinità di casi di composizione matematica

[5] Secondo lo scrivente sarebbe meglio usare i due punti anche dopo i verbi. Ma qui si sta descrivendo lo stile compositivo della Artech House, non lo stile che bisognerebbe usare in generale.

che richiedono comandi speciali e ne indica moltissimi; è una vera miniera da cui attingere idee e soluzioni a problemi compositivi della matematica che non può venire ignorata. Essendo un documento inizialmente scritto da un tedesco per i suoi connazionali, poi tradotto in inglese, si è anche sicuri di trovare informazioni che siano conformi alle tradizioni tipografiche europee, quasi sempre conformi con le norme ISO.

Capitolo 15

LaTeX: composizione di testi letterari e filologici

La composizione di testi letterari o filologici è forse un argomento troppo specializzato per una guida introduttiva come questa.

Tuttavia dopo aver dedicato due capitoli alla matematica, bisogna trattare la filologia che è una scienza molto esatta e che si avvale di una notazione bidimensionale come la matematica. Nello stesso tempo sembra necessario trattare delle classi che consentono di comporre testi letterari con stili assai diversi da quelli dei testi tecnico-scientifici. Sì, è vero che LaTeX si è sviluppato inizialmente nel mondo delle materie tecnico-scientifiche, ma la sua forza compositiva è stata riconosciuta anche da coloro che si occupano di scienze umane e non è un caso che fra i primi umanisti ci siano stati proprio i filologi.

Talvolta è difficile distinguere un testo letterario da un testo filologico perché il testo ha entrambe le caratteristiche; quindi la suddivisione della descrizione che ora si farà con sezioni distinte è puramente semplificativa e una sezione non esclude l'altra.

15.1 La composizione di testi letterari

In un certo senso la composizione di testi letterari, come romanzi, saggi e simili è più semplice di quelli tecnico-scientifici; in realtà segue delle norme e convenzioni di stile compositivo un po' differenti grazie al fatto che la mancanza di numerosi inserti in display permette di essere più rigidi per quel che riguarda la scelta dei font, la giustezza e l'intera geometria della pagina; questa rigidezza si riflette specialmente sullo scartamento che deve assolutamente essere costante e replicarsi esattamente sul recto e sul verso dello stesso foglio.

Se non occorrono altre strutture compositive di sezionamento gerarchico la geometria della pagina è piuttosto semplice e permette di usare le proporzioni della gabbia descritte nei capitoli 6 e 20. Tuttavia merita citare qui alcune

Formato	base per altezza	*Formato*	base per altezza
foolscalp	108 mm × 171 mm	*crown*	127 mm × 191 mm
post	122 mm × 194 mm	*largepost*	137 mm × 210 mm
demi	143 mm × 222 mm	*medium*	146 mm × 229 mm
royal	159 mm × 254 mm	*superroyal*	171 mm × 267 mm
imperial	191 mm × 279 mm		

Tabella 15.1: Formati britannici della pagina

semplici classi preconfezionate che consentono di comporre testi letterari molto semplici.

Fra i pacchetti che offrono queste possibilità non può essere trascurata la classe `octavo`. Questa semplice classe si basa sulla classe `book` ma ha dei formati di carta dai nomi molto britannici (vedi la tabella 15.1) ed è prevista per la composizione in ottavo, proprio come dice il nome. I margini sono prefissati e la gabbia è proporzionata al corpo normale. Sono disponibili tutte le strutture compositive della classe `book`, ma, secondo lo scrivente, questa classe è particolarmente indicata per i testi letterari. Si richiama, anzi, l'attenzione sul formato *largepost* la cui altezza corrisponde al formato A5, mentre la larghezza è un poco inferiore; facendo un po' di attenzione e sfruttando adeguatamente le indicazioni della documentazione, è possibile stampare nell'ordine giusto, fronte retro, quattro fogli A4 per poter disporre di un fascicoletto di 8 pagine e 16 facciate che può costituire una piccola brochure già da sola, oppure può costituire una segnatura in ottavo di un documento più vasto. Stampate tutte la segnature anche su una normale stampante per formato A4, le segnature possono essere cucite in modo da formare un libretto completo che risulta di un formato un po' più stretto del formato A5, quando si siano rifilati i margini esterni con una taglierina così da rispettare le dimensioni secondo il disegno originario del formato *largepost*.

Per avere disponibile anche il formato A5 non è difficile modificare la classe (dopo averle cambiato il nome, per esempio in `ottavo`) aggiungendo le tre righe relative a questo formato:

```
\@mainmattertrue
%%%% Opzione a5paper prima dei formati inglesi
\DeclareOption{a5paper}
    {\setlength\paperheight{210mm}%
    \setlength\paperwidth{148.5mm}}
%%%% Seguono le altre opzioni per i formati inglesi
```

Anche la classe `layaureo`, che è stata descritta meglio nel capitolo 6 funziona davvero bene per la stampa di lavori letterari in formato A4 ma con la gabbia e i margini disposti rispettando in parte la proporzione aurea. Naturalmente questa classe serve anche per i documenti tecnico-scientifici, ma, sempre secondo lo scrivente, è molto indicata per i testi letterari.

Nel capitolo 6 viene descritta anche la classe *memoir*; sempre secondo lo scrivente, questa classe è la più versatile, specialmente dopo le aggiunte presenti nella versione del 2008; usando bene le opzioni disponibili e i comandi di personalizzazione è abbastanza agevole disporre di una gabbia adatta al testo letterario, con una scelta assolutamente libera per i font dei titolini correnti in modo da rispettare facilmente le indicazioni fornite nel testo [57], indicate principalmente per i testi letterari.

Nel capitolo 6 è stata descritta anche la classe *suftesi*, esplicitamente creata da un umanista per testi umanistici, anche se non è fuori luogo il suo uso per testi tecnico-scientifici. Il suo pregio è di essere fortemente personalizzabile nei formati e nei layout di pagina; è usabile per comporre riviste complete contenenti molti articoli scritti da autori diversi; il suo creatore la sta già usando per comporre una rivista..

Se però si vogliono scrivere testi letterari di tipo antologico, con le righe numerate sia per le parti in prosa sia per quelle in poesia, allora conviene riferirsi al pacchetto *technica*, il cui nome, almeno nella documentazione, è scritto in greco: τεχνικὰ.

Si tratta di un lavoro molto completo e di largo respiro; consente di comporre il testo letterario puro e semplice, quello relativo ai libretti teatrali, il testo poetico; gli esempi allegati alla documentazione riportano testi in latino, italiano, greco classico, inglese, francese, spagnolo e tedesco, così da mostrare anche le possibilità compositive in lingue diverse dall'italiano. Per ogni esempio è indicato lo stile della casa editrice e/o della collana che è stato riprodotto con una certa (in realtà ottima) approssimazione. La cosa curiosa è che i comandi della classe sono tutti in latino, e l'autore, Gianfranco Boggio-Togna, afferma che l'utente può anche usare i casi per i nomi latini in modo da non urtare la sensibilità dei latinisti, che così possono scrivere dei costrutti di comandi in latino (quasi) perfetto. Il "quasi" è d'obbligo, perché buona parte dei comandi si riferisce a comandi di formattazione che certamente non hanno equivalenti latini, nemmeno cercando sul vocabolario ufficiale edito dalla Tipografia Vaticana[1].

Quello che può trovare un poco stupiti è che gli esempi greci non fanno uso dei font di default di *babel*, e nemmeno del nome *greek* per l'opzione per la lingua greca; l'autore non dice che il suo pacchetto non sia adatto ad essere usato con i font di default e con la normale opzione *greek*, ma lui usa sempre esempi con la lingua *ibycus* che a sua volta fa riferimento ad un altro font greco tramite una codifica chiamata *Beta Code*. L'aspetto dei singoli caratteri è molto simile, anche se non proprio identico ai caratteri mostrati nella tabella 18.12, tuttavia la scrittura con una tastiera latina prevede una corrispondenza dei tasti leggermente diversa e, quel che più conta, i segni diacritici sono tutti indicati invariabilmente *dopo* la lettera a cui si riferiscono (tranne per le maiuscole, dove vanno indicati prima) e vengono ottenuti con tasti diversi da quelli che vengono usati con i font

[1] La lingua latina è quella ufficiale della Santa Sede, da non confondere con lo Stato della Città del Vaticano, la cui lingua ufficiale è l'italiano; vi è anche una Commissione apposita per adattare il latino moderno con neologismi o locuzioni inedite necessarie per riferirsi a oggetti e/o a situazioni della vita contemporanea.

ΕΠΤΑ ΕΠΙ ΘΗΒΑΣ

ΕΤΕΟΚΛΗΣ

Κάδμου πολῖται, χρὴ λέγειν τὰ καίρια
ὅστις φυλάσσει πρᾶγος ἐν πρύμνῃ πόλεως
οἴακα νωμῶν, βλέφαρα μὴ κοιμῶν ὕπνῳ.
εἰ μὲν γὰρ εὖ πράξαιμεν, αἰτία θεοῦ·
εἰ δ' αὖθ', ὃ μὴ γένοιτο, συμφορὰ τύχοι,
Ἐτεοκλέης ἂν εἷς πολὺς κατὰ πτόλιν
ὑμνοῖθ' ὑπ' ἀστῶν φροιμίοις πολυρρόθοις
οἰμώγμασὶν θ', ὧν Ζεὺς ἀλεξητήριος
ἐπώνυμος γένοιτο Καδμείων πόλει.
ὑμᾶς δὲ χρὴ νῦν, καὶ τὸν ἐλλείποντ' ἔτι
ἥβης ἀκμαίας καὶ τὸν ἔξηβον χρόνῳ,
ὥραν τ' ἔχονθ' ἕκαστον ὥστε συμπρεπές,
βλαστημὸν ἀλδαίνοντα σώματος πολύν,
πόλει τ' ἀρήγειν καὶ θεῶν ἐγχωρίων
βωμοῖσι, τιμὰς μὴ 'ξαλειφθῆναί ποτε.

Figura 15.1: Breve estratto dell'opera di Eschilo *Sette contro Tebe*

di default. Questo dipende dal fatto che gli ellenisti molto spesso fanno riferimento al Theasurus Linguae Graecae in rete, e questo Thesaurus ha i suoi testi scritti in Beta Code; per il letterato è quindi molto agevole servirsi del metodo "copia e incolla" per riportare brani già presenti sul Thesaurus e per i quali sia concesso di operare in questo modo. Il letterato risparmia non poco tempo e fatica, ma principalmente risparmia un gran numero di errori di battitura.

Inoltre è curioso che il Beta Code preveda nel file sorgente solo simboli ASCII (e fin qui va bene) ma tutte le lettere devono essere maiuscole nel file sorgente; gli applicativi che consentono di leggere i file sorgente già resi in lettere greche, rendono tutte le lettere in minuscolo, a meno che la maiuscola non sia marcata con un asterisco. La codifica Beta Code di ibycus è molto più semplice, nel senso che le minuscole vengono scritte con lettere minuscole e le maiuscole vengono scritte direttamente con lettere maiuscole; un file sorgente scritto in lettere latine sfruttando il Beta Code non appare a prima vista molto diverso dallo stesso testo scritto in lettere latine con la codifica di default del font usato da `babel`; tuttavia le due codifiche sono profondamente diverse e incompatibili l'una con l'altra.

Perché allora il font greco di default non è stato fatto secondo la codifica Beta Code? Ci sono almeno tre spiegazioni: (*a*) i font CBgreek sono stati fatti estendendo un font (già presente negli archivi CTAN) che aveva già una sua codifica e questa codifica è stata mantenuta; (*b*) la codifica del font preesistente

era stata definita da Sylvio Levy, uno scienziato di origine greca – nato in Brasile e ora operante negli USA – che sapeva usare bene sia la tastiera greca sia quella latina; se un greco aveva scelto quella codifica, aveva le sue buone ragioni, ed era certo che Beccari non poteva arrogarsi il diritto di mettere a disposizione di un pubblico, comprendente anche tutti i greci, un font incompatibile con le abitudini di diteggiatura sulla tastiera greca, dove i diacritici hanno sempre la posizione *prefissa* invece che *postfissa* come nel Beta Code; (*c*) in rete è disponibile il pacchetto `bgreek` che permette di usare i font di default con la notazione del file sorgente che segue le convenzioni Beta Code.

Ciò premesso il pacchetto `technica` va usato con la classe `book`, anche se ne modifica tutte le impostazioni e aggiunge un numero enorme di estensioni; non permette ancora di comporre una edizione critica, ma ci va molto vicino.

Nella figura 15.1 è rappresentato un breve brano in versi della tragedia di Eschilo *Sette contro Tebe*; il parlante, Eteocle, è riportato sul margine sinistro in maiuscolo inclinato, mentre i versi sono numerati con una numerazione che ad un certo punto presenta i versi 12 e 13 scambiati rispetto al codice originale. Si noti che l'esempio della figura 15.1 è stato ricomposto con l'opzione *greek* di `babel`, specificando il modificatore *polutoniko*, tanto per verificare se i font di default, con la loro notazione dei diacritici "prefissa", invece che "postfissa", dessero qualche fastidio al pacchetto `technica`, ma non si sono osservati difetti di sorta.

15.2 Scrivere in greco

Vale la pena, a questo punto, aggiungere poche righe per ricordare come si fa a comporre in greco usando la tastiera latina. Innanzi tutto il file sorgente deve invocare `babel` indicando (solo o) anche la lingua greca; se si vuole scrivere in greco moderno con l'ortografia monotonica, non occorre altro; se però si vuole scrivere in greco classico, o in greco moderno con l'ortografia politonica, bisogna aggiungere una dichiarazione di attributo per la lingua greca, così:

Solo pdflatex

```
\usepackage[...,greek.polutoniko,...,italian]{babel}
```

oppure:

```
\usepackage[...,greek.ancient,...,italian]{babel}
```

Volendo si potrebbe caricare anche il pacchetto `teubner`,

```
\usepackage[...,greek,...,italian]{babel}
\usepackage{teubner}
```

ma di solito, se non occorrono speciali strutture filologiche, l'uso del pacchetto `teubner` è eccessivo.

Le corrispondenze fra i caratteri della tastiera 'latina' con le lettere greche sono riportate nella tabella 15.2. Gli accenti e gli spiriti vanno preposti in una

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	w	x	y	z
α	β	ς	δ	ε	φ	γ	η	ι	ϑ	κ	λ	μ	ν	ο	π	χ	ρ	σ	τ	υ	ω	ξ	ψ	ζ
			>	<	’	‘	~	”	”␣	|			:	;	?			((	))					
			̓	̔	́	̀	̃	̈	̓	̣			:	·	;			«	»						

Tabella 15.2: Corrispondenze fra i caratteri della tastiera latina e i segni dell’alfabeto greco

sequenza qualunque alle vocali (o al ρ) *anche* senza l’uso di backslash; lo iota sottoscritto va inserito invece posponendo il segno | alla vocale che lo deve accogliere. Il segno " seguito da uno spazio si traduce in un apostrofo, mentre se precede un iota o un upsilon vi colloca la dieresi; in questa funzione la dieresi può essere anche preceduta o seguita dal segno di accento grave, o acuto o circonflesso. L’unica difficoltà che si incontra a scrivere con la tastiera italiana è che da questa manca una combinazione di tasti per generare l’accento grave e l’accento tilde che funziona da circonflesso per il greco; la tastiera italiana del Mac consente di ottenere questi segni con l’aiuto del tasto ‘comando’, ma con Windows è necessario ricorrere al tasto Alt e alla digitazione dei numeri 96 e 126 rispettivamente sul tastierino numerico; su un laptop Windows è una vera contorsione, ma se lo shell editor lo consente, questi segni possono essere ottenuti con un semplice click del mouse in una apposita tabellina proposta dall’editor. Da Win8 in poi è possibile disporre ud una visualizzazione sullo schermo del layout della tastiera ed è possibile installare diversi driver di tastiera; quindi selezionando il driver per il greco ed avendo sullo schermo la tastiera virtuale, non è diffcile scrivere direttamente in greco con tutti e caratteri possibili e con tutti i diacritici possibili. Sul Mac la tastiera virtuale è disponibile da anni.

Scrivere in greco diventa ora abbastanza facile, anche se sullo schermo appare la versione traslitterata in lettere latine. Si prenda ad esempio il piccolo file seguente:

```
\documentclass{book}
\usepackage[greek,italian]{babel}
\usepackage{teubner}
%
\begin{document}
%
\begin{quote}\begin{otherlanguage*}{greek}\itshape
To’utou q’arin >ap’elip’on se >en Kr’hth|, <’ina t‘a le’iponta
>epidiort’wsh| ka‘i katast’hsh|s kat‘a p’olin presbit’erous, <ws
>eg’w soi dietax’amen, e>’i t’is >estin >an’egklhtos, mi~as
gunaik‘os >an’hr, t’ekna >’eqwn pist’a, m‘h >en kathgor’ia|
>aswt’ias >‘h >anup’otakta. de~i g‘ar t‘on >ep’iskopon
>an’egklhton e>~inai <ws Jeo~u o>ikon’omon, m‘h a>uj’adh,
m‘h  >org’ilon, m‘h p’aroinon, m‘h pl’hkthn, m‘h a>isqrokerd~h,
>all‘a fil’oxenon, fil’agajon. s’wfrona, d’ikaion, <’osion,
```

Τούτου χάριν ἀπέλιπόν σε ἐν Κρήτῃ, ἵνα τὰ λείποντα ἐπιδιορτώσῃ καὶ καταστήσῃς κατὰ πόλιν πρεσβιτέρους, ὡς ἐγώ σοι διεταξάμεν, εἴ τίς ἐστιν ἀνέγκλητος, μιᾶς γυναικὸς ἀνήρ, τέκνα ἔχων πιστά, μὴ ἐν κατηγορίᾳ ἀσωτίας ἢ ἀνυπότακτα. δεῖ γὰρ τὸν ἐπίσκοπον ἀνέγκλητον εἶναι ὡς Θεοῦ οἰκονόμον, μὴ αὐθάδη, μὴ ὀργίλον, μὴ πάροινον, μὴ πλήκτην, μὴ αἰσχροκερδῆ, ἀλλὰ φιλόξενον, φιλάγαθον. σώφρονα, δίκαιον, ὅσιον, ἐνκρατῆ, ἀντεχόμενον τοῦ κατὰ τὴν διδαχὴν πιστοῦ λόγου, ἵνα δυνατὸς ᾖ καὶ παρακαλεῖν ἐν τῇ διδασκαλίᾳ τῇ ὑγιαινούσῃ καὶ τοὺς ἀντιλέγοντας ἐλέγκειν. T. 1.5-9

Figura 15.2: Esempio di composizione in greco

```
>enkrat~h, >anteq'omenon to~u kat'a t'hn didaq'hn pisto~u l'ogou,
<'ina dunat'os <~h| ka'i parakale~in >en t~h| didaskal'ia| t~h|
<ugiaino'ush| ka'i to'us >antil'egontas >el'egkein.
\null\hfill{\upshape{T. 1.5-9}}
\end{otherlanguage*}
\end{quote}
%
\end{document}
```

Solo pdflatex

Dopo averlo elaborato con pdflatex, si ottiene il risultato mostrato nella figura 15.2 (dove fra l'altro il testo è composto con il font Lipsiano). Si noti anche che il testo sorgente non usa mai la traslitterazione 'c' per il sigma terminale, ma la lettera 's'; questo è voluto per agevolare chi scrive in greco, ma questa non è la sua lingua madre tanto che ha bisogno di usare la tastiera latina. Scrivendo abitualmente in una lingua occidentale sarebbe troppo facile sbagliarsi nello scrivere il sigma terminale; per questo il font è congegnato in modo da rendersi conto da solo di quando il sigma cade in posizione terminale, e automaticamente esegue una legatura di sostituzione, introducendo un sigma terminale, benché nel file sorgente comparisse una 's'. Questo meccanismo è talmente robusto, che diventa difficile scrivere un σ iniziale o mediale isolato; per questo scopo bisogna introdurre nel file sorgente `sv`, dove la 'v' serve per far credere al meccanismo di sostituzione che la 's' introdotta non sia terminale; come si vede dalla tabella 15.2, la lettera latina 'v' non corrisponde a nessun segno greco.

Se si deve scrivere a lungo in greco e si dispone di una tastiera bilingue o, meglio ancora, di mezzi comodi per scrivere direttamente in greco nel file sorgente, si può usare la codifica *utf8*, in questa maniera:

Anche lualatex e xelatex

```
\documentclass{book}
\usepackage{iftex}% Per riconoscere il programma
                  % di composizione
\ifPDFTeX % compilazione con pdflatex
   \usepackage[T1]{fontenc}
   \usepackage{lmodern}
```

```
   \usepackage[utf8]{inputenc}
   \usepackage[greek.ancient,italian]{babel}
\else % compilazione con xelatex o lualatex
   \usepackage{polyglossia}
   \setmainlanguage{italian}
   \setotherlanguage[variant=ancient]{greek}
   \usepackage{fontspec}[Ligatures=TeX]
   \newfontfamily{greekfont}{CMU Classical Serif}
\fi
\begin{document}

\begin{otherlanguage*}{greek}%

   Τούτου χάριν ἀπέλιπόν σε ἐν Κρήτῃ, ἵνα τ‘α λείποντα
   ἐπιδιορτώσῃ κα‘ι καταστήσῃς κατ‘α πόλιν πρεσβιτέρους, ὡς
   ἐγώ σοι διεταξάμεν, εἴ τίς ἐστιν ἀνέγκλητος, μιᾶς
   γυναικ‘ος ἀνήρ, τέκνα ἔχων πιστά, μ‘η ἐν κατηγορίᾳ
   ἀσωτίας ’‘η ἀνυπότακτα. δεῖ γ‘αρ τ‘ον ἐπίσκοπον
   ἀνέγκλητον εἶναι ὡς θεοῦ οἰκονόμον, μ‘η αὐθάδη,
   μ‘η ὀργίλον, μ‘η πάροινον, μ‘η πλήκτην, μ‘η αἰσχροκερδῆ,
   ἀλλ‘α φιλόξενον, φιλάγαθον. σώφρονα, δίκαιον, ὅσιον,
   ἐνκρατῆ, ἀντεχόμενον τοῦ κατ‘α τ‘ην διδαχ‘ην πιστοῦ λόγου,
   ἵνα δυνατ‘ος ᾖ κα‘ι παρακαλεῖν ἐν τῇ διδασκαλίᾳ τῇ
   ὑγιαινούσῃ κα‘ι το‘υς ἀντιλέγοντας ἐλέγκειν.

\null\hfill{\upshape{T. 1.5-9}}
\end{otherlanguage*}

Altro testo in italiano.
\end{document}
```

Nel precedente esempio di codice sorgente, si è mostrato come caricare il file di estensione *iftex*, che mette a disposizione dell'utente i tre test condizionali `\ifPDFTeX`, `\ifLuaTeX` e `\ifXeTeX` con i quali si può controllare con quale programma di compilazione si sta lavorando; Se si sta lavorando con pdflatex vengono eseguiti i comandi di impostazione del preambolo per l'uso con questo compilatore; altrimenti si suppone che l'utente non si sbagli e compili o con xelatex o con lualatex (non sarebbe difficile eseguire i test necessari per verificarlo e per emettere un messaggio di errore nel caso che si usi un altro programma di compilazione) e si imposta il preambolo per lavorare con uno di questi due programmi mediante l'uso di font OpenType. Non si sono specificati i font latini, che per impostazione predefinita sono i Latin Modern. Il resto, in particolare il corpo del documento è lo stesso in entrambi i casi.

15.3 Font adatti ai testi classici

Fra i font usabili con il sistema TEX (in numero sterminato) pochi in verità contengono le legature presenti nei testi classici a stampa, secondo la tradizione dei secoli XVI e XVII. Questi font contenevano sia la s 'lunga' sia la s 'corta', la prima usata come s iniziale e mediale, la seconda come s finale e come secondo elemento della doppia s: *ss* ⟶ *ß*. Contenevano inoltre diverse legature per esempio nei nessi 'ct' e 'st'; contenevano anche le legature æ e œ, assenti nel latino classico, ma entrate in uso sia nei codici tardomedievali, sia negli incunaboli e nelle cinquecentine, assieme all'uso della 'u' minuscola per indicare sia la 'u' sia la 'v' e 'V' maiuscola per indicare sia la 'U' sia la 'V'. La maggior parte di queste abitudini tipografiche cessarono verso la fine del Settecento a partire dai Paesi Bassi per poi estendersi al resto d'Europa e nel resto del mondo.

Desiderando comporre testi classici o antichi conservando l'atmosfera della composizione originale, si possono usare font OpenType 'esperti' che contengano anche le numerose legature. A questo proposito conviene segnalare il lavoro di Massimiliano Dominici [18] che ha usato i font FellType per i quali ha dovuto creare l'insieme di file di descrizione di questi font, rimodificandoli in modo da poterli usare con pdflatex, che non gestisce i font OpenType direttamente; anche i TrueType usati da Dominici si estendevano ben oltre i 256 caratteri gestibili con LATEX.

Per lavorare con pdflatex possiamo fare le seguenti considerazioni. Solo pdflatex

Probabilmente ne esistono altri, ma qui si segnalano i font chiamati con la sigla Kp-fonts[2] e richiamabili nei file da comporre con pdflatex mediante il pacchetto `kpfonts`. Secondo la descrizione che ne ha fatto l'autore Christophe Caingnaert [12], sono direttamente derivati dal font Palatino, adeguatamente reinterpretati dando loro un asse ottico distintamente obliquo. La proprietà di questi font è che dispongono delle serie chiara, media e nera; il maiuscoletto vero può essere composto con lettere 'minuscole' di dimensioni piccole o medie; il maiuscoletto chiaro, medio e nero può essere composto anche in forma inclinata; esistono le varianti che contengono oppure escludono le legature 'ct' e 'st' e usano di default la 'Q' a coda lunga oppure la 'Q' normale; nello stile *veryoldstyle* possono usare anche altre legature; sono presenti sia la s lunga sia la s corta; sono componibili sia con la codifica *OT1* sia con quella *T1*; dispongono delle cifre 'minuscole'; consentono la composizione della matematica in vari stili moderni e antichi, utili anche per le edizioni critiche di testi matematici antichi. Le possibili varianti si ottengono con opportune opzioni esplicitate durante la chiamata del pacchetto `kpfonts`. I simboli disponibili sono moltissimi, molti di più che non unendo le collezioni dei simboli matematici di LATEX e quelli della American Mathematical Society. Questi font sono quindi raccomandabili sia per i testi letterari, sia per le edizioni critiche, sia per testi moderni, anche se contengono molta matematica, composti con una collezione di font molto completa, forse una delle più ricche fra quelle distribuite con il sistema TEX.

[2] Forse le lettere 'kp' derivano dal nome di Kepler, essendo stati sviluppati per il JOHANNES KEPLER project. Non hanno invece nessuna relazione con i font Kepler della Adobe.

Nella figura 18.4 nella pagina 395 si vede l'uso di questi font sia con lo stile compositivo *oldstyle*, sia con lo stile *veryoldstyle* specificati come opzioni al pacchetto *kpfonts*.

lualatex o xelatex

Per lavorare con lualatex e xelatex bisognerebbe esaminare le particolarità dei font OpenType disponibili sulla propria macchina; sul mio Mac tra quelli di sistema, quelli aggiunti dal sistema TEX e quelli che corredano alcuni software, ne ho alcune centinaia; difficili da contare perché i nomi delle famiglie sono diverse centinaia, ma le famiglie contengono da da uno a una dozzina di file di font, i quali a loro volta prevedono diverse varianti. Certamente c'è solo l'imbarazzo della scelta; diversi font contengono varianti che prevedono le legature antiche. Si suggerisce di usare il programma otfinfo da lanciare dalla finestra comandi con le debite opzioni per sapere di quali varianti dispone un dato font; per quali lingue è adatto, se ha corpi ottici, eccetera. Siccome bisogna passare a questo programma il nome completo di path per riceverne le informazioni richieste, bisogna sapere dove si trovano nel disco i font OpenType; ovviamente ogni sistema operativo ha le sue cartelle, quindi qui non si può dire altro se non che i font OpenType distribuiti con il sistema TEX si trovano nella cartella `.../texmf-dist/fonts/opentype/`.

Per gli altri font chi scrive può dirlo solo per il Mac: essi si trovano nella cartella `~/Library/Fonts` (anche quelli installati fra quelli del sistema TEX) e le informazioni richieste si ottengono cliccando col destro, oppure control + click, il nome di un font `.otf` qualsiasi in quella cartella, che apre l'applicativo FontBook.app; la finestra di questo applicativo contiene nella parte di sinistra l'elenco alfabetico di tutti i font installati sulla macchina, e di ciascuno può mostrare le informazioni scritte, un esempio di testo composto con quei font, la tabella dei caratteri, eccetera.

15.4 La composizione del tedesco classico

Fra le lingue classiche e moderne solo il tedesco è molto noto per usare i font gotici, decisamente diversi da quelli latini, anche se le lettere, semanticamente sono le stesse.

Dalla prima Bibbia di Gutenberg alla fine della seconda guerra mondiale il gotico è stato spesso usato ma, anche impropriamente, come simbolo di una nazione. Per il nazismo era stato l'alfabeto caratterizzante quel regime.

I modelli della scrittura a cui in italiano ci si riferisce con l'aggettivo *gotico* sono sostanzialmente tre: (*a*) il textur usato da Gutenberg, (*b*) il fraktur, usato nei secoli più recenti e, (*c*) lo schwabacher usato come alfabeto elegante, sempre di tipo gotico, ma con le lettere più rotondeggianti, specialmente le maiuscole. Nella figura 15.3 sono mostrati alcuni alfabeti gotici disponibili con il sistema TEX.

In realtà anticamente non c'era nessuna distinzione fra la 'i' e la 'j'; nei font mostrati nella figura 15.3 le due lettere minuscole sono leggermente differenziate, mentre le maiuscole sono rimaste identiche.

Font textur

abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ

Font fraktur

abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ

Font schwabacher

abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ

Capilettera

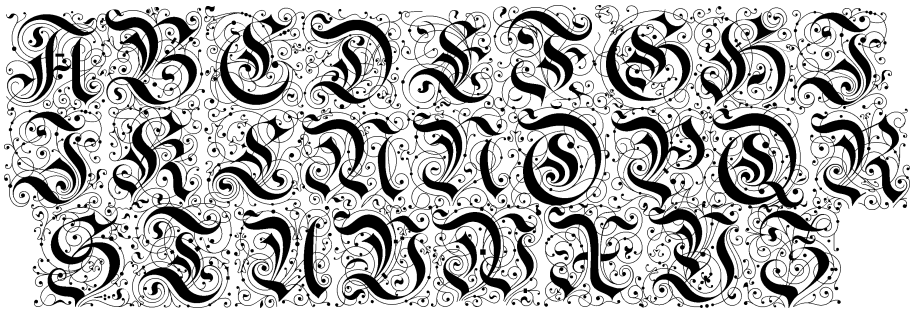


Figura 15.3: Alcuni alfabeti gotici che includono anche i capilettera ornati

Si noti che nelle distribuzioni di TEX Live recenti, certamente nella distribuzione del 2010, i tre font textur, fraktur e schwabacher sono disponibili come font vettoriali e sono già disponibili come tali senza bisogno di ricorrere all'installazione delle loro mappe. I font corrispondenti ai bellissimi e ornatissimi capilettera sono in forma bitmapped; forse la loro delicata e fitta ornamentazione impedisce loro di venire trasformati in font vettoriali, ma nelle prove fatte su carta non si nota nessun difetto tipico dei font bitmapped; anche sullo schermo si presentano molto bene.

Per fare uso di questi font basta impiegare il pacchetto *mfnfss*, che definisce i comandi usuali `\gothfamily` e `\textgoth` per selezionare il font textur, `\frakfamily` e `\textfrak` per selezionare il font fraktur, `\swabfamily` e `\textswab` per selezionare il font schwabacher. Per selezionare il font dei capilettera il file di documentazione suggerisce una macro che contemporaneamente

Solo pdflatex

sceglie il font e lo compone ribassato; qui la si riporta adeguatamente "corretta" visto che come è riportata nella documentazione fa riferimento ad una macro non descritta:

```
\newcommand*\yinitial[1]{%
\hangindent=2.54cm
\hangafter=-4
\hskip-3.24cm
\lower-2.7mm
\hbox{\fontencoding{U}\fontfamily{yinit}\selectfont #1}%
\hskip1.5mm}
```

dove le misure indicate possono essere ritoccate a discrezione dell'utente.

I caratteri contengono le necessarie legature antiche e per le umlaut consentono di usare sia il doppio apice (`"a` produce ä) sia l'asterisco (`*a` produce una 'a' a cui è sovrapposta come una segno diacritico una minuscola 'e'); queste seconde legature sono specificamente descritte per lo schwabacher.

In sostanza, se si dovesse riprodurre una pagina della Bibbia di Gutenberg, lo si potrebbe fare in modo quasi identico, compresi i capilettera, anche se questi hanno un disegno di un paio di secoli successivi a quelli usati da Gutenberg stesso.

15.5 La composizione di testi in lingue classiche

In questo breve paragrafo si accenna alla scrittura in latino e in greco, che si ritiene che siano le lingue classiche più diffuse fra gli umanisti italiani. Non si scende nei dettagli di che cosa sia necessario fare con scritture molto particolari; si segnala solo che fra i font del sistema TEX compare anche il Lineare A, per esempio, al fine di riportare testi molto antichi.

15.5.1 Scrivere in latino

Solo pdflatex

Scrivere in latino non richiede altro che specificare l'opzione *latin* a `babel`; tuttavia se si desidera estendere la funzionalità alla marcatura della lunghezza delle sillabe, bisogna provvedere alla specificazione dell'attributo della lingua:

```
...
\usepackage[...,latin,...]{babel}
\languageattribute{latin}{withprosodicmarks}
...
```

Un altro attributo che si può specificare è quello relativo all'ortografia medievale:

```
...
\usepackage[...,latin,...]{babel}
\languageattribute{latin}{medieval}
...
```

oppure al latino classico:

```
...
\usepackage[...,latin,...]{babel}
\languageattribute{latin}{classic}
...
```

La differenza nel latino medievale è che in lettere minuscole le lettere 'u' e 'v' si scrivono entrambe 'u', mentre in lettere maiuscole esse vengono scritte entrambe 'V'. Di per sé non sarebbe un problema per il tastierista a scriverle come appena detto, ma quando ci sono delle trasformazioni automatiche in minuscolo o in maiuscolo, i comandi `\MakeUppercase` e `\MakeLowercase` devono operare correttamente anche con l'ortografia medievale. Questa regola vale anche per il latino classico, ma questo non usa le legature æ ed œ, mentre il latino medievale ne fa ampio uso. Le due ortografie differiscono anche per la divisione in sillabe.

Una cosa che va osservata attentamente è che la divisione in sillabe sia con l'ortografia moderna sia con quella medievale funziona bene in modo fonetico; quella del latino classico funziona con le regole etimologiche tradizionalmente usate dai latinisti; buona parte dei suffissi e dei prefissi sono riconosciuti correttamente in modo da dividerli etimologicamente, ma l'insieme di pattern per la sillabazione latina non garantisce sempre una divisione corretta. Per questo diventa particolarmente importante l'uso del carattere attivo `"` che introduce un punto di cesura facoltativo senza impedire la divisione del resto della parola. Si scriverà pertanto `transi"erat` per consentire la divisione dopo la 'i' senza impedirla negli altri punti leciti.

Le date vengono composte con il calendario gregoriano, ma i numeri relativi ai giorni e agli anni vengono scritti con la numerazione romana sottrattiva; in maiuscoletto il numero dei giorni, in maiuscolo quello degli anni. Un tentativo di trasformare la data gregoriana in una data "ab urbe condita" non ha avuto seguito (per ora).

Per usare gli altri programmi di compilazione non ci sono problemi con i font latini; per la scrittura medievale e liturgica sono sempre disponibili i caratteri æ ed œ, semplici e accentati; veramente œ́ non è codificato UNICODE, quindi o lo si ottiene con la macro per l'accento acuto, o si usa il carattere UNICODE `U+0301`, *self combining acute accent* che va indicato in posizione postfissa, `œ^^^^0301`, con una scrittura veramente inconsueta. Certo se ne può costruire una macro, ma il problema non è quello. Il vero problema è che per polyglossia il team *tex-hyphen* prende tempo nell'accettare ciò che i curatori della sillabazione latina propongono e, in particolare, per lualatex, che carica i pattern al momento del bisogno, non accetta la definizione di nuovi nomi di lingue e di conseguenza non accetta nuovi file `gloss-`⟨*latin*⟩ con nomi diversi per le varietà delle ortografie latine. Il tutto ora (inizio 2017) è in mano a nuovi curatori e speriamo che questo intoppo si sblocchi in breve tempo. Dal 2018 è possibile usare la selezione del latino classico, moderno e medievale, e liturgico con nomi espliciti, quindi la funzionalità di LuaLaTeX è sfruttata al meglio.

xelatex o lualatex

15.5.2 Scrivere in greco

Si è già detto molto sia in questo capitolo, sia nel paragrafo 15.2. Non si ripeteranno quelle cose, qui, ma si sottolinea l'uso opportuno e consigliabile del pacchetto *teubner* nella versione 4.x. Infatti in questa versione sono stati aumentati molto i comandi per gestire le varie forme dei font greci (CBgreek) tenendo conto che in un testo letterario o filologico verosimilmente si passa attraverso un'alternanza di testo in lingua greca con altro testo in una lingua occidentale.

Solo pdflatex

La gestione dei segni diacritici è anche molto migliorata e permette di evitare di usare le decine di macro strane per comporre i caratteri marcati con i loro diacritici senza ricorrere al meccanismo delle legature; questo si rende necessario specialmente quando si usa il font Lipsiano i cui segni hanno disegni dalle forme delicate che richiedono crenature accurate.

Quando il testo greco è abbondante, sarebbe desiderabile che l'umanista disponesse di una tastiera bilingue, oppure di comodi mezzi software per disporre di una tastiera virtuale su cui comporre il testo greco. In queste condizioni il 'tastierista' può comporre direttamente in greco senza doversi ricordare nessuna corrispondenza fra la tastiera con caratteri latini e quella con i caratteri greci. Però necessita dell'opzione *utf8* per la codifica di entrata; questa particolare opzione permette di tradurre i segni greci UNICODE, che appaiono sullo schermo quando si compone il file sorgente, nei corrispondenti comandi interni (LICR, LaTeX Internal Character Representation) che pdflatex usa e che solo in fase di scrittura del file di uscita trasforma nei caratteri presenti poi per il font specificato con la codifica di uscita; si veda anche il capitolo 26.

Per il greco esistono almeno tre metodi di divisione in sillabe, corrispondenti alle tre ortografie o varietà della lingua: la sillabazione per l'ortografia monotonica, quella per l'ortografia politonica moderna e quella per l'ortografia politonica antica. Si tenga presente che di default le impostazioni per la sillabazione greca con qualunque ortografia prevede una lunghezza minima della prima e dell'ultima sillaba di qualunque parola formata da una sola lettera, quando in italiano esse devono essere formate da almeno due lettere, e nelle altre lingue occidentali con molte parole che finiscono in consonante la prima sillaba deve essere di almeno due lettere e l'ultima sillaba di almeno tre lettere. Queste vistose differenze fanno sì che la prosa normale possa avere capoversi che terminano con un righino contenente una sola lettera, tipograficamente molto brutto; LaTeX pone una fortissima penalità alla cesura nella penultima riga di un capoverso, ma questo non impedisce che possa verificarsi quell'evento.

I vari esempi di testo greco classico scritti in questa guida sono stati tutti composti usando il pacchetto *teubner* e l'attributo `ancient` e senza avvalersi di una tastiera greca né della codifica *utf8*. A questo proposito, visto che oggi gli elaboratori con schermo *touch screen* diventano sempre più diffusi, disponendo di un tale dispositivo, non è impossibile scrivere direttamente in greco toccando lo schermo in corrispondenza dei tasti virtuali della tastiera greca; basta aver installato il driver della tastiera greca ed aver toccato la sua icona nella barra

opportuna dello schermo. Ecco allora che i sistemi *touch screen*, che lo scrivente considerava validi solo per certi usi che nulla hanno a che far con la composizione dei testi, diventano utili proprio a questo scopo quando si debbano usare alfabeti diversi o disposizioni diverse dei tasti.

Solo xelatex e lialatex

Va poi sottolineato che per scrivere agevolmente con alfabeti diversi da quello latino è utile (o necessaria) una tastiera apposita, come detto sopra, ma diventa utilissimo anche usare il programma di composizione xelatex o lualatex; come già spiegato, questi programmi consentono di usare la codifica UNICODE e di gestire direttamente i font OpenType, cosicché la scelta dei font latini, greci, cirillici, eccetera, non è più legata a quanto offre di default il sistema TEX, ma si estende alla moltitudine di font che formano la dotazione del sistema operativo della macchina che si sta usando. Esso apre le porte all'impiego di qualsiasi font commerciale, gratuito o a pagamento, senza bisogno di ricorrere alle procedure di installazione necessarie con pdfLATEX, che però non può gestire direttamente i font OpentType. Per il resto i programmi di composizione xelatex e lualatex hanno un mark-up che differisce da quello di LATEX solo per qualche dettaglio riguardante la gestione dei font, ma per il resto un file sorgente, concepito per essere elaborato con pdfLATEX, spessissimo viene compilato senza modifiche anche da XELATEX e LuaLATEX.

Vale la pena di citare il pregevolissimo lavoro fatto da Antonis Tsolomitis che ha reso disponibili i font greci predisposti dalla società GFS (Greek Font Society) in formato OpenType e che questa società ha distribuito in uso gratuito al pubblico; Tsolomitis ha reso questi font accessibili anche a pdflatex; creando i debiti file virtuali e codificando questi font con la codifica LGR, come i font greci preinstallati. I file per usarli sono i seguenti:

`gfsartemisia` corrisponde al pacchetto da invocare per usare i font greci e latini Artemisia.

`gfsbaskerville` corrisponde al pacchetto da invocare per usare i font greci Baskerville, ed è caratterizzato da diverse forme per i glifi iniziali o mediali di gamma, theta, rho e per le legature anticheggianti fra sigma e theta, sigma e chi, upsilon e iota, omicron e sigma e per il doppio tau; si possono usare altre legature, riguardanti specialmente il dittongo omicron-upsilon, mediante comandi specifici.

`gfsbodoni` corrisponde al pacchetto da invocare per usare i font latini e greci Bodoni caratterizzati da lettere latine dalle grazie finissime e dal forte contrasto e da lettere greche dalle forme leggermente calligrafiche.

`gfscomplutum` corrisponde al pacchetto da invocare per usare i font greci Complutum con forme anticheggianti e difficile da leggere per chi non ha familiarità con questo stile.

`gfsdidot` corrisponde al pacchetto da invocare per usare i font latini e greci Didot; questo è il font che si trova usato più spesso nei testi greci; questa particolare polizza distingue la forma della theta iniziale aperta, dalla

forma finale chiusa; altre legature sono possibili mediante comandi appositi; la forma obliqua è ottenuta automaticamente mediante il font Olga.

`neohellenic` corrisponde al pacchetto da invocare per usare i font greci di tipo bastoncino Neohellenic; il font lineare o bastoncino è caratterizzato dalla mancanza di chiaroscuri e tutti i tratti sono sostanzialmente della stessa larghezza, abbastanza sottile.

`gfsporson` corrisponde al pacchetto da usare per invocare i font inclinati Porson, caratterizzati da un disegno moderno, ma con l'accento circonflesso ridotto a un cappellino rotondeggiante; purtroppo la documentazione non offre nessun esempio d'uso del font Porson, ma basta scrivere un breve testo per verificare lo stile di questo font.

`gfssolomos` corrisponde al pacchetto da usare per invocare i font greci Solomos la cui forma è molto calligrafica con i discendenti delle lettere minuscoli abbastanza ornati da riccioli e curve sinuose.

Scrivendo nel terminale il comando `texdoc` ⟨*pacchetto*⟩ (per esempio `texdoc gfsdidot`) si apre il file di documentazione che nelle ultime pagine mostra un po' di testo, eventualmente contenente un po' di matematica, dove si possono notare le caratteristiche del font; purtroppo questi testi sono quasi tutti scritti in greco monotonico, quindi non è possibile osservare la forma dei segni diacritici.

Non si tenga conto in tutti questi file di documentazione del paragrafo intitolato "Transformations by dvips"; i medesimi effetti si ottengono oggi lavorando direttamente con pdflatex probabilmente anche negli anni in cui Tsolomitis creava queste funzionalità aggiuntive.

Il lavoro di Tsolomitis è decisamente importante perché integra molto bene sia il testo latino, sia quello greco, fra testo e matematica. La scelta di integrare con alcuni font i segni latini insieme a quelli greci permette di eseguire degli abbinamenti ottimali, che sono poi quelli che generalmente fanno i tipografi greci in Grecia.

Si noti ancora che Antonis Tsolomitis ha prodotto anche un file `txfontsx` da usare insieme al pacchetto per i caratteri latini `txfonts`. Le due serie di caratteri sono perfettamente integrate. Ne tengano conto coloro che debbono comporre in greco, certamente monotonico, ma anche in generale in greco politonico, integrando il testo con i corrispondenti caratteri latini. Come parte interessata alla codifica *LGR*, devo però avvisare che con i font di Tsolomitis è difficile sfruttare appieno le funzionalità del pacchetto *`teubner`*, perché non tutti i glifi della codifica *LGR* sono sono presenti nelle polizze di Tsolomitis. Gli stessi problemi nascono dal confronto sulla dotazione di caratteri dei font diritti rispetto a quelli inclinati, che non si corrispondono completamente. Con i font Times eXtended, il pacchetto *`teubner`* ha posto rimedio, ma non è detto che quella soluzione possa andare bene anche con gli altri font descritti qui.

Infine Tsolomitis si è dedicato alla creazione di una famiglia di font OpenType che contengono i caratteri latini, greci e cirillici, già presente nella dotazione di

TEX Live; si tratta della collezione NewComputerModern il quale è accompagnato anche dalla collezione dei font matematici; si legga il file di documentazione con `texdoc newcomputermodern` per verificare le famiglie, le serie e le forme insieme a qualche formula matematica.

15.6 La composizione di testi filologici

La pubblicazione di scritti filologici include le edizioni critiche e include scritti in prosa o in versi in lingue scritte non necessariamente in caratteri latini. Essi sono scritti letterari, ma in più richiedono diversi apparati critici formati da annotazioni composte in varie maniere. Questi apparati di note spesso compaiono assieme nella stessa pagina, ma devono venire distinti in modo inequivocabile gli uni dagli altri e, chiaramente, non basta solo un filetto orizzontale per distinguerli.

Serve anche una numerazione delle singole righe sia per i testi in prosa, sia per quelli in poesia; generalmente i numeri sono nel margine esterno e sono esplicitati quelli multipli di 5; le note, invece che legate a richiami come numerini a esponente, possono essere legate al numero di riga, per cui nel richiamo bisogna ripetere una breve sequenza di testo a cui la nota si riferisce.

Chiaramente LATEX non è in grado da solo di svolgere questi compiti e bisogna ricorrere a pacchetti di estensione.

Il più noto e storico pacchetto di estensione si chiama *ledmac* ed è in grado di comporre ogni aspetto delle edizioni critiche; la sua documentazione si legge con `texdoc ledmac`; oggi è però soppiantato dalla sua versione estesa costituita dal pacchetto *eledmac*; documentazione `texdoc eledmac`; ultimamente il curatore ha anche prodotto *reledmac*, documentazione `texdoc reledmac`.

L'autore di *ledmac*, Peter Wilson, ha cercato di implementare le capacità sviluppate dai pacchetti di macro per Plain TEX: *edmac*, *tabmac* e *edstanza* aggiungendovi del suo. Oggi il curatore è Maïeul Rouquette, che ha esteso ulteriormente le funzionalità del pacchetto originale di Wilson, adattandolo anche a mezzi moderni di programmazione del sistema TEX.

Esso è particolarmente indicato per le edizioni critiche e il suo forte sono le numerazioni marginali e gli apparati di note sia a piè di pagina sia come note finali a fine capitolo o a fine documento. È possibile legare le note finali al numero di pagina e di riga, invece che a richiami numerici ad esponente. Anche per quel che riguarda l'indice analitico è possibile fare riferimento anche al numero di riga invece che solamente alla pagina.

Altri pacchetti per edizioni critiche sono *endnotes* specializzato nelle note finali, e *poemscol* destinato alle forme poetiche, ai testi in versi.

Va per esempio citato il pacchetto di estensione *opcit* che dispone di un suo particolare file di stile per la composizione della bibliografia con BIBTEX; il pacchetto consente di scrivere i riferimenti bibliografici in nota, invece che in elenco a fine capitolo o a fine documento; le citazioni successive alla prima vengono ripetute in forma abbreviata nella forma "Autore, *op. cit.*"; se la precedente citazione corrisponde alla stessa opera, il comando di citazione riporta solamente

"*Ibidem*, p. 33" (dove l'indicazione della pagina, la post-citazione, costituisce l'argomento facoltativo del comando `\cite`). Sono possibili diverse personalizzazioni così che gli umanisti possano continuare ad usare il loro metodo tradizionale di citazione in nota. L'uso di LaTeX e dei suoi pacchetti assicura l'aspetto omogeneo delle citazioni in modo da garantire la massima professionalità compositiva. La documentazione del pacchetto si trova in `.../doc/latex/opcit/opcit.pdf`.

Quando si va nella filologia vera e propria, le cose si possono fare molto dure, anche perché i filologi delle varie discipline hanno notazioni diverse per rendere gli stessi concetti.

Chi scrive ha collaborato con un filologo di greco classico e con un filologo di lingua copta altomedievale; come prevedibile, i due filologi non erano d'accordo su come esprimere i concetti comuni alle due lingue; questa non è una critica, ma una osservazione; anch'io che scrivo usando molta matematica ho dovuto arrangiarmi con i simboli disponibili perché certi concetti che dovevo esprimere, malgrado le norme ISO, non avevano un simbolo specifico, così alla fine ho dovuto usare simboli che in altre branche della matematica hanno significati diversi. Succede in ogni scienza e anche all'interno della stessa scienza.

Per quanto concerne il greco classico, come già detto, esiste il pacchetto *teubner* già contenuto in ogni installazione aggiornata e completa e abbastanza ben documentato mediante il file `.../doc/latex/teubner/teubner-doc.pdf`. Però per ora questo pacchetto funziona solo con pdflatex. Si noti che l'interfaccia fra *babel* e il greco è cambiata nel 2020, per cui ora *teubner*, versione 4.x, funziona solo con una versione aggiornata dell'installazione del sistema TeX.

Dalla versione 3.x il pacchetto *teubner* dispone di una nuova funzionalità, oltre a macro per gli accenti che rendono inutile il ricorso a quel centinaio di macro per accedere per nome ai caratteri accentati invece che con le comode legature, permette di far convivere i font con codifica *LGR* virtualmente con tutti i font disponibili sull'elaboratore che si sta usando. Già la versione 2.9 era compatibile con i font CM-super e i font Latin Modern, tutti con la codifica *T1*. Con la versione 3.x il pacchetto riconosce da solo se si stanno usando i font Times o Palatino eXtended e si crea da solo i file di descrizione dei font necessari per questa convivenza. Il pacchetto mette a disposizione anche il comando `\ifFamily` che crea le necessarie associazioni anche con famiglie di font diverse da quelli sopra menzionati.

Per esempio, se si volessero usare il font Helvetica come font latini proporzionali senza grazie, allora basterebbe dare i seguenti comandi:

```
\renewcommand{\sffamily}{phv} % Font latini senza grazie
\ifFamily{phv}{lmss}          % Font greci senza grazie
```

In questo modo *teubner* genera i file di descrizione dei font che sostituiscono agli inesistenti Helvetica codificati *LGR* i font greci della collezione CBgreek con la codifica giusta; certo i segni grafici sono diversi, ma se è possibile i segni greci sono meno vistosi e ingombranti dei segni lineari con lo stile Helvetica. Nessuno vieta poi di usare lualatex o xelatex, come mostrato alla fine di questo capitolo,

così da evitare le piccole acrobazie necessarie per poter usare il programma pdflatex, che non è in grado di gestire da solo i font codificati in UNICODE.

Il pacchetto consente di accentare ogni lettera con qualsiasi segno diacritico, persino con accenti impilati, collocandoli correttamente tenendo conto dell'inclinazione dei caratteri. Esso può inserire segni speciali sopra o sotto singole lettere o gruppi di lettere; può racchiudere gruppi di lettere con delimitatori speciali per indicare la lectio incerta, o altre simili indicazioni per interpolazioni o omissioni; può numerare i versi di poemi anche con due serie di numerazioni contemporaneamente; può disegnare le metre dei vari versi con una notevole ricchezza di segni che consentono anche di specificare, per esempio, se una sillaba ancipite è più sovente lunga invece che breve, eccetera.

Va detto che Gianfranco Boggio-Togna ha prodotto anche il pacchetto *metre* che svolge alcune funzioni svolte dal pacchetto *teubner*. I due pacchetti differiscono per la loro estensione, ma anche, quando si tratta delle metre, le notazioni ritmiche dei versi, per come sono prodotti i simboli da usare; *teubner* usa un font particolarmente disegnato allo scopo, mentre *metre* usa segni costruiti componendoli con i simboli dei vari font già disponibili, per esempio, usando simboli tratti dalle polizze per la matematica.

Per quanto concerne il copto, oltre ai font necessari per scrivere con i caratteri altomedievali, c'è una certa varietà di simboli e segni che permettono di annotare il testo antico; con le distribuzioni recenti di TeX Live, anche il pacchetto *coptic* è già preinstallato.

Certamente esistono altri pacchetti in rete destinati a risolvere problemi specifici di composizione filologica; tuttavia ogni lavoro di filologia richiede la predisposizione di comandi particolari per problemi particolari; lo scrivente è al corrente che da alcuni anni a Pisa si sta costruendo l'edizione critica delle opere del matematico Maurolico; sono già disponibili diverse parti, ma l'opera non è finita; si capisce bene che, nonostante il Maurolico scrivesse in latino, egli scriveva di matematica e di geometria quando le convenzioni a cui siamo oggi abituati non esistevano ancora, non parliamo delle norme ISO! I curatori di questa edizione critica stanno lavorando con passione ma devono inventarsi soluzioni nuove per problemi di composizione che finora non sono mai stati affrontati, meno che mai risolti.

In Italia, oltre che a Pisa, si è al corrente che anche alla Pontificia Università di Santa Croce di Roma c'è un gruppo di utenti che si occupa di filologia e di edizioni critiche. Essi hanno anche predisposto un corso e un CD-ROM per gli utenti del corso.

15.7 Un esempio di composizione con XeLaTeX

Si vuole qui presentare un breve esempio di composizione in greco classico politonico facendo uso del programma xelatex. Si presenta dapprima il file sorgente e dopo qualche commento il risultato della composizione.

Solo xelatex e lualatex

Dividiamo l'esempio in due parti: (*a*) l'esempio vero e proprio, e (*b*) alcuni commenti in merito alla selezione dei font.

Un'utile introduzione a XeLaTeX, scritta in modo relativamente semplice, è costituita dal testo di Enrico Gregorio, [31]; essa permette di verificare e di approfondire molti degli argomenti esposti qui e di studiarne diversi altri. Esiste anche un manuale iniziale e più o meno ufficiale di XeTeX, [26], che però deve essere ancora considerato sperimentale, almeno non così approfondito come gli altri manuali della serie "The [...] Companion". Un'altra breve guida per XeTeX è stata predisposta da Will Robertson, [55], uno dei collaboratori più importanti alla squadra che si occupa della diffusione di questo programma. In italiano non bisogna assolutamente trascurare il testo [40] scritto apposta per gli umanisti. Per LuaLaTeX i manuali spiegano prevalentemente come servirsi del linguaggio Lua, perché per il resto si comportano come XeLaTeX.

15.7.1 Esempio di composizione in greco

Il file sorgente è stato predisposto con lo *shell editor* TeXShop su una piattaforma Mac nella quale era installato anche il driver per la tastiera greca politonica; le parti in greco sono state scritte direttamente usando la tastiera greca.

Il preambolo è il seguente:

```
%!TEX encoding = UTF-8 Unicode
%!TEX TS-program = xelatex

\documentclass[b5paper,11pt,titlepage]{book}

\setmainfont[Ligatures=TeX,Numbers=OldStyle]{TeX Gyre Bonum}
\setsansfont[Ligatures=TeX,Scale=MatchUppercase]{Futura}

\newfontfamily{\greekfont}{Old Standard}

% Scelta delle lingue

\usepackage{polyglossia}
\setmainlanguage{italian}
\setotherlanguage[variant=ancient]{greek}

\begin{document}
```

Le prime due righe del file specificano a TeXShop l'encoding con cui è stato scritto il file sorgente e il nome del programma con il quale, cliccando il pulsante Typeset, egli deve agire, indipendentemente dalle impostazioni predefinite. Non si insisterà mai abbastanza sulla comodità ed opportunità della presenza delle due righe suddette; TeXShop, TeXworks e TeXstudio interpretano queste righe ed eventualmente modificano temporaneamente le proprie impostazioni; gli altri

```
\chapter{Su Archimede}

\section{Testo originale in greco}

\begin{otherlanguage}{\greek}
Οἱόνται τινές, βασιλεῦ Γέλων, τοῦ ψάμμον τὸν
ἀριθμὸν ἄπειρον εἶμεν τῷ πλήθει· λέγω δὲ οὐ μόνον τοῦ περὶ Συρακούσας τε
καὶ τὰν ἄλλαν Σικελίαν ὑπάρχοντος, ἀλλὰ καὶ τοῦ κατὰ πᾶσαν χώραν τάν τε
οἰκημέναν καὶ τὰν ἀοίκητον. ἐντί τινες δέ, οἳ αὐτὸν ἄπειρον μὲν εἶμεν οὐχ
ὑπολαμβάνοντι, μηδένα μέντοι ταλικοῦτον κατωνομασμένον ὑπάρχειν, ὅστις
ὑπερβάλλει τὸ πλῆθος αὐτοῦ. οἱ δὲ οὕτως δοξαζόντες δῆλον ὡς εἰ νοήσαιεν
ἐκ τοῦ ψάμμον ταλικοῦτον ὄγκον συγκείμενον τὰ μὲν ἄλλα, ἁλίκος ὁ τᾶς γᾶς
ὄγκος, ἀναπεπληρωμένων δὲ ἐν αὐτῷ τῶν τε πελαγέων πάντων καὶ τῶν
κοιλωμάτων τᾶς γᾶς εἰς ἴσον ὕψος τοῖς ὑψηλοτάτοις τῶν ὀρέων,
πολλαπλασίως μὴ γνωσόνται μηδένα κα ῥηθήμεν ἀριθμόν ὑπερβάλλοντα τὸ
πλῆθος αὐτοῦ. ἐγὼ δὲ πειρασούμαι τοι δεικνύειν δι' ἀποδειξίων γεομετρικᾶν,
αἷς παρακολουθήσεις, ὅτι τῶν ὑφ' ἁμᾶς κατωνομασμένων ἀριθμῶν καὶ
ἐνδεδομένων ἐν τοῖς ποτὶ Ζεύξιππον γεγραμμένοις ὑπερβάλλοντί τινες οὐ
μόνον τὸν ἀριθμὸν τοῦ ψάμμου τοῦ μέγεθος ἔχοντος ἴσον τᾷ γᾷ
πεπληρωμένᾳ, καθάπερ εἴπαμες, ἀλλὰ καὶ τὸν τοῦ μέγεθος ἴσον ἔχοντος τῷ
κόσμῳ.

\end{otherlanguage}
\end{document}
```

Figura 15.4: Completamento del file sorgente

editor possono mancare di questa funzionalità o possono averla ma con una sintassi diversa. La persona che ricevesse questo file da altri vedrebbe subito non appena apre il file se eventualmente deve riconfigurare il suo *shell editor* per poter comporre il documento.

Le tre righe dopo la dichiarazione di classe (che è del tutto normale) specificano il modo di dichiarare i font Opentype da usare nel documento. Per il testo in caratteri latini si usa un facsimile del font Bookman attraverso il font della collezione TeX Gyre, già presenti nella distribuzione del sistema TeX. Le opzioni chiedono di usare le legature standard del sistema TeX e chiedono di usare le cifre arabe minuscole (old style).

Il secondo font, Futura, viene usato solo per i caratteri senza grazie; l'opzione serve per scalarlo un pochino affinché le sue lettere minuscole senza ascendenti e discendenti abbiano la stessa grandezza di quelle del font con grazie. Il terzo font, Old Standard, di per sé potrebbe essere usato anche come "mainfont", perché contiene anche i caratteri latini, greci e cirillici, oltre a una quantità di altri segni; ma qui lo si definisce solo come famiglia di caratteri da assegnare alla composizione per la lingua greca; poiché il suo nome `\greekfont` comincia con il nome della lingua, esso viene usato automaticamente come font di default quando si scrive in quella lingua; lo stesso espediente può essere usato per differenziare con la forma o lo stile del font la lingua in cui si sta scrivendo.

Seguono poi i tre comandi per la scelta delle lingue; il pacchetto *polyglossia* sostituisce *babel*; in realtà xelatex e lualatex accettano anche di usare il pacchetto *babel* e il suo modo di specificare le lingue; ma, secondo lo scrivente, è preferibile

usare *polyglossia* perché è già costruito per lavorare con xelatex e lualatex e i loro font OpenType. Inoltre è chiaro qual è la lingua di default, il "mainlanguage" e quali sono le lingue accessorie; qui, in particolare, si specifica il greco con l'opzione che si tratta della variante del greco antico,

Come si vede il preambolo è piuttosto semplice (oltre che esuberante per il piccolo esempio che si sta mostrando).

Il corpo del documento contiene le righe contenute nel medaglione della figura 15.4. Lanciando xelatex o lualatex su questo piccolo file sorgente si ottiene il risultato mostrato nella figura 15.5.

Come si vede chiaramente il font TEX Gyre Bonum propone dei caratteri molto scuri mentre il greco tratto dal font Old Standard è della serie media che si accorderebbe benissimo con i font Latin Modern usati in questo testo. L'accoppiata è quindi infelice, ma la si è mostrata lo stesso per indicare nell'esempio quali generi di font si possono usare con i programmi xelatex e lualatex.

15.7.2 XeLaTeX, i font OpenType e i font Type 1

Quanto si dice qui per XeLaTeX vale anche per LuaLaTeX.

Nel paragrafo precedente abbiamo visto come specificare i font per il testo, come scegliere diversamente i font per una diversa famiglia (sans serif rispetto ai font con grazie) o per una lingua diversa. Quel modo di specificare i font vale anche per i font TrueType se contengono le tabelle con le specificazioni delle varianti come gli OpenType. Secondo gli standard e l'uso corrente, i font Type 1 hanno solo 256 caratteri per ogni polizza, senza varianti e hanno i contorni dei vari segni descritti mediante spline di Bézier di terzo grado. I font TrueType hanno polizze con un numero di segni non limitati a 256, ma possono rispettare le specifiche UNICODE; hanno i contorni dei segni descritti da spline di secondo grado.

Talvolta potrebbe essere necessario specificare l'uso di font Type 1; di questi bisogna specificare la codifica e chiamarli con il loro nome vero, non quello che usa pdflatex; questi nomi sono indicati nel file `pdftex.map` come seconda voce di ogni riga; bisogna stare attenti che possono essere costituiti da diverse parole. Per esempio se uno desiderasse usare i font URW Avant Garde Light Book, la riga corrispondente nel file `pdftex.map` sarebbe:

```
pagk8r URWGothicL-Book "TeXBase1Encoding ReEncodeFont" <8r.enc"<uagk8a.pfb
```

dove `pagk8r` è il nome del file `pagk8r.pfb` che contiene i disegni dei segni di questo particolare font a cui si riferisce il file `pag.fd` di descrizione dei font invocato in definitiva dal pacchetto *avant*. Questo font Type 1 deve essere ricodificato, perché nativamente non ha la codifica *T1*; ecco perché la riga del file `pdftex.map` è così relativamente complessa. Per il nostro caso quello che importa è il nome vero del font da dedurre dal nome compresso `URWGothicL-Book`: il nome completo è `URW Gothic L` e questo bisogna usare nei comandi accettati dal programma xelatex.

Capitolo 1

Su Archimede

1.1 Testo originale in greco

Οἰόνται τινές, βασιλεῦ Γέλων, τοῦ ψάμμον τὸν ἀριθμὸν ἄπειρον εἶμεν τῷ πλήθει· λέγω δὲ οὐ μόνον τοῦ περὶ Συρακούσας τε καὶ τὰν ἄλλαν Σικελίαν ὑπάρχοντος, ἀλλὰ καὶ τοῦ κατὰ πᾶσαν χώραν τάν τε οἰκημέναν καὶ τὰν ἀοἰκητον. ἐντί τινες δέ, οἳ αὐτὸν ἄπειρον μὲν εἶμεν οὐχ ὑπολαμβάνοντι, μηδένα μέντοι ταλικοῦτον κατωνομασμένον ὑπάρχειν, ὅστις ὑπερβάλλει τὸ πλῆθος αὐτοῦ. οἱ δὲ οὕτως δοξαζόντες δῆλον ὡς εἰ νοήσαιεν ἐκ τοῦ ψάμμον ταλικοῦτον ὄγκον συγκείμενον τὰ μὲν ἄλλα, ἁλίκος ὁ τᾶς γᾶς ὄγκος, ἀναπεπληρωμένων δὲ ἐν αὐτῷ τῶν τε πελαγέων πάντων καὶ τῶν κοιλωμάτων τᾶς γᾶς εἰς ἴσον ὕψος τοῖς ὑψηλοτάτοις τῶν ὀρέων, πολλαπλασίως μὴ γνωσόνται μηδένα κα ῥηθήμεν ἀριθμόν ὑπερβάλλοντα τὸ πλῆθος αὐτοῦ. ἐγὼ δὲ πειρασούμαι τοι δεικνύειν δι' ἀποδειξίων γεομετρικᾶν, αἷς παρακολουθήσεις, ὅτι τῶν ὑφ' ἁμῶν κατωνομασμένων ἀριθμὼν καὶ ἐνδεδομένων ἐν τοῖς ποτὶ Ζεύξιππον γεγραμμένοις ὑπερβάλλοντί τινες οὐ μόνον τὸν ἀριθμὸν τοῦ ψάμμου τοῦ μέγεθος ἔχοντος ἴσον τᾷ γᾷ πεπληρωμένᾳ, καθάπερ εἴπαμες, ἀλλὰ καὶ τὸν τοῦ μέγεθος ἴσον ἔχοντος τῷ κόσμῳ.

Figura 15.5: Risultato della composizione con XeLaTeX

In particolare bisogna assicurare che xelatex conosca l'encoding *T1* e che sappia cercare correttamente lo stesso file di descrizione dei font che userebbe pdflatex. Per far questo bisogna specificare attentamente nella giusta successione i comandi seguenti:

```
\usepackage[T1]{fontenc}
\usepackage{fontspec}
\renewcommand{\sfdefault}{pag}
\usepackage{etoolbox}
\cspreto{sffamily }{\fontencoding{T1}}
```

dove

```
\usepackage[T1]{fontenc}
...
\renewcommand{\sfdefault}{pag}
```

sono gli stessi comandi che si darebbero a pdflatex, ma se li si usasse, l'encoding di default sarebbe l'encoding *T1*; se non si specificasse la chiamata a *fontspec*, non si potrebbero usare i font OpenType; per cui è necessario chiamare questo pacchetto, ma a questo punto l'encoding di default diventerebbe quello dei font OpenType, cioè la codifica UNICODE *TU*. Perciò bisogna caricare il pacchetto *etoolbox* che mette a disposizione l'ultimo comando `\cspreto`; questo permette di associare l'encoding *T1* al font specificato solo per la famiglia senza grazie tramite comando `\sfdefault`.

Sembra complicato, e forse la spiegazione lo è davvero, ma in fondo si tratta solo di aggiungere due righe, nell'ordine giusto, a quelle che andrebbero comunque specificate per usare sia i font OpenType in generale sia i font Type 1 particolari che si desiderano.

15.8 Conclusione

Sulla composizione di testi letterari e filologici ci sarebbe da scrivere molto di più di quanto si è esposto in questo capitolo tutto sommato introduttivo nel vastissimo territorio della composizione di questo tipo di documenti.

Si sono indicati pacchetti e programmi di composizione che possono aiutare molto l'umanista nel procedere lungo questo particolare sentiero di composizione tipografica. I pochi esempi mostrati indicano moltissime possibilità di azione. Sta dunque all'umanista, che tanto cura i suoi studi, curare anche l'aspetto compositivo dei suoi lavori. I programmi pdflatex, xelatex e lualatex gli offrono degli strumenti potentissimi di composizione; gli resta quindi il compito di usarli a piena potenza.

Capitolo 16

LaTeX: le presentazioni

16.1 Introduzione

Non è possibile tralasciare, nemmeno in una guida introduttiva, l'argomento delle presentazioni.

Per 'presentazioni' si intendono quelle proiezioni che accompagnano una conferenza, un discorso pubblico, un seminario, una lezione, eccetera.

Se il contenuto delle schermate da proiettare è prevalentemente o completamente testuale, al massimo include delle figure, vanno benissimo i programmi liberi, oppure commerciali, i quali spesso producono proiezioni molto accattivanti, ricche di effetti speciali e piene di parti colorate; spessissimo anche lo sfondo della schermata contiene delle parti disegnate, o delle fotografie, o il logo dell'istituzione di appartenenza dell'oratore; perché non disturbino la lettura queste immagini di sfondo dovrebbero essere molto chiare o comunque alterare il meno possibile il colore dominante dello sfondo.

Tuttavia preparare delle belle presentazioni è un'arte; se poi la presentazione deve contenere dello scritto tecnico (formule matematiche, schemi filologici, o altre cose non facenti parte della prosa 'normale') allora è opportuno usare LaTeX, il che comporta la necessità di saper usare LaTeX al meglio delle sue potenzialità; bisogna conoscere l'arte di comporre con LaTeX oltre che l'arte di predisporre delle schermate efficaci per il trasferimento sintetico dell'informazione dall'oratore all'ascoltatore.

16.2 Le classi per le presentazioni

LaTeX nasce con una semplicissima classe per le presentazioni, `slides`; nonostante la sua semplicità, estensibile facilmente con i vari pacchetti di estensione descritti anche negli altri capitoli, essa contiene alcuni elementi importanti da tenere presenti.

1. Poiché l'ascoltatore recepisce e ricorda meglio quello che legge, che sente, e che vede sullo schermo, è importante che ogni slide, o schermata, contenga un solo concetto e che l'ascoltatore non sia sottoposto a più di un nuovo concetto al minuto.
   In realtà un concetto può essere svolto anche su due o tre slide, ma è preferibile essere sintetici e non è il caso di sommergere l'ascoltatore sotto una valanga di slide, ognuna necessariamente proiettata per poco tempo. Per questo scopo è quindi necessario essere molto sintetici.
2. I due elementi illustrati sopra implicano che ogni slide non deve contenere più di una dozzina di righe di testo (e 12 forse sono già tante) contando anche le righe occupate da espressioni matematiche o da testi in alfabeti speciali o contenenti segni speciali; ognuna di queste righe va forse contata per due!
3. Perciò se la slide viene stampata su fogli di metacrilato (i *lucidi*, chiamati anche impropriamente 'trasparenze') allora i font 'normali' devono essere oltre che facilmente leggibili anche di corpo molto grande. La classe `slides` usa di default per il corpo normale il corpo da 20 pt (per l'esattezza 19,907 pt) della famiglia senza grazie; per migliorare la leggibilità questa famiglia in realtà è una famiglia speciale con l'altezza delle minuscole maggiore del solito: abcdefg; l'esempio ha lo stesso corpo usato nel resto del testo, ma come si vede le sue minuscole sembrano decisamente più grandi; in realtà è più grande la *x-height* ma sono più corti gli ascendenti.
4. Tenendo conto che lo schermo di proiezione ha solitamente il rapporto 4 : 3 fra base ed altezza, anche il foglio di metacrilato o la slide viene stampata facendo riferimento alla base come il lato lungo del foglio e bisogna usare nella dichiarazione di classe l'opzione *landscape*; in questo modo aumenta un poco la giustezza, ma diminuisce l'altezza e il numero di righe disponibile rimane necessariamente basso; non è possibile, perciò mettere troppa informazione in ciascuna slide.
5. La classe `slides` non consente di eseguire animazioni, specialmente se si fa uso dei lucidi di metacrilato; altre classi lo consentono se e solo se non si usano i lucidi, ma si proietta direttamente dal PC o dal laptop mediante un videoproiettore.
   Però con l'uso del pacchetto `color` è possibile colorare lo sfondo e scegliere colori diversi di ogni parte della slide composta con LaTeX, sia testo, sia matematica, sia disegni eseguiti con l'ambiente `picture` o altri ambienti nativi del sistema TeX.

Sia chiaro: oggi è difficilissimo trovare dei proiettori di slide in metacrilato. Oggi la classe `slides` è diventata obsoleta, ma i suoi font non lo sono affatto.

16.3 Altre classi per le presentazioni

Ogni distribuzione del sistema TeX contiene diverse classi per predisporre ottime presentazioni; si possono citare i pacchetti `prosper` e il suo successore `powerdot`,

seminar, *pdfscreen*, *pdfslide*, *ppower4*, *texpower*, per finire con *beamer*.

Ognuno di questi pacchetti produce slide bellissime e le presentazioni di dimostrazione, che accompagnano ogni pacchetto fra la sua documentazione, sono tutte accattivanti.

Ogni pacchetto offre una o più classi per la predisposizione di vari tipi di presentazione; ognuno finisce poi per affezionarsi ad un particolare pacchetto, ma le possibilità di personalizzazione sono tali e tante che non si corre il rischio di eseguire presentazioni con lo stesso stile standard.

16.4 La classe *beamer*

In inglese *beamer* significa videoproiettore. Il pacchetto *beamer* mette a disposizione del compositore sia la classe *beamer* sia una serie di file di estensione modulari con i quali è possibile personalizzare le proprie presentazioni e le modalità per attribuire valori diversi ai diversi elementi che caratterizzano sia il layout delle slide, sia i tipi di comandi o di segni particolari che appaiono nelle slide.

Queste sono concepite per file PDF interattivi, per cui ogni slide contiene, disegnati in colore molto tenue, tanto da non disturbare la lettura, anche i 'pulsanti' per muoversi nella sequenza di slide che costituiscono la presentazione. Inoltre è possibile avere a disposizione una specie di indice generale della presentazione sempre visibile, così che gli ascoltatori possano seguire il cambiamento di colore del titoli delle 'sezioni' che costituiscono la presentazione; questi stessi titoli costituiscono degli *hyperlink* utili all'oratore per navigare rapidamente attraverso la propria presentazione senza avere la necessità di far scorrere la presentazione fino a trovare la slide giusta.

La classe prevede l'uso dei pacchetti *pgf* e *xcolor* così che si possano sfruttare le molteplici funzionalità del pacchetto *pgf* per quel che riguarda la possibilità di eseguire dei bellissimi disegni o per inserire le funzionalità di quel pacchetto per l'inserimento delle immagini esterne.

Invece il pacchetto *xcolor* consente di gestire molto semplicemente i colori per ottenere sia colori misti, sia gradienti di colore lineari o sferici, il tutto con comandi di alto livello molto semplici per l'utente.

Le slide possono avere il loro contenuto esposto incrementalmente; vale a dire che, per esempio, la dimostrazione di un teorema può essere fatta esponendo una dopo l'altra le frasi che ne costituiscono la sequenza logica, senza che il testo già esposto cambi posizione via via che nuovo testo viene esposto. Uno dei vantaggi per l'oratore è che quanto verrà esposto in slide successive appare scritto in colore chiarissimo anche sulle slide precedenti, così che l'ascoltatore quasi non se ne accorge, ma l'oratore ne trae un notevole aiuto perché sa già in anticipo, vedendola, quale sarà la slide successiva.

Ovviamente la classe *beamer* consente di eseguire delle transizioni fra una slide e l'altra agevolando l'ascoltatore nel capire sia i collegamenti fra una slide e la successiva, sia, con transizioni diverse, nel capire quando si passa da una sezione all'altra della presentazione.

La predisposizione delle slide è piuttosto semplice: ognuna corrisponde ad un ambiente *frame*; ogni frame accetta un titolo e il testo della slide; se la slide deve essere presentata in fasi successive, il file di uscita conterrà diverse 'pagine', ma il frame resta invariato; semplicemente i comandi per l'esposizione incrementale delle varie parti della slide fanno parte, nel file sorgente, del testo contenuto nel frame.

Non è possibile in un testo stampato presentare questi effetti dinamici; ma si invita il lettore ad esaminare con attenzione uno dei file 'demo' che accompagna il pacchetto *beamer*; si trovano tutti in `.../doc/latex/beamer/examples/`; suggerisco `beamerexample5.pdf`, ma certi file sono solo per presentare alcuni dettagli, altri sono per presentazioni complete. Un altro esempio è formato dalla coppia di file `beamerexample2.article.pdf` e `beamerexample2.beamer.pdf` derivati dal medesimo file sorgente; a seconda di quali parametri vengono stabiliti nel preambolo, dallo stesso sorgente si possono ricavare sia la presentazione sia un articolo che può costituire direttamente il materiale 'cartaceo' da distribuire all'uditorio.

16.5 La documentazione

Tutti i pacchetti nominati dispongono della loro propria documentazione sul percorso `.../doc/latex/`⟨*pacchetto*⟩`/`. Il file di documentazione di *beamer* si chiama `beameruserguide.pdf` ed è particolarmente interessante, non solo perché contiene una documentazione molto ben fatta, tenendo conto anche della versatilità del pacchetto, e quindi della moltitudine di comandi e ambienti che esso fornisce, ma anche per le preziose informazioni sulla preparazione di presentazioni professionali; questo genere di informazioni non è facilmente reperibile in altra documentazione o in altri libri.

La sostanza è che quasi tutto quello che è opportuno fare da un punto di vista tecnico-compositivo è disponibile con LaTeX e con i comandi messi a disposizione dai pacchetti di servizio *pgf* e *xcolor*, ma quello che deriva dalla professionalità e dal buon gusto non può e non è solitamente descritto da nessuna parte, tranne, appunto, nel manuale di *beamer*.

È sorprendente notare come la maggior parte delle presentazioni composte con noti programmi dedicati, sia commerciali sia freeware, pur essendo quei programmi del tutto validi per lo scopo, peccano perché gli autori non conoscevano e non avevano nessun modo di ricavare dalla documentazione le regole di stile e di buon gusto per la predisposizione di slide efficaci; queste sono efficaci se sono prive di quegli elementi 'decorativi' che, invece di favorire la trasmissione della comunicazione, distraggono lo spettatore/ascoltatore, il quale alla fine della conferenza o della lezione si ricorda di queste animazioni ma non si ricorda di quello che ha detto l'oratore.

È quindi assai raccomandabile leggersi attentamente il manuale di *beamer*; si impareranno moltissime cose che permetteranno di usare al meglio lo strumento

compositivo per ottenere lo scopo di una presentazione, cioè quello di trasmettere un messaggio all'ascoltatore nel modo più efficace possibile.

16.6 Una breve presentazione

Nella figura 16.1 sono rappresentate 8 slide corrispondenti a 5 frame di una brevissima presentazione. Esse sono disposte in verticale; la prima colonna contiene le slide da 1 a 4 e la seconda colonna da 5 a 8.

Il codice per produrre queste slide è il seguente e serve per interpretare il risultato ottenuto nelle slide rappresentate nella figura 16.1.

```
% !TEX encoding = UTF-8 Unicode
%%% file per produrre alcune slide per la GuidaGuit
\documentclass{beamer}
%                                    Preambolo
\usetheme{AnnArbor}
\useoutertheme[right]{sidebar}
\setbeamercolor{alerted text}{fg=red!90!black}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage{pgf}
\usepackage{curve2e}
\usepackage[color]{guit}
\usepackage[italian]{babel}

\setbeamercovered{transparent}

\usefonttheme{professionalfonts}
\usepackage{lxfonts}

\title{I numeri complessi}
\subtitle{Conferenza internazionale
del~Gruppo~Italiano~degli~utenti~di~\TeX}
\author{Mario Rossi}
\institute{\GuIT}

\date{Pisa, 20--22 ottobre 2015}


\pgfdeclareimage[width=\textwidth]{CifreIndiane}%
                               {CifreIndianeVIIsecolo}
\pgfdeclareimage[width=15.5mm]{guitlogo}{GuITlogo}

\logo{\pgfuseimage{guitlogo}}
% Fine del preambolo
```

Figura 16.1: Otto slide per una presentazione di cinque frame. Vale la pena di osservare che i comandi usati per il diagramma dell'ultima slide sono stati eseguiti con i comandi definiti nel pacchetto *curve2e*.

```
\begin{document}% Inizio della presentazione


\begin{frame}% Primo frame, prima slide
\titlepage
\end{frame}

\begin{frame}% Secondo frame, seconda slide
\frametitle{Indice}
\tableofcontents
\end{frame}

\section{Introduzione storica}

\begin{frame}% Terzo frame, terza slide
\frametitle{Una breve storia dei numeri}
All'inizio del \textsc{vii} secolo gli Indiani inventarono
la notazione posizionale e le <<nove>> cifre dall'1 al 9;
lo zero veniva detto a voce ma non aveva ancora un suo segno.

\begin{center}
\pgfuseimage{CifreIndiane}
\end{center}
\end{frame}

\section{Nascita dei numeri complessi}


\begin{frame}% Quarto frame,  diviso in quattro slide
\frametitle{I numeri complessi nascono nel '500}
\begin{enumerate}
\item<1-> Nel XVI secolo Tartaglia e Cardano introducono
la radice quadrata di $-1$
\item<2-> Il nome di \emph{unità immaginaria} viene creato
da Réné Descartes nel 1637
\item<3-> Gauss nel 1799 contribuisce con i suoi scritti
a diffondere i numeri complessi
\item<4-> Hamilton nel 1833 pubblica la teoria dei numeri
complessi
\end{enumerate}
\end{frame}


\section{I numeri complessi in \texttt{pict2e}}
```

```
\begin{frame}% Quinto frame, ottava slide
\frametitle{I numeri complessi come operatori geometrici}
\begin{center}\unitlength=1mm
   \begin{picture}(60,40)
   \put(0,0){\vector(1,0){60}}\put(60,1){\makebox(0,0)[br]{$x$}}
   \put(0,0){\vector(0,1){45}}\put(1,45){\makebox(0,0)[lt]{$y$}}
   \thicklines
   \put(0,0){\textcolor{red}{\vector(3.4641,2){40}}}
   \thinlines
   \multiput(40,-.5)(0,2){12}{\line(0,1){1}}
   \multiput(-.5,23.094)(2,0){20}{\line(1,0){1}}
   \put(41,1){\makebox(0,0)[bl]{$a$}}
   \put(1,24.1){\makebox(0,0)[bl]{$b$}}
   \VectorArc(0,0)(20,0){30}
   \put(20,4){\makebox(0,0)[bl]{$\varphi$}}
   \put(20,12.6){\rotatebox{30}{\makebox(0,0)[b]{$m$}}}
   \put(0,0){\thicklines\textcolor{blue}{\vector(1,0){10}}}
   \put(5,-1){\textcolor{blue}{\makebox(0,0)[t]{vettore
              unitario}}}
   \end{picture}
\end{center}
Visto come operatore geometrico, il numero complesso
$m\mathsf{e}^{\mathsf{i}\varphi}=a+\mathsf{i}b$ agisce su
un vettore; in questa figura agisce sul
\textcolor{blue}{vettore unitario blu}; lo scala tramite
il fattore \alert{$m$} e lo ruota dell'angolo \alert{$\varphi$}
producendo il \alert{vettore rosso}.
\end{frame}

\end{document}
```

Nel preambolo appaiono diversi comandi, alcuni dei quali sono già familiari, per invocare *babel* con l'opzione *italian* al fine di comporre in italiano; per invocare *inputenc* con l'opzione *utf8* per scrivere usando le lettere accentate; per invocare *fontenc* con l'opzione *T1* per scrivere nel file di uscita con le lettere accentate direttamente, non ottenute per sovrapposizione del segno dell'accento sopra il segno della vocale. Si invoca anche il pacchetto *pgf* per poter usufruire della grafica avanzata, e per la gestione delle figure; il pacchetto *guit* per gestire i loghi del Gruppo degli utilizzatori Italiani di TeX; il pacchetto *curve2e*, che a sua volta carica *pict2e*, per il disegno in uno dei frame della presentazione.

Ma, ai fini di questo esempio, sono interessanti gli altri comandi:

`\usetheme` serve per scegliere il layout generale dei frame, sia come colori, sia come disposizione delle informazioni che corredano la presentazione; in questo caso si è scelto lo stile `AnnArbor` che è uno dei vari stili messi a

disposizione dell'utente di **beamer**; ovviamente questi sono predefiniti, ma l'utente può crearsi dei temi a suo piacimento partendo da zero, oppure può copiare e modificare quelli esistenti mediante gli altri comandi che si vedranno qui di seguito.

`\useoutertheme` serve per scegliere le parti che contornano i frame ai fianchi; in questo caso si è scelto di usare la barra laterale (una striscia con un colore di fondo nella quale appariranno certe informazioni) e si è scelto di mantenerla a destra.

`\setbeamercolor` serve per impostare uno dei colori che beamer usa per certi tipi di testo; in questo caso il colore dell'*allerted text*, il testo su cui richiamare l'attenzione, viene impostato con il 90% di rosso e il restante (10%) nero; insomma si preferisce un rosso un po' più scuro del normale.

`\setbeamercovered` è una dichiarazione che accetta diverse impostazioni con la sintassi:

```
\setbeamercovered{⟨impostazioni⟩}
```

dove ⟨*impostazioni*⟩ può valere: `transparent=`⟨*valore*⟩, `invisible`, `dynamic`, `highly dynamic`, `still covered=`⟨*lista*⟩, `again covered=`⟨*lista*⟩; gli argomenti sono essenzialmente delle impostazioni booleane il cui significato è piuttosto chiaro; ma quelli che accettano una ⟨*lista*⟩ sono piuttosto delle opzioni del tipo ⟨*opzione*⟩=⟨*valore*⟩; che cosa significhi ⟨*lista*⟩ è bene approfondirlo nella documentazione di **beamer**. Il ⟨*valore*⟩ di trasparenza è prefissato a `0.85`, ma l'utente può esplicitamente specificarne un altro. Con l'impostazione `transparent` si chiede che il testo da esporre in slide successive sia 'trasparente', molto chiaro in modo da intravederlo anche quando il testo non è esposto.

`\title` `\author` e `\date` sono vecchie conoscenze, ma **beamer** consente di specificare anche un sottotitolo per la presentazione mediante il comando `\subtitle`. Il modo particolare con cui è scritto il sottotitolo serve per garantire che la frase sia divisa in due sottofrasi in modo che ognuna contenga le sue congiunzioni e preposizioni semplici o articolate.

`\institute` serve per inserire il nome o il logo dell'istituzione a cui appartiene l'oratore.

`\pgfdeclareimage` è un comando che consente di assegnare non solo certe caratteristiche all'immagine da usare, in questo caso la larghezza, ma anche un nome con cui richiamarle insieme alle loro proprietà. Attenzione, non si tratta di una cosa da poco; l'immagine `GuITlogo` viene riprodotta in ogni slide, quindi se ogni volta si ricaricasse il file del disegno, si correrebbe il rischio di finire con file dalle dimensioni enormemente grandi; usando invece `\pgfdeclareimage` il particolare file che contiene l'immagine viene caricato una sola volta grazie al fatto che il programma pdflatex è in grado di riutilizzare quell'immagine un numero illimitato di volte senza copiarla ma facendovi riferimento mediante i suoi meccanismi interni di gestione degli hyperlink. Questa funzionalità è disponibile anche se si usa lualatex o

xelatex; salvo casi particolari dove sia necessario usare font OpenType, si suggerisce di usare ogni volta che si può il compilatore pdflatex.

`\logo` dice quali comandi devono venire usati per produrre l'immagine del logo nel punto in cui questo deve apparire.

`\pgfuseimage` infine è il comando che, riferendosi al nome di un'immagine precedentemente dichiarato mediante `\pgfdeclareimage`, effettivamente crea gli hyperlink interni che rendono visibile l'immagine nel punto voluto.

`\usefonttheme` con l'argomento `professionalfonts` informa *beamer* che si intende usare una collezione di font completa, vale a dire che non contiene solo i font testuali, ma anche i corrispondenti font matematici; chi scrive usa sempre i font della collezione LXfonts che, appunto, sono quelli usati in questi esempi e sono completi dei corrispondenti font matematici. Questi font sono una ricostruzione estesa dei font usati dalla classe *slides*, particolarmente leggibili e privati di alcune lettere che potevano essere confuse con altre[1]. Con il comando `\usepackage{lxslides}` si ridefiniscono tutte le impostazioni per i font testuali e matematici, permettendo l'uso anche dei caratteri greci sia nel testo sia in matematica.

Con l'inizio del documento appaiono le dichiarazioni dei vari frame mediante i successivi ambienti *frame*. Il primo frame contiene il titolo della presentazione; il comando `\titlepage` fa sì che il frame sia composto in modo un po' differente dagli altri, ma che contenga buona parte delle informazioni relative al titolo, al presentatore, e alla sua istituzione di appartenenza, all'evento nell'ambito del quale si svolge la presentazione, eccetera; la prima slide e primo frame della figura 16.1 mostra tutti questi dettagli. Nell'angolo in basso a destra compare anche il numero progressivo del frame riferito al numero complessivo dei frame che compongono la presentazione; nell'esempio compare la frazione 1/5 per dire che si tratta del primo di cinque frame.

Il secondo frame contiene l'indice della presentazione; è utile che questo indice venga presentato all'inizio, perché è un semplice mezzo per consentire all'oratore di delineare all'uditorio il piano della sua presentazione. Questo stesso elenco è collegato internamente agli hyperlink che consentono di navigare all'interno della presentazione; le stesse voci di questo indice compaiono anche nella barra laterale destra, e si illuminano di un colore più chiaro ogni volta che si cominciano ad esporre le varie slide che compongono una data sezione; gli ascoltatori seguendo questa barra possono rendersi conto istante per istante del punto a cui è arrivato l'oratore, senza bisogno di controllare l'orologio con mosse ampie e vistose per farsi notare dall'oratore affinché ci dia un taglio... Spero che nessuno dei lettori di queste note si trovi mai a fare o a subire queste azioni!

Tra il secondo frame e il terzo è inserito un comando di sezionamento; il suo contenuto consente di creare l'indice, ma non compare mai al di fuori dell'indice; è bene che una presentazione sia strutturata in sezioni, eventualmente anche in sottosezioni, ma non si frammenti troppo l'esposizione; durante un

[1] I caratteri che potevano ingenerare confusione sono stati ridisegnati in modo da evitare questo inconveniente; quindi non si tratta degli *slide font* originali, ma di loro derivati.

congresso è possibile che ogni oratore abbia a sua disposizione 20 minuti di cui cinque sono destinati a rspondere alle domande poste dagli ascoltatori; quindi la presentazione non dovrebbe contenere più di 15–20 slide; due o tre sezioni sono più che sufficienti per strutturare la presentazione. Per una lezione, magari di due ore, è evidente che si può strutturare maggiormente, ma anche in questa circostanza non è il caso di fare un unico file con un indice interminabile; gli allievi si demoralizzerebbero, specialmente tenendo conto che durante una lezione assistita da una presentazione, quanto viene detto è molto di più di ciò che si potrebbe esporre con le sole parole o con l'aiuto di una lavagna su cui scrivere o disegnare. Una simile lunga lezione è meglio che sia strutturata in due parti (due distinti file) da presentare con un congruo intervallo fra le due.

Si vede che nel terzo frame è richiamata per nome una delle due immagini precedentemente dichiarate e si fa uso di `\pgfuseimage` che svolge il ruolo che normalmente avrebbe il comando `\includegraphics` del pacchetto *graphicx*.

Il quarto frame corrisponde a 4 slide; si vede che esso contiene una enumerazione, ma dopo ogni comando `\item` compare una strana espressione: due parentesi 'acute' racchiudono un intervallo nel quale sono indicate le slide da proiettare; il primo termine dell'elencazione deve essere proiettato dalla slide 1 in poi; il secondo elemento dalla slide 2 in poi, eccetera; la sintassi di questo intervallo di numerazione delle slide segue questa regola: se le parentesi acute contengono un solo numero, senza il trattino, la slide viene proiettata solo quando tocca a lei (in base al numero d'ordine); se invece l'espressione contiene un intervallo del tipo `2-3` il termine dell'elencazione appare solo per le slide da 2 a 3 comprese, ma non compare mentre viene proiettata la slide numero 1, né quando viene proiettata la numero 4. Se manca un numero prima del trattino, si intende il numero 1; se manca un numero dopo il trattino, si intende fino all'ultima slide di questo frame. Il fatto, però, è che le slide dello stesso frame hanno tutte lo stesso contenuto; ma quelle che devono mostrare solo una parte del contenuto, ce l'hanno in colore trasparente molto chiaro, quasi invisibile per chi non sa che cosa c'è scritto, ma non per l'oratore che invece intravede quanto apparirà nella slide successiva.

Infine nel quinto frame c'è un disegno eseguito con i comandi dell'ambiente *picture*, come esteso dal pacchetto *curve2e*, che, lo si ripete, carica il pacchetto *pict2e*.

16.7 Creare un nuovo stile

Gli stili di presentazione nel gergo di *beamer* si chiamano *temi*, più precisamente con la parola inglese *theme*; la cartella di *beamer* contiene diverse sottocartelle per le configurazioni di diversi temi, separati in modo modulare, in modo che cambiare un tema, non vuol dire cambiare gli altri. Se un autore vuole costruirsi uno stile di presentazione proprio, lo può fare in modi diversi; il più semplice, forse, è quello di vedere sulla documentazione di *beamer* i vari temi già predisposti, scegliere quello che più si avvicina allo stile desiderato e poi modificarlo caricando

dopo quel tema, gli altri moduli tematici; uno per i colori, uno per le barre, uno per i campi da inserire nelle riga di testa e nella righe in calce, eccetera.

L'altra via è quella di leggere con attenzione la documentazione relativa ai temi e scrivere autonomamente le macro che occorrono; non è semplice, ma non è nemmeno troppo difficile.

Il modo più semplice è quello di ricorrere al sito: `http://titilog.free.fr/indexEng.htm`; in questo sito c'è solo una finestra con una slide di modello; si clicca sulle varie parti che si vogliono personalizzare scegliendo dal menù sul lato destro sia i colori dello sfondo, si quelli dei font da usare, eccetera. Alla fine si clicca il grosso bottone ovale contenente la scritta *Finish* e il gioco è fatto; sotto alla slide modello c'è il bottone di download, per scaricare direttamente il file `.sty` contenente il tema appena creato. Il nome di questo file deve cominciare con la locuzione `beamertheme` e deve proseguire con una parola o locuzione mnemonica, per esempio il nome del congresso durante il quale la presentazione viene esposta ai congressisti, per esempio `ecoc2010`; il file del tema avrà quindi il nome complessivo: `beamerthemeecoc2010.sty`. Cliccando sul bottone *Download* si esegue lo scaricamento del file sul proprio disco.

Per usare questo nuovo tema basta inserire nel preambolo

```
\usetheme{ecoc2010}
```

Ovviamente, dopo aver creato il nuovo tema nessuno vieta di eseguire delle altre modifiche agendo direttamente sul file, ma il grosso è già fatto in modo grafico interattivo, quindi il tema dovrebbe essere già a posto e usabile immediatamente.

16.8 Osservazioni

Questo capitolo serve per richiamare l'attenzione del lettore sull'opportunità di documentarsi a fondo nelle rispettive guide in merito alle potenzialità dei vari pacchetti citati. Alcuni, come *beamer*, sono nati per produrre il risultato finale direttamente in formato PDF, altri, come *prosper* e il suo discendente *powerdot*, ricorrono al pacchetto *PSTricks* e quindi devono subire due trasformazioni: prima devono essere composti con latex per ottenere il file DVI, poi questo deve essere trasformato in formato PDF attraverso il doppio passaggio dvips e ps2pdf[2]. Il gioco può valere la candela, come si dice, ma dipende dal tipo di disegni che è necessario inserire nelle slide.

È inevitabile che ognuno si affezioni ad una particolare classe, anche perché bisogna usare molti nuovi comandi dai nomi talvolta non così facili da ricordare; perciò l'abitudine diventa un fattore importante.

Però parlando di arte di comporre con LaTeX, non si può negare che LaTeX è uno strumento formidabile, ma da solo non è sufficiente per predisporre presentazioni efficaci e accattivanti. La necessità di documentarsi nell'arte della comunicazione orale e visiva diventa essenziale; padroneggiando bene quest'arte diventa poi abbastanza facile usare lo strumento LaTeX con i suoi pacchetti di estensione in modo da ottenere il meglio.

[2]Il programma dvipdfm non è in grado di interpretare il codice PostScript complesso generato dai comandi di *PSTricks*.

Capitolo 17

LaTeX: la microgiustificazione

17.1 Introduzione

Quando si vogliono fare bene le cose, meglio ancora di come le farebbe LaTeX da solo, che rasenta già la perfezione in fatto di composizione tipografica, bisogna ricorrere alle potenzialità di LaTeX, mediante i compilatori pdflatex e lualatex, che non sono documentate né nel libro/manuale di Leslie Lamport, né in molti altri autorevoli libri, ma sono riportate solo nella documentazione che accompagna il pacchetto *microtype*. Il programma xelatex per ora può sfruttare la microgiustificazione in modo parziale, come d'altra parte succede anche con latex; questo è dovuto al passaggio intermedio per il formato DVI (semplice o esteso).

Si tratta di questo: la composizione tipografica di qualità richiede tra l'altro che il testo composto appaia privo di 'ruscelli' e sia di un 'colore' uniforme. I *ruscelli* tipografici sono quelle sequenze di spazi interparola che si susseguono in diverse righe successive in corrispondenza quasi perfetta gli uni con gli altri, così che essi formano una striscia bianca che serpeggia attraverso il testo come un ruscelletto bianco.

Non solo ma gli spazi interparola dovrebbero essere il più omogenei possibile e non troppo grandi, in modo che il testo, visto da una distanza alla quale non si riesca a leggere quanto esso contiene, appaia di un colore (grigio) uniforme.

Gli artisti tipografi sanno che lo spazio interparola ideale dovrebbe essere quello di una lettera 'i' nel font nel quale è composto il testo. Knuth, invece, nel suo TeXbook afferma che lo spazio interparola ideale sia largo quanto una 'e' nel font corrente. Probabilmente non esiste un valore di spazio da prescrivere come ideale, perché la spaziatura piccola della 'i' può andare bene per un testo letterario, quello di una 'e' probabilmente è un buon compromesso in generale, ma potrebbero esserci dei testi specifici, specialmente di carattere scientifico, dove la spaziatura interparola diventa importante per separare bene il testo,

dalla matematica, dalle parole tecniche, eccetera. Quello che è certo è che la composizione con noti word processor e la composizione in colonne strette praticata nelle tipografie di molti quotidiani e molti settimanali rinuncia a questa omogeneità degli spazi interparola producendo risultati 'terribili' nonostante i mezzi tecnici a disposizione del compositore.

A questi problemi pone rimedio il pacchetto *microtype* che offre soluzioni modeste a latex e xelatex, ma che sviluppa tutta la sua forza con pdflatex e lualatex.

17.2 La composizione dei capoversi

Il TeXbook spiega dettagliatamente come l'interprete pdftex componga il testo sorgente in capoversi giustificati. I dettagli sono piuttosto complicati, ma sotto sotto c'è tutta l'abilità del matematico Knuth che ha prodotto un algoritmo di ottimizzazione capace di dividere il capoverso in righe giustificate minimizzando la somma totale dei *demerits* associati alle righe in cui sarebbe possibile dividere un capoverso; questi *demerits* sono legati quadraticamente alla *bruttezza* della divisione scelta fra le divisioni possibili oltre che ad altri elementi costituiti generalmente da *penalità*.

La *bruttezza* è un indice definito a partire dall'ammontare dell'allargamento o del restringimento dello spazio interparola nelle righe giustificate dell'intero capoverso. Le penalità sono presenti esplicitamente, per averle specificate nel testo sorgente, o anche indirettamente, perché sono state introdotte dai comandi usati; alcune penalità derivano dall'aver inserito divisioni di parole in fin di riga. Le *penalità* da usare nei diversi contesti sono predefinite nel file di formato e in quelli di classe, ma l'utente può inserirne di altre attraverso comandi come `\break`, `\nobreak`, `\linebreak`, e simili, oppure inserendo alcune cesure esplicite mediante il comando `\-`.

Altri *demerits* sono ulteriormente introdotti dal programma di composizione se la penultima riga di un capoverso riceve la cesura. Il pacchetto *babel*, per la lingua italiana, impone il valore di 50 000 000 al parametro `\finalhyphendemerits` che controlla, appunto, la cesura della penultima riga di un capoverso. In italiano, infatti sarebbe quanto mai indesiderabile che venga divisa la penultima riga di un capoverso, consentendo che nell'ultima riga ci possa essere solo un frammento di parola, al limite due sole lettere, corrispondenti alla minima lunghezza di una sillaba terminale secondo l'algoritmo di sillabazione di LaTeX per l'italiano. Quell'enorme valore di *demerits* non è 'infinito', ma consente praticamente sempre che l'ultimo righino di un capoverso cominci con una parola intera.

Tutte le tabelle dei font 18.3–18.12 contengono anche i parametri metrici caratteristici dell'intera polizza, in particolare l'ammontare dello spazio interparola e il suo allungamento e il suo accorciamento; per esempio nella tabella 18.5, relativa al font a 10 pt usato per comporre questo testo, si legge che lo spazio interparola è pari a 3,3333 pt, che l'allungamento vale 1,6666 pt e che l'accorciamento vale 1,1111 pt. Questo significa che nel giustificare le righe TeX aggiusta gli

spazi interparola aggiungendo o togliendo allo spazio 'normale', pari a un terzo del corpo, quanto consentito dall'allungamento o dall'accorciamento previsti, aggiungendo alla bruttezza una quantità dipendente dal cubo dell'ammontare di questi allungamenti o accorciamenti relativi rispetto ai valori naturali[1].

Quando si inserisce il comando di legatura fra due parole o due informazioni tra le quali si vieta la divisione della riga, quando, insomma, si usa il carattere attivo `~`, per ottenere questo effetto, il comando inserisce una penalità pari a 10 000 (che per TEX è sinonimo di 'infinito') per cui una eventuale divisione in quel punto provocherebbe una bruttezza infinita, al punto che qualunque altra divisione produrrebbe una bruttezza inferiore.

Le cesure in fin di riga vengono minimizzate ricorrendo a due distinte composizioni; il programma esegue una prima divisione delle righe giustificate senza dividere in sillabe e calcola la (minima) bruttezza del capoverso così ottenuto. Se questo valore è inferiore a quello conservato in `\pretolerance`, allora questa divisione viene accettata e le righe giustificate del capoverso vengono accodate alla pagina parzialmente composta. Se questo valore di `\pretolerance` viene superato, allora il programma esegue una seconda divisione delle righe consentendo la cesura in fin di riga; esso però attribuisce una penalità ad ogni cesura introdotta, una penalità ad ogni sequenza di due righe consecutive contemporaneamente contenenti la cesura, e una ulteriore dose di *demerits* se la penultima riga del capoverso contiene la cesura. Tutte queste penalità contribuiscono ad incrementare i *demerits* della divisione in righe; fra tutte le possibili combinazioni di divisioni, con le corrispettive cesure e i corrispettivi allungamenti o accorciamenti degli spazi interparola, il programma sceglie quella che produce la minima bruttezza complessiva e la confronta con un altro valore conservato in `\tolerance`. Se questa minima bruttezza non supera `\tolerance`, la divisione in righe viene accettata, altrimenti TEX ci prova una terza volta ma il risultato finale viene accodato alla pagina in composizione accompagnato dai messaggi relativi alla bruttezza e alla presenza di righe 'overfull' o 'underfull'; le prime sporgenti fuori della giustezza, e le seconde con un allargamento eccessivo degli spazi interparola.

Questo fatto accade molto raramente; nonostante tutto può accadere. Se l'eccesso di lunghezza non supera 1 pt–1,5 pt, spesso si può chiudere un occhio, ma se si può, è meglio provvedere. Se lo spazio interparola viene allargato troppo ma la bruttezza della riga non supera qualche migliaio (1000–2000) si può chiudere un occhio, ma di nuovo se si può rimediare, è meglio provvedere. Generalmente questo implica la modifica del testo. In linea di massima, senza ricorrere alla microgiustificazione, spetta al compositore, d'accordo con l'autore, il compito di modificare il testo del capoverso affinché la sua bruttezza sia ridotta; certamente il programma è molto potente, ma non arriva a modificare il testo da solo!

[1] La funzione di bruttezza legata a questi allungamenti o accorciamenti relativi è sempre cubica, ma è diversa per l'allungamento, che può teoricamente assumere qualunque valore, rispetto all'accorciamento che non può essere superiore al valore 'normale', altrimenti una parola si sovrapporrebbe alla precedente.

17.3 Metodi per migliorare la giustificazione

Dai paragrafi precedenti si evince chiaramente che latex 'normale' può ricorrere solamente all'aggiustamento dello spazio interparola e alla cesura in fin di riga per comporre i capoversi.

I tipografi che lavoravano a mano con i caratteri mobili avevano anche altre tecniche per giustificare meglio i capoversi.

spazieggiatura In inglese *tracking*. Spazieggiare i caratteri significa comporli con uno spazio interlettera leggermente maggiore di quello di default. Per i più autorevoli *book designer* britannici spazieggiare è paragonato all'abigeato (furto di bestiame). Per noi il paragone è quasi incomprensibile, ma la sua forza risiede nel fatto che il furto di bestiame era un reato molto grave nella Gran Bretagna del XIX secolo, tanto che veniva punito con la pena di morte.

Oggi si conviene che la spazieggiatura sia accettabile e talvolta consigliabile nella composizione in lettere maiuscole o maiuscolette di titoli o di simili brevi tratti di testo; assolutamente improponibile in un testo in lettere minuscole. Il lettore attento provi ad esaminare una qualunque rivista pubblicata ai nostri giorni e cerchi di determinare quali righe siano spazieggiate; constaterà che la spazieggiatura non solo è visibile ad occhio nudo, ma anche che contribuisce non poco a rendere mal composto il capoverso che contenga tali righe spazieggiate. Egli constaterà anche che la raccomandazione di non spazieggiare il testo scritto in lettere minuscole è abbondantemente disattesa in quel genere di pubblicazioni; probabilmente una composizione giustificata solo a sinistra sarebbe molto più raccomandabile.

protrusione della punteggiatura La protrusione della punteggiatura è quella tecnica che consente di comporre con il margine destro leggermente seghettato, invece che perfettamente allineato, ma in modo che le parti sporgenti siano costituite solo da segni di punteggiatura, segnatamente il punto e la virgola, oltre che da una frazione del trattino di cesura. L'occhio, data la scarsa forza visiva di questi segni, vede il margine destro giustificato anche se in effetti esso non lo è esattamente.

protrusione dei segni grafici Con lo stesso criterio si possono far uscire piccole parti dei segni alfabetici, numerici, di punteggiatura, eccetera, fuori da entrambi i margini, purché questa protrusione ammonti ad una frazione piccolissima del segno che protrude. Questo implica che fra le caratteristiche metriche del font deve essere indicata per ciascun segno una coppia di valori numerici che rappresentano la parte millesimale della larghezza del singolo segno che può protrudere nel margine di sinistra o, rispettivamente, nel margine di destra. Spesso nei font con grazie ci si può accontentare di far sporgere nei margini piccole porzioni delle grazie orizzontali.

larghezza del font Si può ancora ricorrere a varianti del font corrente formate da font dello stesso corpo, forma e nerezza, ma leggerissimamente più 'estesi' o 'compressi'. Una variazione di larghezza del $\pm 2\%$ dovrebbe essere

considerata come la variazione massima otticamente accettabile affinché l'occhio del lettore non si accorga della variazione.

legature e segni diacritici Nel comporre si potrebbero usare font dotati di segni legati in numero maggiore dei soliti ff, fi, ffi, fl, ffl e ß presenti nei font delle tabelle 18.3 e 18.5. Gli antichi usavano molte altre legature, presenti oggi solo nei font esperti, e ricorrevano a segni diacritici, come per esempio l'uso della tilde sulla lettera precedente per marcare una 'm' o una 'n' omessa: non è raro trovare scritture come *librũ* al posto di *librum*. Oggigiorno questo genere di artifici non viene più usato in italiano. In tedesco, invece, la umlaut sopra certe vocali è il residuo di questa pratica e rappresenta una 'e'; infatti quando in tedesco bisogna scrivere con font che mancano della umlaut, si mette esplicitamente la 'e' a destra della vocale che non ha potuto ricevere la umlaut: *Universität* o *Universitaet*; *Müller* o *Mueller*.

17.4 La microgiustificazione

I prototipografi ricorrevano spesso alle tecniche illustrate sopra; Gutenberg stesso avrebbe usato varianti più larghe o più strette del font 'normale' per comporre una delle sue famose Bibbie. Gutenberg stesso non avrebbe usato allargamenti o restringimenti maggiori del 2%, tanto che sarebbero state necessarie misure molto accurate da parte degli studiosi di tipografia antica per rendersi conto del 'trucco' usato al fine di comporre così bene.

I metodi descritti nel paragrafo precedente possono essere usati anche con i programmi di composizione del sistema TEX.

Segnatamente la protrusione della punteggiatura può essere usata con latex e xelatex, mentre questa protrusione e le altre tecniche possono essere tutte usate con i programmi pdftex e lualatex.

La scelta, quindi, di usare la piena potenzialità dei meccanismi di microgiustificazione o soltanto la protrusione dei segni di interpunzione dipende da altri fattori; come si è detto, ma conviene ripeterlo, latex e xelatex sono limitati alla protrusione dei segni, mentre pdflatex e lualatex non hanno questa limitazione ma possono servirsi sia della protrusione sia dell'espansione dei font.

Detto ciò, la microgiustificazione si ottiene richiamando il pacchetto *microtype*. Questo pacchetto riconosce da solo con quale programma viene composto il particolare documento e si comporta conseguentemente per usare solo le funzionalità di cui il programma è capace.

Ovviamente il pacchetto *microtype* è configurabile mediante opzioni passate al comando con il quale lo si richiama; alcune funzionalità possono essere attivate o disattivate dal compositore a seconda di che cosa egli sta componendo.

Non solo, ma adeguatamente istruito con comandi da inserire nel preambolo o lungo il file sorgente, il pacchetto è in grado di cambiare impostazioni a seconda del contesto nel quale sta operando. Per esempio nelle note a piè di pagina le opzioni da usare potrebbero essere diverse da quelle che sono in azione durante la composizione del testo; per esempio, in francese, dove la spaziatura dei segni

di interpunzione è particolare, potrebbero essere in vigore le opzioni per lo spazio extra prima dei segni di interpunzione, oppure all'interno delle parti delimitate dalle virgolette caporali, mentre in italiano queste opzioni devono essere disabilitate; in un documento scritto in italiano ed in francese le spaziature extra prima dei segni di interpunzione apparirebbero solo nelle parti scritte in francese.

Le possibilità di gestire tutte le funzioni di microgiustificazione sono numerose; quindi si rinvia il lettore alla documentazione in `.../doc/latex/microtype/microtype.pdf`.

Questo testo è stato composto con l'uso del pacchetto *microtype*; con l'aiuto di un righello e una versione stampata di almeno alcune pagine del testo è possibile verificare la protrusione dei segni di interpunzione e la protrusione nei due margini di piccole parti di diversi caratteri. L'espansione o la contrazione dei caratteri è difficile da misurare; tuttavia è possibile che la stessa parola compaia in righe diverse con differenti livelli di compressione o di espansione; misurando accuratamente la larghezza di tali parole si può verificare la presenza di questo particolare tipo di microgiustificazione.

Si noterà anche che le parole con cesura in fin di riga sono proprio poche; generalmente si tratta di parole lunghe almeno otto caratteri, spesso decisamente più lunghe; gli spazi interparola non sono proprio uguali, ma sono molto più omogenei di quelli che si avrebbero senza l'uso del pacchetto.

Disponendo dei file sorgente di questo testo si può provare a vedere che cosa si ottiene senza usare il pacchetto *microtype*; si noteranno spazi interparola più disomogenei, molte più righe con cesure in fin di riga, e molti più messaggi di `overfull` oppure di `underfull hbox`.

Si osservino comunque nella pagina 387 le tre composizioni dello stesso testo a confronto; la prima è stata ottenuta componendo un capoverso sufficientemente lungo su due colonne e con la microgiustificazione attivata; la seconda composizione contiene le stesse parole, ma la microgiustificazione è disattivata; nella terza composizione, invece, la microgiustificazione è attiva solo per quel che riguarda la protrusione, mentre è disattivata l'opzione per ricorrere alle varianti del font con larghezze diverse. Si tenga presente che all'interno dell'ambiente *multicols*, con cui sono eseguite le tre composizioni, le penalità per la presenza di cesure in fin di riga sono drasticamente ridotte, proprio per tener conto della composizione in colonne strette.

Nel secondo testo, non microgiustificato, si osserva una riga in più, spazi interparola più grandi e si vedono chiaramente alcuni ruscelli, mentre non ne appaiono nel testo microgiustificato; il testo non microgiustificato contiene anche più cesure in fin di riga e diverse sequenze di righe consecutive terminanti con la cesura. Nel terzo testo si ha una situazione intermedia, ma è tutto quello che si può ottenere quando il documento è composto per l'uscita in formato DVI.

Oggi si conviene che la spazieggiatura sia accettabile e talvolta consigliabile nella composizione in lettere maiuscole o maiuscolette di titoli o di simili brevi tratti di testo; assolutamente improponibile in un testo in lettere minuscole. Il lettore attento provi ad esaminare una qualunque rivista pubblicata ai nostri giorni e cerchi di determinare quali righe siano spazieggiate; constaterà che la spazieggiatura non solo è visibile ad occhio nudo, ma anche che contribuisce non poco a rendere mal composto il capoverso che contenga tali righe spazieggiate. Egli constaterà anche che la raccomandazione di non spazieggiare il testo scritto in lettere minuscole è abbondantemente disattesa in quel genere di pubblicazioni; probabilmente una composizione giustificata solo a sinistra sarebbe molto più raccomandabile.

Oggi si conviene che la spazieggiatura sia accettabile e talvolta consigliabile nella composizione in lettere maiuscole o maiuscolette di titoli o di simili brevi tratti di testo; assolutamente improponibile in un testo in lettere minuscole. Il lettore attento provi ad esaminare una qualunque rivista pubblicata ai nostri giorni e cerchi di determinare quali righe siano spazieggiate; constaterà che la spazieggiatura non solo è visibile ad occhio nudo, ma anche che contribuisce non poco a rendere mal composto il capoverso che contenga tali righe spazieggiate. Egli constaterà anche che la raccomandazione di non spazieggiare il testo scritto in lettere minuscole è abbondantemente disattesa in quel genere di pubblicazioni; probabilmente una composizione giustificata solo a sinistra sarebbe molto più raccomandabile.

Oggi si conviene che la spazieggiatura sia accettabile e talvolta consigliabile nella composizione in lettere maiuscole o maiuscolette di titoli o di simili brevi tratti di testo; assolutamente improponibile in un testo in lettere minuscole. Il lettore attento provi ad esaminare una qualunque rivista pubblicata ai nostri giorni e cerchi di determinare quali righe siano spazieggiate; constaterà che la spazieggiatura non solo è visibile ad occhio nudo, ma anche che contribuisce non poco a rendere mal composto il capoverso che contenga tali righe spazieggiate. Egli constaterà anche che la raccomandazione di non spazieggiare il testo scritto in lettere minuscole è abbondantemente disattesa in quel genere di pubblicazioni; probabilmente una composizione giustificata solo a sinistra sarebbe molto più raccomandabile.

17.5 Come funziona la microtipografia

Il pacchetto *microtype* è formidabile nel realizzare le funzionalità descritte nei precedenti paragrafi, ma non funziona da solo. Perché il pacchetto possa svolgere il suo lavoro, bisogna che possa esercitare un controllo sui font che fanno parte del documento da comporre.

Questo implica che tutti i font del documento devono essere corredati di file adeguati per informare *microtype* delle caratteristiche di ogni segno e/o di ogni gruppo di segni.

Il pacchetto *microtype* dispone di un file di configurazione, `microtype.cfg`, nel quale sono contenute le impostazioni predefinite per alcuni insiemi di font, di alcune codifiche, di alcune famiglie, di alcune serie, di alcune forme e di alcuni corpi. Se l'utente vuole usare font che non appartengono agli insiemi predefiniti, può caricare il pacchetto con l'opzione *config* ma per i suoi font speciali deve predisporre file di configurazione che si devono chiamare `mt-`⟨*famiglia*⟩`.cfg`; per esempio, volendo usare i font MinionPro, allora bisogna predisporre un file di configurazione con il nome `mt-MinionPro.cfg`.

I file di configurazione devono specificare una quantità di informazioni molto dettagliate relative alle varie codifiche, famiglie, serie, forme, corpi dei font a cui si applicano e ai singoli caratteri o insiemi di caratteri presenti nel font stesso; queste informazioni riguardano sia l'ammontare di quanto ogni carattere o insieme di caratteri affini può protrudere nei margini di sinistra o di destra; sia l'ammontare di quanto ogni carattere può essere allargato o contratto in relazione ai millesimi della sua larghezza naturale. Si tratta di informazioni molto dettagliate sia nel significato di ogni misura millesimale specificata, sia in relazione al fatto che i caratteri, anche se raggruppati in insiemi di caratteri affini, sono molto numerosi.

L'utente deve essere conscio che senza queste descrizioni le funzionalità disponibili con il pacchetto *microtype* non sono usabili per determinati font, a maggior ragione per font personali, o acquistati o, comunque, scaricati dalla rete se sono gratuiti e di uso non riservato. Tutto è possibile, ma la costruzione di file di configurazione per *microtype* relativi a font particolari sono di costruzione molto delicata, quindi molto difficile. Il lettore è inviato a leggere con molta attenzione la documentazione del pacchetto *microtype* con il solito comando da terminale `texdoc microtype`.

Tuttavia queste parole non scoraggino chi vuole usare *microtype* con font di cui non dispone dei file di configurazione. Certo le informazioni metriche relative a ciascun glifo o glifi affini (per esempio: a, à, á, â,...) potrebbero essere migliori se fossero accuratamente calibrate per ogni collezione specifica di font. Infatti le impostazioni disponibili per i font preimpostati sono dei ragionevoli compromessi, perché non hanno nessuna caratteristica 'eccessiva'; per esempio i font con grazie non hanno grazie molto sporgenti, né grazie appena visibili; sono una giusta via di mezzo, per una microgiustificazione valida anche con altri font. È ovvio che se ogni raccolta di font avesse i file specifici per *microtype* il lavoro di questo pacchetto sarebbe ancora migliore, ma in mancanza bisogna fare di necessità virtù, e accontentarsi del lavoro già buono che *microtype* riesce a fare anche in questi casi.

Si esamini con cura la pagina 390, dove l'esempio della pagina 387 è ripetuto usando i font del Johannes Kepler Project (file `kpfonts`), invece dei Latin Modern.

Il primo blocco è composto con la piena funzionalità di *microtype*, cioè con la protrusione e la compressione dei font; il secondo blocco è composto senza né protrusione né compressione dei font; il terzo blocco è composto con la sola protrusione dei font. Come si vede il primo e il terzo blocco risultano composti

nello stesso modo, mentre il secondo è visibilmente differente, con spazi maggiori fra le parole e con un numero maggiore di cesure in fin di riga e con una riga in più nella composizione dell'intero capoverso. I font JKP non dispongono del loro file di configurazione, quindi le informazioni metriche sono quelle di default. Nonostante questa mancanza, il risultato con la piena funzionalità del pacchetto *microtype*, o anche con la sola protrusione, è decisamente migliore che senza tale funzionalità.

Oggi si conviene che la spazieggiatura sia accettabile e talvolta consigliabile nella composizione in lettere maiuscole o maiuscolette di titoli o di simili brevi tratti di testo; assolutamente improponibile in un testo in lettere minuscole. Il lettore attento provi ad esaminare una qualunque rivista pubblicata ai nostri giorni e cerchi di determinare quali righe siano spazieggiate; constaterà che la spazieggiatura non solo è visibile ad occhio nudo, ma anche che contribuisce non poco a rendere mal composto il capoverso che contenga tali righe spazieggiate. Egli constaterà anche che la raccomandazione di non spazieggiare il testo scritto in lettere minuscole è abbondantemente disattesa in quel genere di pubblicazioni; probabilmente una composizione giustificata solo a sinistra sarebbe molto più raccomandabile.

Oggi si conviene che la spazieggiatura sia accettabile e talvolta consigliabile nella composizione in lettere maiuscole o maiuscolette di titoli o di simili brevi tratti di testo; assolutamente improponibile in un testo in lettere minuscole. Il lettore attento provi ad esaminare una qualunque rivista pubblicata ai nostri giorni e cerchi di determinare quali righe siano spazieggiate; constaterà che la spazieggiatura non solo è visibile ad occhio nudo, ma anche che contribuisce non poco a rendere mal composto il capoverso che contenga tali righe spazieggiate. Egli constaterà anche che la raccomandazione di non spazieggiare il testo scritto in lettere minuscole è abbondantemente disattesa in quel genere di pubblicazioni; probabilmente una composizione giustificata solo a sinistra sarebbe molto più raccomandabile.

Oggi si conviene che la spazieggiatura sia accettabile e talvolta consigliabile nella composizione in lettere maiuscole o maiuscolette di titoli o di simili brevi tratti di testo; assolutamente improponibile in un testo in lettere minuscole. Il lettore attento provi ad esaminare una qualunque rivista pubblicata ai nostri giorni e cerchi di determinare quali righe siano spazieggiate; constaterà che la spazieggiatura non solo è visibile ad occhio nudo, ma anche che contribuisce non poco a rendere mal composto il capoverso che contenga tali righe spazieggiate. Egli constaterà anche che la raccomandazione di non spazieggiare il testo scritto in lettere minuscole è abbondantemente disattesa in quel genere di pubblicazioni; probabilmente una composizione giustificata solo a sinistra sarebbe molto più raccomandabile.

Capitolo 18

LaTeX: i caratteri e i font

18.1 Introduzione

I caratteri da stampa, o meglio, i segni che i caratteri imprimono sulla carta ci sono familiari perché abbiamo una grande abitudine a leggere testi scritti a stampa; tuttavia siamo assolutamente ignoranti per quanto riguarda la loro storia, l'evoluzione, gli usi, i periodi storici, e quant'altro riguardi questi piccoli oggetti e i corrispondenti piccoli segni.

Tuttavia si sono già dette alcune cose in merito ai caratteri, che qui brevemente si richiamano.

18.2 Terminologia relativa ai caratteri

font è praticamente sinonimo di 'carattere' con l'unica avvertenza che in italiano si usa la parola 'carattere', specialmente al plurale, per designare una intera polizza di segni accomunati da una omogeneità stilistica, ma si usa anche, specialmente al singolare, per designare il singolo segno. Font è equivalente al primo significato, quello di 'polizza', mentre il singolo segno in inglese viene designato *glyph*; anche in italiano si potrebbe dire glifo, ma è insolito vedere usata questa parola[1].

polizza indicava l'intera cassa di caratteri accomunati dalle stesse caratteristiche grafiche e tipografiche, incluso il numero esatto dei caratteri metallici presenti in ogni scomparto della cassa. Oggi che i caratteri metallici si usano assai poco e solo per lavori molto particolari, non ha più significato

[1] Curiosamente quando si usano parole inglesi, per lo più di genere neutro, in italiano, dove il neutro non esiste, i diversi parlanti usano il genere grammaticale che più li ispira, senza un criterio particolare. La parola *font* è una di quelle che non sembra ancora essersi stabilizzata su un determinato genere grammaticale. Qui si usa il maschile (che in italiano spesso è usato anche come neutro, come genere grammaticale indifferente); spesso si sente dire e si vede scritto *le font*, però *la font* è una espressione rarissima. In ogni caso, visto che font ormai è entrato nell'italiano, ricordiamoci che in questa lingua non prende la 's' del plurale inglese.

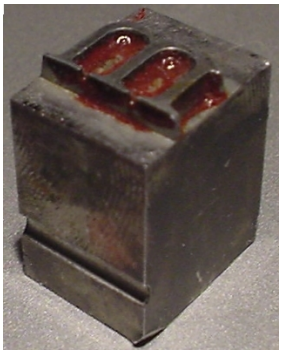


Figura 18.1: Il carattere metallico della 'm'; il disegno, ovviamente, è come allo specchio e la sinistra e la destra sono scambiate

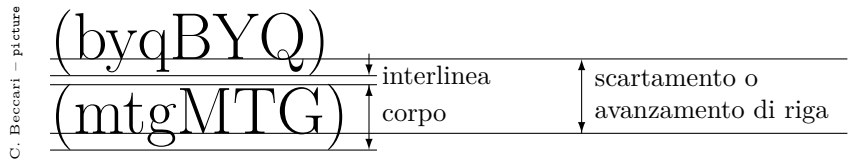


Figura 18.2: Relazioni fra il corpo, l'interlinea e l'avanzamento di riga nel caso di caratteri elettronici

usare la parola 'polizza' per designarne anche la consistenza, in quanto questa è illimitata con i caratteri elettronici, ma la parola viene usata solo per indicare una collezione di segni; quasi sempre viene sostituita dall'equivalente parola inglese *font*, al punto che questa parola si può considerare definitivamente entrata nell'uso della lingua italiana.

corpo è la dimensione reale per i caratteri metallici, e quella virtuale per i caratteri elettronici, dell'altezza del rettangolo costituito dalla faccia in rilievo del parallelepipedo che costituiva il carattere mobile metallico. Nella figura 18.1 è riportato uno di questi blocchetti e vi si può riconoscere il disegno in rilievo e rovesciato di una 'm'.

interlinea era una striscia di metallo di un dato spessore da inserire fra due righe di testo composte con i caratteri mobili; per estensione, spazio verticale da aggiungere all'altezza di una riga composta con caratteri elettronici. Oggi questo nome viene impropriamente usato al posto della locuzione corretta "avanzamento di riga" o "scartamento", per cui diventa necessario usare il termine "interlineatura" per indicare l'operazione di distanziamento di due righe di testo, quell'operazione che un tempo veniva eseguita con l'inserimento dell'interlinea metallica.

sterlineatura è il processo opposto a quello dell'interlineatura; interlineando si allontanano le righe fra di loro, sterlineando le si accosta. È evidente che questa operazione non poteva essere fatta con i caratteri metallici, perché

LA RIVOLUZIONE FRANCESE
LA PRESA DELLA BASTIGLIA

LA RIVOLUZIONE FRANCESE
LA PRESA DELLA BASTIGLIA

Figura 18.3: Composizione normale e sterlineata

non li si poteva compenetrare gli uni negli altri; al massimo potevano venire limati, o costruiti apposta, ma era una pratica costosissima e la si evitava. Con i caratteri metallici si diceva che le linee erano sterlineate quando non veniva inserita fra di loro nessuna interlinea; componendo in corpo 10/10, per esempio, si soleva dire che il testo era sterlineato.

Con i caratteri elettronici, al contrario, le righe possono venire accostate fino anche a sovrapporsi; può essere una tecnica interessante per ottenere effetti speciali in stampati pubblicitari. Nei testi di cui ci stiamo occupando, invece, la sterlineatura viene usata specialmente quando si compone con le sole lettere maiuscole, le quali sono praticamente prive di discendenti; a causa di ciò la composizione in maiuscolo con lo scartamento abituale appare troppo aperta. Nella figura 18.3 c'è un esempio di sterlineatura; nella parte superiore il testo è composto in corpo 25/30, mentre nella parte inferiore è sterlineato, composto in corpo 25/23,5, vi è, cioè, una interlineatura negativa.

Va da sé che la scelta dell'interlinea o dell'ammontare della sterlineatura va fatta con molto buon senso e con molto buon gusto da parte del grafico editoriale; dipende molto dal tipo di carattere, dalla sua serie e dalla sua forma. In generale la composizione in sole lettere maiuscole oppure in maiuscoletto e maiuscole consente delle operazioni che non sono lecite o praticabili con gli alfabeti maiuscoli/minuscoli a causa della forma delle lettere e in particolare per l'assenza o della presenza dei discendenti.

avanzamento di riga è la distanza fra le righe di base di due righe di testo composte con caratteri dello stesso corpo. Si può pensare a questa grandezza come alla somma del corpo e dell'interlinea; numericamente si tratta dello stesso valore, ma questa somma parte dalla base del rettangolo della faccia porta carattere, mentre l'avanzamento di riga parte dalla linea di base e raggiunge la linea di base della riga precedente; si veda la figura 18.2.

In Italia lo *scartamento*, o*avanzamento di riga*, viene spesso chiamato *riga* e i *tipometri* spesso sono graduati in righe e servono per misurare sia sullo stampato sia sulla gabbia della composizione in caratteri metallici proprio lo scartamento; spesso hanno diverse scale con righe di 12 punti, di 10

punti, eccetera. In sostanza l'espressione *riga di 12 punti* in Italia viene usata con il significato di *pica* nei paesi anglosassoni o di *cicero* nell'Europa continentale. Qui si eviterà accuratamente di usare la parola 'riga' in questo significato, perché è un termine troppo ambiguo.

scartamento è sinonimo di avanzamento di riga; richiama alla mente il significato ferroviario come distanza fra i binari; in tipografia è la distanza fra due linee di base di due righe di testo consecutive composte con lo stesso font; il parallelismo di queste due linee di base e quello dei due binari ferroviari è l'elemento che giustifica questo termine.

x-height è l'altezza delle lettere minuscole senza ascendenti né discendenti, cioè 'a c e m n o r s u v w x z'; questa altezza definisce anche l'unità di misura 'ex' che permette di prendere le misure in termini di altezza dei font che sono in uso. Il termine italiano per x-height è *occhio*.

quadro o quadrato era un blocco metallico quadrato usato come spaziatore nel comporre con i caratteri mobili; i due lati del quadrato erano lunghe quanto il corpo del font in uso, se non altro perché l'altezza del quadrato doveva essere uguale al corpo. Con i font elettronici si conviene che la dimensione di ciascun lato del quadrato sia pari alla larghezza della 'M' per cui si usa parlare della distanza di una M, 'em' in inglese. Se si leggono i dati forniti in calce alle tabelle 18.3–18.12, si vede che 1 em spesso non coincide affatto con il corpo anzi ne è sensibilmente diverso. Nonostante ciò questa spaziatura di un quadrato continua a fare riferimento all'unità em. In italiano il blocchetto si chiamava quadrato, e un blocchetto rettangolare largo il doppio di un quadrato si chiamava 'quadratone'; i comandi per realizzare queste spaziature si sono già incontrati negli esempi matematici; una spaziatura di un quadrato si ottiene con il comando `\quad` e di un quadratone con il comando `\qquad`.

legatura è l'operazione per la quale, per ottenere il giusto effetto estetico, certe lettere distinte devono essere accostate in modo così forte che vengono a formare un unico segno; si pensi a 'fi', 'ff', 'ffi', 'fl', 'ffl' presenti in ogni font latino; ma queste non sono le sole legature; scrivere `--` produce – e scrivere `---` produce —; si tratta sempre di legature; anche la lettera tedesca ß è il risultato della legatura di una 's lunga' e di una 's corta', in definitiva di due lettere 's' di forma diversa. In Italia non si usa più questo segno da alcuni secoli, ma i prototipografi italiani nel Quattrocento e nel Cinquecento la usavano abbondantemente; l'uso è cessato con la sparizione della 's lunga' nel diciottesimo secolo; questo segno assomigliava ad una lettera 'f', solo che la barretta non attraversava il fusto; se si esamina il segno (corsivo) della *ß*, si riconosce che la parte di sinistra è proprio una 's lunga'. Questa forma della lettera 's' veniva usata sempre come lettera iniziale o come lettera mediana, mentre la 's corta', quella che usiamo oggi, era in posizione terminale o come secondo elemento della doppia 's', da cui la legatura. Spesso nel passato erano legati anche i nessi 'ct' e 'st'. Nella figura 18.4 sono riportati due brevi testi che mostrano l'uso delle legature

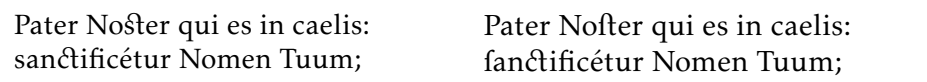


Figura 18.4: Legature antiche; il testo a sinistra mostra solo le legature dei nessi 'ct' e 'st'; quello di destra mostra la compresenza della s lunga e dalla s corta e delle legature antiche.

antiche e della s lunga[2].

grazie sono quei piccoli tratti alla fine delle aste spesse o sottili che permettono, in particolare, di allineare esattamente i singoli caratteri sulla linea di base, riempiono leggermente lo spazio fra un carattere e l'altro e danno l'idea di una linea più compatta, così che l'occhio la segue meglio e ritrova più facilmente l'inizio della riga seguente quando l'occhio deve andare a capo. I caratteri con grazie sono più adatti alla lettura continua, mentre i caratteri senza grazie affaticano il lettore se egli dovesse leggere con continuità molte pagine di seguito. Invece i caratteri senza grazie, a pari corpo, apparentemente sono più facilmente leggibili a distanza, tanto che i cartelli stradali sono composti esclusivamente con caratteri senza grazie. Non è sempre facile classificare i caratteri senza grazie; per esempio il font Optima, disegnato da Hermann Zapf, pur non avendo grazie, almeno nel senso tradizionale della parola, non può essere definito *sans serif* in virtù della sua particolare conformazione ad aste svasate che lo avvicina ai font dotati di grazie.

sans serif è l'espressione inglese per designare i font senza grazie; in italiano si chiamano preferibilmente caratteri 'lineari', o 'bastone', o 'bastoncino'; si usa anche l'aggettivo 'sgraziato', ma questa parola si presta a interpretazioni diverse.

maiuscoletto è una raccolta di segni dove l'alfabeto è composto dalle lettere maiuscole e da una serie di 'minuscole' costituite da segni con lo stesso disegno delle maiuscole ma di altezza e larghezza minori; non si tratta di maiuscole rimpicciolite, ma di lettere disegnate con le stesse caratteristiche grafiche; per esempio hanno lo stesso spessore delle aste corrispondenti; una 'A' rimpicciolita alle dimensioni di una 'a' appare così: 'ᴀ'; affiancate appaiono così: ᴀᴀ; si può percepire che la 'ᴀ' rimpicciolita è leggermente più chiara della ᴀ maiuscoletta.

proporzionale è il tipo di carattere che destina uno spazio diverso a ciascuna lettera; come ognuno può constatare la larghezza della 'm' minuscola è decisamente più grande della 'i' minuscola; non confrontiamo nemmeno la differenza fra maiuscole e minuscole perché è troppo evidente. La maggior parte, per non dire la totalità, dei caratteri usati in tipografia sono caratteri

[2]Si è scelto il testo latino anche per poter mostrare la legatura del nesso 'ct'. Il testo usa gli accenti secondo la consuetudine del latino ecclesiastico. Inoltre, rispetto alla versione della Vulgata, l'aggettivo *cotidianus* sostituisce l'originale *transubstantialis* secondo la lezione moderna.

proporzionali. Tuttavia esiste ed è utilissimo, specialmente nei testi dove si parla di informatica, poter disporre di un font a spaziatura fissa come quello usato in questo testo per descrivere i comandi usati per il mark-up di LaTeX; si osservino i due alfabeti maiuscolo e minuscolo sovrapposti:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
```

per rendersi conto di che cosa sia un carattere non proporzionale confrontandolo con i due corrispondenti alfabeti di un font proporzionale:

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

famiglia rappresenta una collezione di font di diversi disegni ma con una caratteristica comune; le famiglie più importanti per LaTeX sono quella dei caratteri a spaziatura fissa, chiamati *typewriter type*, che costituiscono la famiglia identificata dall'istruzione `\ttfamily` o dal comando `\texttt`; la famiglia dei font proporzionali senza grazie, *sans serif*, identificata dall'istruzione `\sffamily` o dal comando `\textsf`; la famiglia dei font proporzionali con grazie, *roman*, identificata dall'istruzione `\rmfamily` o dal comando `\textrm`.

serie è la caratteristica di ogni famiglia di font di apparire più o meno scura; la tipografia tradizionale prevede diversi livelli di nero che vanno dal chiarissimo al nerissimo; per LaTeX si distinguono tre livelli di nero; il medio, il nero e il nero esteso; di norma il nero 'semplice' è assai raro nelle famiglie di caratteri usate di default da LaTeX, perciò le istruzioni per scegliere queste caratteristiche di default sono solamente `\mdseries` e `\bfseries` oppure i comandi `\textmd` e `\textbf`; non tutte le famiglie dispongono di serie più scure di quella che è considerata media.

forma rappresenta il tipo di disegno di una serie di caratteri della stessa famiglia e della stessa serie; per LaTeX le forme disponibili normalmente sono la forma diritta, *upright*, la forma corsiva, *italics*, e quella maiuscoletta, *small caps*; le istruzioni per scegliere queste forme sono `\upshape`, `\itshape` e `\scshape`, mentre i comandi sono `\textup`, `\textit` e `\textsc`. Sono disponibili anche la forma tondo inclinato, *slanted*, e il corsivo diritto, *upright italics*; queste in Italia sono meno usate, tuttavia LaTeX mette a disposizione l'istruzione `\slshape` e il comando `\textsl` per il tondo inclinato. Invece per il corsivo diritto non sono disponibili né l'istruzione né il comando, sebbene non sia difficile servirsi di questa forma con comandi di livello più basso.

Si confronti il tondo inclinato con il corsivo: *abcd* e *abcd*; si nota che si tratta di due disegni (diversamente) inclinati, ma sono completamente diversi; questo è il motivo per il quale il tondo inclinato è poco usato. Il corsivo diritto appare così: abcd.

codifica è la speciale modalità informatica con la quale un particolare segno di un dato font viene identificato nel file che contiene tutti i segni di quel

particolare font. LaTeX conosce di default diverse codifiche, le più usuali per chi scrive in italiano sono la codifica `OT1` e la codifica `T1`; la prima identifica i file che contengono 128 caratteri 'latini', la seconda identifica i file che contengono 256 caratteri 'latini'; i font 'latini' contengono oltre alle cifre, ai segni di interpunzione e ad un certo numero di segni speciali, gli alfabeti minuscolo e maiuscolo delle 26 lettere 'latine':

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

I numerosi segni accentati che compaiono nelle varie lingue che usano i caratteri latini sono ottenuti sovrapponendo l'accento alla lettera con la codifica `OT1`, mentre sono ottenuti con i disegni delle lettere già accentate con la codifica `T1`.
Per capire meglio il concetto di codifica si esaminino le tabelle 18.3–18.12 che contengono i caratteri latini con codifica OT1, i caratteri latini con codifica T1, i caratteri non letterali del Text Companion Font con codifica TS1, i caratteri cirillici con codifica OT2, i caratteri greci con codifica LGR. In ogni casella di quelle tabelle c'è il segno che corrisponde alla posizione indicata dal numero decimale che compare nella stessa casella; i numeri preceduti da un solo apice o dal doppio apice nelle righe e nelle colonne esterne rappresentano parti degli indirizzi in notazione ottale o in notazione esadecimale; per esempio nella tabella 18.5 la lettera 'Í' ha un indirizzo decimale 205; questa casella si trova all'incrocio della riga con indirizzo ottale `'300` e della colonna con indirizzo ottale `'15` da cui si ricava l'indirizzo ottale complessivo eseguendo la somma dei due valori ottali: `'300 + '15 = '315`. Nella stessa maniera si ricava l'indirizzo complessivo esadecimale. Nei file interni di LaTeX sono quasi sempre usati gli indirizzi ottali o quelli esadecimali; quindi è conveniente disporre di tabelle che contengano tutti e tre i tipi di indirizzi.
Se si confrontano le tabelle 18.3 e 18.5, che si riferiscono entrambe ai caratteri latini, oltre alle ovvie differenze relative alle lettere accentate, ci sono alcuni segni uguali che si trovano ad indirizzi diversi; per esempio il segno 'ffi' si trova all'indirizzo 14 con la codifica OT1, ma all'indirizzo 30 con la codifica T1.
È assai raro dover cambiare codifica per comporre un testo; per questo LaTeX non dispone di una istruzione o di un comando di uso corrente per eseguire questo cambio di codifica, ma questo tipo di azione viene svolto dietro le quinte dalle istruzioni contenute nei file di servizio, che provvedono direttamente a cambiare l'alfabeto e la codifica a seconda della lingua che si usa. L'unico momento in cui il compositore decide di usare una delle due codifiche per i caratteri latini è all'inizio del file nel preambolo, quando ordina di usare il *font encoding* T1, la codifica T1, mediante la specifica

Normale con pdflatex; raramente con lualatex o xelatex

```
\usepackage[T1]{fontenc}
```

Se si desiderasse cambiare codifica in una posizione qualunque del testo bisognerebbe usare comandi di basso livello come `\fontencoding` e `\selectfont`, scrivendo per esempio

```
\fontencoding{T1}\selectfont
```

possibilmente all'interno di un gruppo per limitarne gli effetti al solo testo contenuto nel gruppo.
In ogni caso se in un file sorgente da comporre con xelatex o lualatex si dovesse per forza usare un font Type 1 con le codifiche descritte in questo paragrafo, bisognerebbe seguire una precisa sequenza di chiamata dei pacchetti e dei font: si veda il medaglione della sintassi nella pagina 398.

```
% Sequenza di comandi per usare un font Type 1
% in un file sorgente da compilare con XeLaTeX
% o LuaLaTeX
...
\usepackage[utf8]{inputenc}
\usepackage[⟨T1⟩]{fontenc}
\usepackage{⟨pacchetto per usare un dato font Type 1⟩}
\usepackage{fontspec}[⟨opzioni per i font OpenType⟩]
...
\begin{document}
...
{\usefont{T1}{⟨famiglia⟩}{⟨serie⟩}{⟨forma⟩}
⟨testo⟩
\par}
...
```

Ovviamente ci si possono definire comandi appositi per usare la giusta sequenza di font di basso livello. Si noti che la ⟨*famiglia*⟩ dei font Type 1 invocati con il pacchetto specifico è definita nei suoi file con estensione `.fd`. Ma da questo esempio si capisce anche come ci si deve muovere per usare in font Type 1 insieme ai font OpenType usati da xelatex e lualatex. Ci si domanda se il gioco valga la candela, però, anche se questa procedura si può evitare quasi sempre per comporre testi 'normali' potrebbe essere necessario ricorrervi per produrre testi dove, per esempio, si mostrano font codificati in modi diversi.

tipo Importantissimo per i font da usare con la tipografia assistita da calcolatore, la tipografia elettronica, è il tipo dei caratteri. Questi possono essere di tipo a *matrice di punti* o *bitmapped* o *raster*, oppure *vettoriali*. La differenza è essenziale.

I caratteri a matrice di punti sono una versione digitalizzata o discretizzata del disegno del singolo segno; ogni segno è formato da una schiera di puntini 'neri' sufficientemente accostati in modo da rendere l'idea abbastanza precisa del disegno del segno specifico. I puntini vanno benissimo se la loro densità corrisponde esattamente alla densità dei puntini della macchina da stampa o dello schermo con cui il carattere deve venire stampato o visualizzato. Non sono suscettibili di ingrandimenti o rimpicciolimenti significativi, altrimenti il segno si sgrana e si notano i vari 'quadratini' che compongono il disegno. Vanno quindi usati solo per la stampa.

(*a*) (*b*)

Figura 18.5: Il simbolo di *maschio* fortemente ingrandito: (*a*) font a matrice di punti; (*b*) font vettoriale.

I font vettoriali, invece, sono descritti dalle equazioni del loro contorno; la discretizzazione per la stampa o per lo schermo viene eseguita al volo dal programma che trasforma l'informazione codificata nel disegno del segno in un insieme di puntini neri visibili sulla carta o sullo schermo. L'ingrandimento e il rimpicciolimento del disegno vengono ottenuti cambiando semplicemente scala alle coordinate dei punti che individuano i contorni. In particolare i nodi su cui sono 'appoggiati' i vari archi che formano i contorni si possono assimilare alle punte dei vettori che uniscono l'origine ai nodi stessi; di qui la denominazione di disegni *vettoriali*.
La differenza sembra quindi nel fatto che i segni a matrici di punti sono memorizzati come matrice di puntini neri, mentre i segni vettoriali sono trasformati in matrice di puntini neri solo al momento dell'uso.
Non è così, almeno non lo è quando il supporto di 'stampa' è lo schermo; in realtà non è così nemmeno quando il supporto è la carta, ma la differenza è molto maggiore quando si considera lo schermo.
Chi legge un testo direttamente da uno schermo tende ad adattare l'ingrandimento della pagina sia alle dimensioni dello schermo fisico sia al tipo di dettaglio che desidera osservare. Si consideri allora la figura 18.5.
Si vede chiaramente, specialmente lungo il contorno, la presenza dei quadratini neri nel disegno (*a*) che è a matrice di punti, ma non si vede nessuna granulosità nel contorno del disegno (*b*) che è vettoriale.
In pratica, con il sistema TEX i font originariamente prodotti con il programma METAFONT, quando la grafica degli schermi non era granché, sono a matrici di punti; il programma METAFONT è però sofisticatissimo ed è

in grado di generare delle matrici di punti anche con una densità adeguata ad una macchina di fotocomposizione professionale, con densità di punti di più di 4000 punti al pollice; inoltre METAFONT può tenere conto anche di densità diverse in orizzontale rispetto alla direzione verticale.
Con l'avvento dei primi font vettoriali da parte della Adobe, cioè dei font PostScript Type 1, la tecnologia si è sviluppata moltissimo[3]; con una iniziativa di collaborazione fra Apple e Microsoft vennero sviluppati poi i font vettoriali TrueType; successivamente sono stati sviluppati anche i font OpenType, sempre vettoriali, ma molto più completi dei Type 1 e dei TrueType. Con i programmi del sistema TeX i font OpenType possono essere usati direttamente solo da xelatex e lualatex, oltre che da ConTeXt MK IV. Nel seguito si darà molta importanza ai font vettoriali e si descriveranno molte azioni per poter rendere disponibili all'uso anche i font Type 1 e TrueType commerciali o anche quelli non ancora completamente integrati nella distribuzione del sistema TeX.

18.3 I comandi per la scelta dei font

Per scegliere i font il compositore dispone di una serie di istruzioni e di comandi che sono già stati quasi tutti esposti nel paragrafo precedente. Vale però la pena di riepilogare anche per distinguere sistematicamente i vari comandi e le corrispondenti dichiarazioni a seconda del contesto in cui si usano.

18.3.1 La scelta del corpo e dello scartamento

LaTeX dispone di una serie di comandi che si riferiscono al corpo dei caratteri in modo simbolico, invece che numerico; essi sono raccolti nella tabella 18.1 e indicano tutti quanti (in inglese) la relazione che il corpo prescelto ha rispetto al corpo normale.

Il corpo che funziona da corpo normale è quello di default, cioè di 10 pt, oppure quello che si specifica fra le opzioni del comando di apertura `\documentclass`; fra queste opzioni, in generale, ci sono solo *11pt* e *12pt*. Alcune classi non standard offrono anche font normali di corpo più piccolo o più grande.

Bisogna rilevare una differenza sostanziale fra i font; ne esistono di due tipi diversi; (*a*) quelli con i corpi ottici e (*b*) quelli in un corpo unico. La differenza sostanziale è la seguente e riguarda prevalentemente i font vettoriali, ma non è una loro esclusività. I font vettoriali, proprio perché la loro natura permette di scalarli con continuità, vengono solitamente distribuiti in un corpo unico e vengono ingranditi o rimpiccioliti con i comandi della tabella 18.1 partendo dalla

[3]Incidentalmente sono classificati font di tipo Type 3 i font ottenuti per accostamento di tante parti distinte con possibili sovrapposizioni di alcune parti; i font bitmapped, quando vengono inclusi in un documento PDF, vengono classificati come Type 3. Si ricordi che per qualunque operazione di stampa più o meno commerciale un documento che faccia uso di font Type 3 viene generalmente respinto.

Istruzione	Esempio
`\tiny`	ABCDEFG abcdefg
`\scriptsize`	ABCDEFG abcdefg
`\footnotesize`	ABCDEFG abcdefg
`\small`	ABCDEFG abcdefg
`\normalsize`	ABCDEFG abcdefg
`\large`	ABCDEFG abcdefg
`\Large`	ABCDEFG abcdefg
`\LARGE`	ABCDEFG abcdefg
`\huge`	ABCDEFG abcdefg
`\Huge`	ABCDEFG abcdefg

Tabella 18.1: Istruzioni per la scelta del corpo dei caratteri

loro unica realizzazione fisica; questa è generalmente di corpo 10 pt, talvolta leggermente più grande, ma queste informazioni sono conservate nel file metrico che li accompagna e che il programma di composizione interpreta correttamente per inserirli nel documento finale nel corpo richiesto.

Invece il primo tipo di font, dotato di corpi ottici, dispone di una varietà di disegni leggermente diversi da corpo a corpo in modo che quelli più piccoli del font "normale" non diventino troppo fini e leggeri, con il pericolo di perdere qualche dettaglio molto fine, e quelli più grandi del normale non appaiano troppo neri.

Nella tabella 18.2 si vede direttamente l'effetto che i comandi di corpo producono su font dotati di corpi ottici e su quelli che vengono distribuiti in un solo corpo. nella prima riga di ogni coppia appare il font in corpo 10 scalato al corpo specificato nella seconda colonna; mentre nella seconda riga di ogni coppia appare il font nella sua forma come prescritta dall'apposito comando di corpo. Si vede chiaramente che i font in corpo unico hanno lo stesso disegno ingrandito o rimpicciolito sia che si scalino sia che si usino i comandi di corpo. Invece i font dotati di corpi ottici hanno un disegno diverso per i vari corpi, per cui scalando verso corpi minori l'alfabeto si accorcia e le linee diventano troppo sottili, mentre scalando verso corpi maggiori l'alfabeto si allunga e le linee diventano troppo spesse così che il font sembra nero invece che di serie media.

In questa tabella infatti si vede che cosa succede ingrandendo o rimpicciolendo

Latin Modern	
abcdefghijklmnopqrstuvwxyz abcdefghijklmnopqrstuvwxyz	Font scalato a 8pt `\footnotesize`
abcdefghijklmnopqrstuvwxyz abcdefghijklmnopqrstuvwxyz	Font scalato a 9pt `\small`
abcdefghijklmnopqrstuvwxyz abcdefghijklmnopqrstuvwxyz	Font scalato a 12pt `\large`
abcdefghijklmnopqrstuvwxyz abcdefghijklmnopqrstuvwxyz	Font scalato a 14.4pt `\Large`
abcdefghijklmnopqrstuvwxyz abcdefghijklmnopqrstuvwxyz	Font scalato a 17.28pt `\LARGE`
Times eXtended	
abcdefghijklmnopqrstuvwxyz abcdefghijklmnopqrstuvwxyz	Font scalato a 8pt `\footnotesize`
abcdefghijklmnopqrstuvwxyz	Font scalato a 9pt
abcdefghijklmnopqrstuvwxyz	`\small`
abcdefghijklmnopqrstuvwxyz abcdefghijklmnopqrstuvwxyz	Font scalato a 12pt `\large`
abcdefghijklmnopqrstuvwxyz abcdefghijklmnopqrstuvwxyz	Font scalato a 14.4pt `\Large`
abcdefghijklmnopqrstuvwxyz abcdefghijklmnopqrstuvwxyz	Font scalato a 17.28pt `\LARGE`)

Tabella 18.2: Confronto fra l'ingrandimento di un font dotato di corpi ottici con un font dotato di un solo corpo

un font di corpo 10 pt oppure regolandone il corpo con i comandi della tabella 18.1; nella seconda metà della tabella si vede che cosa succede con con un font come il Times eXtended dotato di un solo corpo; l'ingrandimento e l'uso dei comandi di corpo producono esattamente lo stesso risultato. Invece nella prima metà della tabella l'uso dei comandi di corpo sui font Latin Modern dotati di corpi ottici

produce il risultato voluto, mentre l'ingrandimento o il rimpicciolimento produce un risultato diverso: più piccolo nel caso del rimpicciolimento al punto di rendere illeggibili i corpi minori, e più grande e più nero nel caso dell'ingrandimento per i corpi maggiori.

Questo fatto deve essere tenuto in conto quando si scelgono font diversi da quelli predefiniti. Si tenga presente che quasi tutti i font vettoriali commerciali distribuiti sia insieme al sistema TEX (gratuitamente usabili senza restrizioni) sia quelli acquistati preso i produttori (con licenze variamente onerose) sono generalmente prodotti in un solo corpo. Fanno eccezione alcuni di quelli nati con il sistema TEX e alcuni font molto professionali, e molto costosi, che spesso vengono distribuiti solamente in quattro corpi ottici.

Ancora un dettaglio: i font Latin Modern sono corredati da file di descrizione che consentono di ingrandire e rimpicciolire i font a corpi qualunque; l'operazione viene fatta sul corpo ottico di valore più vicino a quello richiesto, affinché la variazione di corpo rispetto a quello di partenza sia la minima possibile. In questo modo i "difetti" prodotti dal cambiamento di scala sono ridotti al minimo e sono del tutto impercettibili. Sul modello di questo file di descrizione sono stati costruiti anche quelli dei font greci predefiniti (dotati di corpi ottici) da usarsi quando i font latini corrispondenti sono i Latin Modern.

I corpi più piccoli di quello normale si differenziano di un punto l'uno dall'altro; i corpi più grandi sono invece in proporzione geometrica approssimativamente di ragione 1,2. Per il corpo normale di 10 pt i corpi specificati con i comandi della tabella 18.1 si susseguono così: 5 pt, 7 pt, 8 pt, 9 pt, 10 pt, 12 pt, 14,4 pt, 17,28 pt, 20,74 pt, 24,88 pt. Fuori della sequenza in progressione geometrica c'è anche il corpo di 10,95 pt, usato come corpo normale con l'opzione di *11pt*; si noti che questo corpo approssima la media geometrica fra i valori adiacenti 10 pt e 12 pt: $10{,}95 \approx \sqrt{10 \times 12}$.

Gli scartamenti sono prefissati a circa il 20% in più del corpo a seconda delle classi dei documenti. Se proprio si desidera allargare o restringere l'avanzamento di riga rispetto al valore di default (a parte scriversi un file di classe personalizzato) si può usare il comando `\linespread` il cui argomento funziona da moltiplicatore dell'avanzamento di default. Per limitarne l'efficacia bisogna dare questo comando dentro un ambiente oppure dentro un gruppo. Si tenga però presente che l'effetto estetico combinato del corpo e dello scartamento, insieme all'effetto che questi valori hanno sulla comodità di lettura, sono estremamente delicati e solo un grafico editoriale sa come modificare i valori di default di frazioni di punto percentuale; se un inesperto si mette a giocare con `\linespread` sappia che non è consigliabile superare mai il 5% in più o in meno; valori maggiori producono di solito risultati pessimi.

18.3.2 La scelta delle altre caratteristiche

Le altre caratteristiche dei font, tranne la codifica, che, come si è detto, non viene cambiata quasi mai o viene cambiata automaticamente quando si cambia alfabeto, possono essere modificate autonomamente l'una dall'altra con i comandi già

	’00	’01	’02	’03	’04	’05	’06	’07	’10	’11	’12	’13	’14	’15	’16	’17	
’000	Γ 0	Δ 1	Θ 2	Λ 3	Ξ 4	Π 5	Σ 6	Υ 7	Φ 8	Ψ 9	Ω 10	ff 11	fi 12	fl 13	ffi 14	ffl 15	"00
’020	ı 16	ȷ 17	` 18	´ 19	˘ 20	ˇ 21	¯ 22	° 23	¸ 24	ß 25	æ 26	œ 27	ø 28	Æ 29	Œ 30	Ø 31	"10
’040	␣ 32	! 33	” 34	# 35	$ 36	% 37	& 38	’ 39	(40	) 41	* 42	+ 43	, 44	- 45	. 46	/ 47	"20
’060	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	: 58	; 59	¡ 60	= 61	¿ 62	? 63	"30
’100	@ 64	A 65	B 66	C 67	D 68	E 69	F 70	G 71	H 72	I 73	J 74	K 75	L 76	M 77	N 78	O 79	"40
’120	P 80	Q 81	R 82	S 83	T 84	U 85	V 86	W 87	X 88	Y 89	Z 90	[91	“ 92	] 93	^ 94	˙ 95	"50
’140	‘ 96	a 97	b 98	c 99	d 100	e 101	f 102	g 103	h 104	i 105	j 106	k 107	l 108	m 109	n 110	o 111	"60
’160	p 112	q 113	r 114	s 115	t 116	u 117	v 118	w 119	x 120	y 121	z 122	– 123	— 124	˝ 125	˜ 126	¨ 127	"70
	"00	"01	"02	"03	"04	"05	"06	"07	"08	"09	"0A	"0B	"0C	"0D	"0E	"0F	

Parametri

inclinazione	0,00000	x-height	4,30554 pt
spazio interparola	3,33332 pt	larghezza del quadrato	10,19999 pt
allungamento interparola	1,66665 pt	spazio extra	1,11111 pt
accorciamento interparola	1,11111 pt	corpo nominale	10,00000 pt

Tabella 18.3: Il font latino a 128 caratteri con codifica OT1

	’00	’01	’02	’03	’04	’05	’06	’07	’10	’11	’12	’13	’14	’15	’16	’17	
’000	Њ 0	Љ 1	Ц 2	Э 3	І 4	Є 5	Ъ 6	Ћ 7	њ 8	љ 9	ц 10	э 11	і 12	є 13	ђ 14	ћ 15	"00
’020	Ю 16	Ж 17	Й 18	Ё 19	Ѵ 20	Ѳ 21	S 22	Я 23	ю 24	ж 25	й 26	ё 27	ѵ 28	ѳ 29	s 30	я 31	"10
’040	¨ 32	! 33	” 34	Ѣ 35	˘ 36	% 37	´ 38	’ 39	(40	) 41	* 42	Ь 43	, 44	- 45	. 46	/ 47	"20
’060	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	: 58	; 59	« 60	ı 61	» 62	? 63	"30
’100	ˇ 64	А 65	Б 66	Ц 67	Д 68	Е 69	Ф 70	Г 71	Х 72	И 73	Й 74	К 75	Л 76	М 77	Н 78	О 79	"40
’120	П 80	Ч 81	Р 82	С 83	Т 84	У 85	В 86	Ш 87	Щ 88	Ы 89	З 90	[91	“ 92	] 93	Ь 94	Ъ 95	"50
’140	‘ 96	а 97	б 98	ц 99	д 100	е 101	ф 102	г 103	х 104	и 105	й 106	к 107	л 108	м 109	н 110	о 111	"60
’160	п 112	ч 113	р 114	с 115	т 116	у 117	в 118	ш 119	щ 120	ы 121	з 122	– 123	— 124	№ 125	ь 126	ъ 127	"70
	"00	"01	"02	"03	"04	"05	"06	"07	"08	"09	"0A	"0B	"0C	"0D	"0E	"0F	

Parametri

inclinazione	0,00000	x-height	4,30554 pt
spazio interparola	3,77774 pt	larghezza del quadrato	11,55450 pt
allungamento interparola	1,74996 pt	spazio extra	1,16665 pt
accorciamento interparola	1,16665 pt	corpo nominale	10,00000 pt

Tabella 18.4: Il font cirillico a 128 caratteri con codifica OT2

descritti; se una certa combinazione di famiglia, serie e forma non è disponibile, LaTeX provvede da solo a sostituire il font mancante con un font di default o con un font che abbia il maggior numero possibile delle caratteristiche richieste; nella tabella 18.6 vengono illustrate quasi tutte le combinazioni. Si noterà che in alcune caselle il contenuto è rosso; LaTeX ha sostituito il font mancante con un font che abbia un certo numero di caratteristiche fra quelle richieste.

Per confrontare l’effetto dei comandi per la scelta dei font in modo testo o in modo matematico si esamini il medaglione nella pagina 728.

	’00	’01	’02	’03	’04	’05	’06	’07	’10	’11	’12	’13	’14	’15	’16	’17	
’000	` 0	´ 1	^ 2	~ 3	¨ 4	˝ 5	° 6	ˇ 7	˘ 8	¯ 9	· 10	¸ 11	˛ 12	, 13	‹ 14	› 15	"00
’020	“ 16	” 17	„ 18	« 19	» 20	– 21	— 22	23	ı 24	ȷ 25	ff 26	fi 27	fl 28	ffi 29	ffl 30	31	"10
’040	␣ 32	! 33	" 34	# 35	$ 36	% 37	& 38	’ 39	(40	) 41	* 42	+ 43	, 44	- 45	. 46	/ 47	"20
’060	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	: 58	; 59	< 60	= 61	> 62	? 63	"30
’100	@ 64	A 65	B 66	C 67	D 68	E 69	F 70	G 71	H 72	I 73	J 74	K 75	L 76	M 77	N 78	O 79	"40
’120	P 80	Q 81	R 82	S 83	T 84	U 85	V 86	W 87	X 88	Y 89	Z 90	[91	\ 92	] 93	^ 94	_ 95	"50
’140	‘ 96	a 97	b 98	c 99	d 100	e 101	f 102	g 103	h 104	i 105	j 106	k 107	l 108	m 109	n 110	o 111	"60
’160	p 112	q 113	r 114	s 115	t 116	u 117	v 118	w 119	x 120	y 121	z 122	{ 123	| 124	} 125	~ 126	- 127	"70
’200	Ă 128	Ą 129	Ć 130	Č 131	Ď 132	Đ 133	Ę 134	Ğ 135	Ĺ 136	Ľ 137	Ł 138	Ń 139	Ň 140	Ŋ 141	Ő 142	Ŕ 143	"80
’220	Ř 144	Ś 145	Š 146	Ş 147	Ť 148	Ţ 149	Ű 150	Ů 151	Ÿ 152	Ź 153	Ž 154	Ż 155	IJ 156	İ 157	đ 158	§ 159	"90
’240	ă 160	ą 161	ć 162	č 163	ď 164	đ 165	ę 166	ğ 167	ı 168	ľ 169	ł 170	ń 171	ň 172	ŋ 173	ő 174	ŕ 175	"A0
’260	ř 176	ś 177	š 178	ş 179	ť 180	ţ 181	ű 182	ů 183	ÿ 184	ź 185	ž 186	ż 187	ij 188	¡ 189	¿ 190	£ 191	"B0
’300	À 192	Á 193	Â 194	Ã 195	Ä 196	Å 197	Æ 198	Ç 199	È 200	É 201	Ê 202	Ë 203	Ì 204	Í 205	Î 206	Ï 207	"C0
’320	Ð 208	Ñ 209	Ò 210	Ó 211	Ô 212	Õ 213	Ö 214	Œ 215	Ø 216	Ù 217	Ú 218	Û 219	Ü 220	Ý 221	Þ 222	SS 223	"D0
’340	à 224	á 225	â 226	ã 227	ä 228	å 229	æ 230	ç 231	è 232	é 233	ê 234	ë 235	ì 236	í 237	î 238	ï 239	"E0
’360	ð 240	ñ 241	ò 242	ó 243	ô 244	õ 245	ö 246	œ 247	ø 248	ù 249	ú 250	û 251	ü 252	ý 253	þ 254	ß 255	"F0
	"00	"01	"02	"03	"04	"05	"06	"07	"08	"09	"0A	"0B	"0C	"0D	"0E	"0F	

Parametri

inclinazione	0,00000	x-height	4,30450 pt
spazio interparola	3,33251 pt	larghezza del quadrato	9,99755 pt
allungamento interparola	1,66625 pt	spazio extra	1,11083 pt
accorciamento interparola	1,11083 pt	corpo nominale	10,00000 pt

Tabella 18.5: Il font latino a 256 caratteri con codifica T1

Vale la pena di mettere in guardia gli utenti che si servono di modelli di documento composti per il vecchio LaTeX 209, nei quali si usano comandi abbreviati del tipo `\rm` per scegliere il font tondo, o `\it` per scegliere il corsivo, eccetera. Quei vecchi comandi possono essere stati conservati per compatibilità in alcune classi standard, ma operano molto diversamente dai comandi della tabella 18.6. Essi infatti simulano perfettamente anche in LaTeX 2_ε il funzionamento che essi avevano in LaTeX 209. Cioè essi impostano una sola caratteristica ripristinando le altre caratteristiche ai valori di default; essi, di fatto, eseguono un comando `\normalfont` che imposta tutti i valori di default (famiglia con grazie, forma diritta, serie media) e poi specificano solo la famiglia o solo la forma o solo la serie richiesta. È ovvio che quei vecchi comandi *non* devono venire più usati.

Inoltre l’abitudine all’uso di quei vecchi comandi può indurre a pensare che la dichiarazione `\rmfamily` e il corrispondente comando `\textrm` siano in tutto e per tutto equivalenti a `\rm`: evidentemente questo è falso, perché la prima dichiarazione e il corrispondente comando impostano solo la famiglia, lasciando inalterata la forma e la serie preesistenti; `\rm`, invece, reimposta tutti i valori di default, anche la serie e la forma, e poi specifica la forma diritta. Gli utenti dei vecchi comandi di LaTeX 209, sono quindi pregati di dimenticarli completamente,

		\upshape	\slshape	\itshape	\scshape	\uishape
\mdseries	\rmfamily	abcABC	*abcABC*	*abcABC*	ABCABC	abcABC
	\sffamily	abcABC	*abcABC*	*abcABC*	ABCABC	abcABC
	\ttfamily	abcABC	*abcABC*	*abcABC*	ABCABC	abcABC
\bfseries	\rmfamily	**abcABC**	***abcABC***	***abcABC***	**abcABC**	**abcABC**
	\sffamily	**abcABC**	***abcABC***	***abcABC***	**abcABC**	**abcABC**
	\ttfamily	**abcABC**	***abcABC***	***abcABC***	**abcABC**	**abcABC**

Tabella 18.6: Le varie combinazioni di serie e di forma per le varie famiglie standard dei font Computer Modern usabili con LaTeX. I caratteri rossi sono stati sostituiti con altri caratteri aventi il maggior numero possibile delle caratteristiche richieste. Con i font latini aventi la codifica T1 si hanno meno sostituzioni; solo la famiglia *typewriter type* continua a non disporre della serie nera. La collezione dei font latini Latin Modern oltre la famiglia *typewriter type*, contiene anche la famiglia *variable typewriter type* (macchina da scrivere a spaziatura variabile); per entrambe esiste la serie nera ma solo nelle forme dritta e inclinata.

benché avessero qualche funzionalità comoda, come per esempio quella di poter essere usati anche in matematica.

Piuttosto tutti gli utenti ricordino che i nuovi comandi `\text`⟨*xx*⟩ possono venire usati anche in matematica e, sotto certe condizioni (vedi più avanti), persino in posizioni di apici o di pedici, per scrivere brevi apposizioni testuali, dove però gli spazi, gli accenti testuali, eccetera possono essere usati regolarmente come se si trattasse di comporre del testo normale. Benché sia possibile, nel paragrafo 18.5 verrà spiegato in dettaglio perché non sia opportuno servirsi di questi comandi per i pedici e per gli apici.

Nella tabella 18.6 compaiono solo le istruzioni per assegnare una certa caratteristica al font scelto. Fra le altre cose si è indicata l'istruzione `\uishape` che in LaTeX non esiste; la si è definita mediante un comando, che verrà descritto meglio in seguito, in modo da creare un comando per l'utente che non lo obblighi a ricorrere a istruzioni di livello inferiore:

```
\newcommand{\uishape}{\fontshape{ui}\selectfont}
\DeclareTextFontCommand{\textui}{\uishape}
```

Le dichiarazioni usate nella composizione della tabella possono essere usate anche durante la composizione del testo; tuttavia per comporre singole parole o brevi frasi è preferibile ricorrere ai comandi; tutti sono della forma `\text`⟨*xx*⟩, dove ⟨*xx*⟩ sono le stesse due lettere che compaiono all'inizio delle istruzioni; avremo quindi `\textrm`, `\textsl`, `\textbf`, `\textsf`, eccetera.

Attenzione: Nel 2020 il LaTeX3 Project Team ha esteso i comandi di serie e di forma; la cosa è diventata più delicata, ma mentre la tabella 18.6 mostra i comandi abitualmente usati dalla maggioranza degli utenti, la guida dei font [1] descrive la moltitudine di nuovi codici da usare in circostanze molto particolari,

nei dettagli delle quali qui non è opportuno scendere; si veda anche quanto scritto a questo proposito nel capitolo **??**.

La sintassi di tutti questi comandi è la seguente:

`\text`⟨*xx*⟩`{`⟨*testo*⟩`}`

Perché è preferibile usare i comandi rispetto alle istruzioni? Perché con il comando il programma LaTeX può conoscere dove comincia e dove finisce il testo da comporre con una particolare caratteristica e quindi sa dove inserire la 'correzione italica', una piccola correzione per allontanare o per avvicinare i segni che precedono o seguono il ⟨*testo*⟩. Spesso i caratteri che seguono sono dei segni di interpunzione 'alti' come i due punti o il punto e virgola, oppure come il punto esclamativo o interrogativo; talvolta si tratta di parentesi; si noti allora l'effetto di chiudere fra parentesi una parola scritta in corsivo se non è presente o se è presente la correzione italica: (*XYZ*) a confronto con (*XYZ*). Chiaramente il secondo caso è stato composto con il comando e il primo con l'istruzione; nel primo la parentesi chiusa batte contro la *Z* nel secondo no[4]. Si osservi infine che il caso più frequente di necessità della correzione italica si ha quando si usa il carattere corsivo; tuttavia il fenomeno si produce anche con il tondo[5]: fF e fF.

Fra i comandi per scegliere la forma dei segni di un font non bisogna scordare l'istruzione `\em` e il comando `\emph`; essi servono per evidenziare (in inglese *emphasize*) una parola o una breve locuzione all'interno di altro testo. Questa istruzione e questo comando hanno la proprietà di cambiare inclinazione al font (eventualmente passando dal tondo al corsivo, oppure al tondo inclinato), per cui in questo testo scritto in tondo la parola *emphasize* è evidenziata in corsivo, mentre *in questo testo composto in corsivo* emphasize *viene evidenziato in tondo*. In entrambi i casi si è scritto `\emph{emphasize}`.

È d'obbligo segnalare un fatto importante; le varie forme, serie e corpi, nonché le varie famiglie di font vanno usate con criterio. L'abitudine di evidenziare le parole in corsivo o in neretto, oppure ancora in corsivo nero è una operazione da inesperto. Esistono delle norme editoriali per l'uso di queste varianti dei caratteri e non è il caso di abusarne. In tipografia non si sottolineano mai parole o frasi per metterle in evidenza, come si faceva con la macchina da scrivere. Esistono poi anche delle 'norme' psicosociologiche che sconsigliano di cambiare font troppo spesso; se qualcosa deve essere evidenziato, questo non vuol dire evidenziare tutto. Alcuni studenti usano il pennarello colorato per evidenziare le parti che ritengono importanti; finiscono con l'evidenziare quasi tutto e il risultato ovvio è che nulla emerge dallo sfondo colorato, se non, appunto, le cose meno importanti!

18.4 Altri font diversi da quelli di default

Talvolta è necessario usare font diversi da quelli di default per ottenere effetti diversi, per uniformarsi allo stile di documenti scritti con altri metodi, per

[4] Si noti che in tipografia si usano le parentesi non inclinate anche quando esse racchiudono testo scritto con caratteri inclinati.

[5] L'esempio non è cervellotico: fF è il simbolo dell'unità di misura 'femtofarad' che si incontra spesso in microelettronica.

aumentare la leggibilità, per rendere più compatto il testo, eccetera.

Sono stati predisposti diversi pacchetti per sostituire le famiglie di font usate di default con altre famiglie, generalmente con lo scopo di usare famiglie di font commerciali, anche se disponibili gratuitamente. Sul mercato, infatti, sono disponibili migliaia di famiglie di font. Essi costituiscono la croce e delizia dei disegnatori editoriali e richiedono, appunto, la loro professionalità per usarli al meglio. Per gli inesperti può essere difficile da un punto di vista estetico e funzionale usare i font commerciali, ma la presenza dei pacchetti per l'uso di questi font rende l'operazione tecnicamente molto più facile e l'uso di questi font diventa alla portata di tutti.

Si possono dunque invocare diversi pacchetti che servono per sostituire una alla volta le tre famiglie principali di font; tuttavia la cosa non è semplice, o meglio è semplicissimo con LaTeX, ma è difficile eseguire scelte in cui le famiglie si accordino alla perfezione, in particolare per quel che riguarda l'altezza delle minuscole, la famosa *x-height*.

Per questo scopo, allora si consiglia ai lettori di fare riferimento a due pacchetti in particolare: `txfonts` e `pxfonts`. Meglio ancora: recentemente sono stati caricati su CTAN e inseriti nella distribuzione di TeX Live, le revisioni di questi due pacchetti; la documentazione si legge con `texdoc newtxfonts` e `texdoc newpxfonts`, ma i pacchetti da caricare sono distinti per il testo e per la matematica: quindi si dispone di `newtxtext` e `newtxmath` per i nuovi font Times, e `newpxtext` e `newpxmath` per i Palatino. Questa separazione consente di usare i font per la matematica anche con altri font testuali, e viceversa di usare altri font matematici con i nuovi Times o Palatino. Questa operazione può portare ad abbinamenti non tanto buoni, quindi è opportuno leggere con attenzione le documentazioni indicate sopra, dove viene spiegato più dettagliatamente il perché di questa novità.

Il lettore confronti i capoversi presenti nella figura 18.6; si tratta di due testi composti con font dello stesso corpo (10 pt) e con lo stesso scartamento (12 pt), ma con collezioni diverse di font. Eventualmente usi una lente di ingrandimento (anche quella fornita dal programma di visualizzazione del file PDF) per esaminare i dettagli di forma delle lettere; inoltre noti il differente livello di nero dei testi composti con questi font.

18.5 I font per la matematica

I font per la matematica sono un argomento un po' particolare perché sono una specialità del sistema TeX. Ci sono anche altri font dotati di segni matematici, ma di solito sono assai modesti come qualità tipografica, a meno che non siano stati adattati completamente al sistema TeX, come per esempio i font Times forniti con il pacchetto `txfonts` o `newtxmath`, oppure i font Palatino forniti con il pacchetto `pxfonts` o `newpxmath`; di questi font si è già detto in precedenza.

Tuttavia è doveroso presentare anche le tabelle dei font matematici, affinché, sia pure sotto forma di una specie di catalogo, ci si possa rendere conto di quali

Testo composto con i nuovi font Times extended

Il pacchetto **newtxtext** consente di usare i font Times per il testo con grazie, il font Helvetica per il testo scritto senza grazie, e una variante del Courier per la famiglia a spaziatura fissa. I font sono caricati con fattori di scala predefiniti in modo che le altezze *x-height* siano il più possibile simili, non esattamente identiche, perché la forma dei segni richiede delle curve leggermente abbondanti specialmente con l'Helvetica.

La 'x' presente nel nome di questo pacchetto richiama il fatto che si tratta di una collezione estesa; infatti con l'uso di questi font diventano disponibili non solo tutti i simboli di LaTeX standard arricchito dai font del pacchetto **textcomp** ma una ulteriore varietà che ne aumenta ancora la versatilità. Quindi **tx** sta per 'Times eXtended'.

Testo composto con i nuovi font Palatino extended

L'altro pacchetto **newpxtext** mette a disposizione il Palatino esteso; il Palatino viene usato al posto del Times, ma per le altre famiglie si usano ancora l'Helvetica e il Courier. In questo caso i fattori di scala con cui sono invocati i font accessori sono diversi da quelli del Times, perché il Palatino a pari corpo appare decisamente più grande e più largo.

Figura 18.6: Esempi di testi composti con i nuovi font Times e Palatino estesi

segni sono disponibili e quali sono gli indirizzi di ogni segno all'interno di ogni polizza.

Le tabelle 18.7 e 18.8 contengono la polizza del corsivo matematico, che ha caratteristiche assai diverse dal font corsivo per il testo; la polizza contiene anche un certo numero di simboli e l'alfabeto greco maiuscolo e minuscolo oltre alle cifre arabe *minuscole*. I grandi maestri della tipografia unanimemente raccomandano di usare le cifre arabe minuscole nel testo e di usare quelle maiuscole nelle tabelle e in matematica. La polizza dei simboli, oltre a numerosi simboli matematici e per altri scopi, contiene anche l'alfabeto calligrafico maiuscolo; l'alfabeto calligrafico viene selezionato in matematica con il comando `\mathcal`.

Vale la pena di commentare la polizza del segni estensibili 18.9, perché la tabella sembra formata in modo diverso dalle altre. Non è 'colpa' della tabella, ma dei font; tutti i segni hanno una altezza nulla e una profondità corrispondente alla effettiva dimensione verticale del segno; questo serve alla sezione di composizione della matematica per poter collocare correttamente tutti i segni in quella disposizione bidimensionale costituita da una espressione matematica. Si notino anche i mezzi delimitatori e i pezzetti che permettono, sovrapponendoli in modo scalato, di disegnare qualunque delimitatore con qualsiasi dimensione sia in altezza per quelli che devono espandersi in verticale, sia in larghezza per quelli che devono espandersi in orizzontale.

La gestione di questi simboli estensibili, come si può facilmente immaginare, è piuttosto complessa; il bello è che l'interprete fa tutto da solo e il compositore

	’00	’01	’02	’03	’04	’05	’06	’07	’10	’11	’12	’13	’14	’15	’16	’17	
’000	Γ_{0}	Δ_{1}	Θ_{2}	Λ_{3}	Ξ_{4}	Π_{5}	Σ_{6}	Υ_{7}	Φ_{8}	Ψ_{9}	Ω_{10}	α_{11}	β_{12}	γ_{13}	δ_{14}	ϵ_{15}	"00
’020	ζ_{16}	η_{17}	θ_{18}	ι_{19}	κ_{20}	λ_{21}	μ_{22}	ν_{23}	ξ_{24}	π_{25}	ρ_{26}	σ_{27}	τ_{28}	υ_{29}	ϕ_{30}	χ_{31}	"10
’040	ψ_{32}	ω_{33}	ε_{34}	ϑ_{35}	ϖ_{36}	ϱ_{37}	ς_{38}	φ_{39}	$\leftharpoonup_{40}$	$\leftharpoondown_{41}$	$\rightharpoonup_{42}$	$\rightharpoondown_{43}$	$\triangleleft_{44}$	$\triangleright_{45}$	$\triangleright_{46}$	$\triangleleft_{47}$	"20
’060	0_{48}	1_{49}	2_{50}	3_{51}	4_{52}	5_{53}	6_{54}	7_{55}	8_{56}	9_{57}	$.{}_{58}$	$,{}_{59}$	$<_{60}$	$/_{61}$	$>_{62}$	$\star_{63}$	"30
’100	∂_{64}	A_{65}	B_{66}	C_{67}	D_{68}	E_{69}	F_{70}	G_{71}	H_{72}	I_{73}	J_{74}	K_{75}	L_{76}	M_{77}	N_{78}	O_{79}	"40
’120	P_{80}	Q_{81}	R_{82}	S_{83}	T_{84}	U_{85}	V_{86}	W_{87}	X_{88}	Y_{89}	Z_{90}	$\flat_{91}$	$\natural_{92}$	$\sharp_{93}$	$\smile_{94}$	$\frown_{95}$	"50
’140	ℓ_{96}	a_{97}	b_{98}	c_{99}	d_{100}	e_{101}	f_{102}	g_{103}	h_{104}	i_{105}	j_{106}	k_{107}	l_{108}	m_{109}	n_{110}	o_{111}	"60
’160	p_{112}	q_{113}	r_{114}	s_{115}	t_{116}	u_{117}	v_{118}	w_{119}	x_{120}	y_{121}	z_{122}	$\imath_{123}$	$\jmath_{124}$	$\wp_{125}$	$\rightarrow_{126}$	$\mathbin{\frown}_{127}$	"70
	"00	"01	"02	"03	"04	"05	"06	"07	"08	"09	"0A	"0B	"0C	"0D	"0E	"0F	

Parametri

inclinazione	0,25000	x-height	4,30554 pt
spazio interparola	0,00000 pt	larghezza del quadrato	10,19999 pt
allungamento interparola	0,00000 pt	spazio extra	0,00000 pt
accorciamento interparola	0,00000 pt	corpo nominale	10,00000 pt

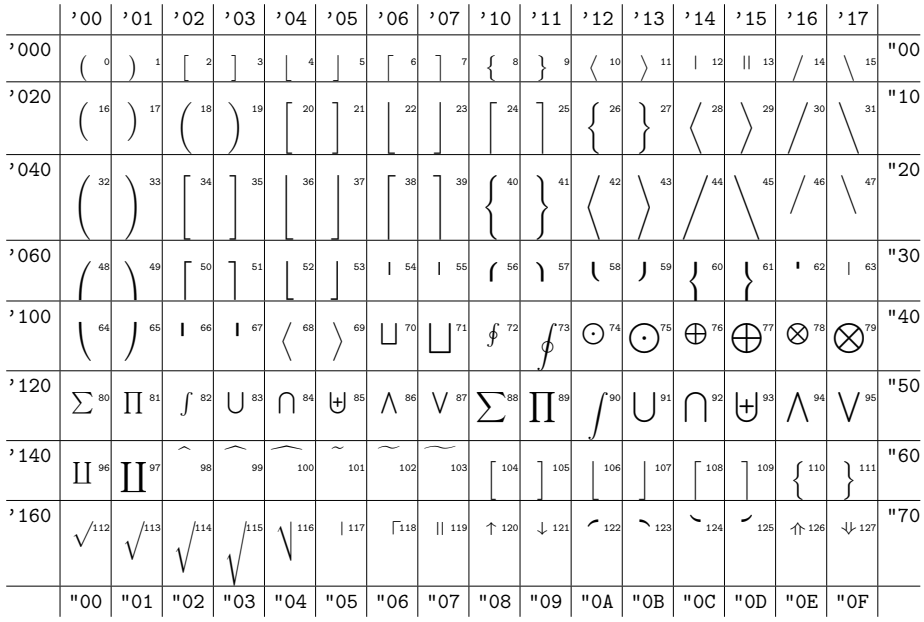
Tabella 18.7: Il corsivo matematico con codifica OML; questa polizza contiene anche le cifre arabe ‘minuscole’.

	’00	’01	’02	’03	’04	’05	’06	’07	’10	’11	’12	’13	’14	’15	’16	’17	
’000	$-_{0}$	$\cdot_{1}$	$\times_{2}$	$*_{3}$	$\div_{4}$	$\diamond_{5}$	$\pm_{6}$	$\mp_{7}$	$\oplus_{8}$	$\ominus_{9}$	$\otimes_{10}$	$\oslash_{11}$	$\odot_{12}$	$\bigcirc_{13}$	$\circ_{14}$	$\bullet_{15}$	"00
’020	$\asymp_{16}$	$\equiv_{17}$	$\subseteq_{18}$	$\supseteq_{19}$	$\leq_{20}$	$\geq_{21}$	$\preceq_{22}$	$\succeq_{23}$	$\sim_{24}$	$\approx_{25}$	$\subset_{26}$	$\supset_{27}$	$\ll_{28}$	$\gg_{29}$	$\prec_{30}$	$\succ_{31}$	"10
’040	$\leftarrow_{32}$	$\rightarrow_{33}$	$\uparrow_{34}$	$\downarrow_{35}$	$\leftrightarrow_{36}$	$\nearrow_{37}$	$\searrow_{38}$	$\simeq_{39}$	$\Leftarrow_{40}$	$\Rightarrow_{41}$	$\Uparrow_{42}$	$\Downarrow_{43}$	$\Leftrightarrow_{44}$	$\nwarrow_{45}$	$\swarrow_{46}$	$\propto_{47}$	"20
’060	$\prime_{48}$	∞_{49}	$\in_{50}$	$\ni_{51}$	$\triangle_{52}$	$\triangledown_{53}$	$/_{54}$	$\mathbin{.}_{55}$	$\forall_{56}$	$\exists_{57}$	$\neg_{58}$	$\emptyset_{59}$	$\Re_{60}$	$\Im_{61}$	$\top_{62}$	$\bot_{63}$	"30
’100	$\aleph_{64}$	$\mathcal{A}_{65}$	$\mathcal{B}_{66}$	$\mathcal{C}_{67}$	$\mathcal{D}_{68}$	$\mathcal{E}_{69}$	$\mathcal{F}_{70}$	$\mathcal{G}_{71}$	$\mathcal{H}_{72}$	$\mathcal{I}_{73}$	$\mathcal{J}_{74}$	$\mathcal{K}_{75}$	$\mathcal{L}_{76}$	$\mathcal{M}_{77}$	$\mathcal{N}_{78}$	$\mathcal{O}_{79}$	"40
’120	$\mathcal{P}_{80}$	$\mathcal{Q}_{81}$	$\mathcal{R}_{82}$	$\mathcal{S}_{83}$	$\mathcal{T}_{84}$	$\mathcal{U}_{85}$	$\mathcal{V}_{86}$	$\mathcal{W}_{87}$	$\mathcal{X}_{88}$	$\mathcal{Y}_{89}$	$\mathcal{Z}_{90}$	$\cup_{91}$	$\cap_{92}$	$\uplus_{93}$	$\wedge_{94}$	$\vee_{95}$	"50
’140	$\vdash_{96}$	$\dashv_{97}$	$\lfloor_{98}$	$\rfloor_{99}$	$\lceil_{100}$	$\rceil_{101}$	$\{_{102}$	$\}_{103}$	$\langle_{104}$	$\rangle_{105}$	$\vert_{106}$	$\Vert_{107}$	$\updownarrow_{108}$	$\Updownarrow_{109}$	$\backslash_{110}$	$\wr_{111}$	"60
’160	$\surd_{112}$	$\amalg_{113}$	∇_{114}	$\int_{115}$	$\sqcup_{116}$	$\sqcap_{117}$	$\sqsubseteq_{118}$	$\sqsupseteq_{119}$	$\S_{120}$	$\dagger_{121}$	$\ddagger_{122}$	$\P_{123}$	$\clubsuit_{124}$	$\diamondsuit_{125}$	$\heartsuit_{126}$	$\spadesuit_{127}$	"70
	"00	"01	"02	"03	"04	"05	"06	"07	"08	"09	"0A	"0B	"0C	"0D	"0E	"0F	

Parametri

inclinazione	0,25000	x-height	4,30554 pt
spazio interparola	0,00000 pt	larghezza del quadrato	10,19999 pt
allungamento interparola	0,00000 pt	spazio extra	0,00000 pt
accorciamento interparola	0,00000 pt	corpo nominale	10,00000 pt

Tabella 18.8: La polizza dei simboli matematici non estensibili con codifica OMS. Questa polizza è usata anche come alfabeto matematico, perché contiene anche le maiuscole calligrafiche.



Parametri

inclinazione	0,00000	x-height	4,30554 pt
spazio interparola	0,00000 pt	larghezza del quadrato	10,19999 pt
allungamento interparola	0,00000 pt	spazio extra	0,00000 pt
accorciamento interparola	0,00000 pt	corpo nominale	10,00000 pt

Tabella 18.9: I segni matematici estensibili con codifica OMX

non deve occuparsi di nulla per gestire questi simboli; al massimo deve ricordarsi di indicare con `\left` i delimitatori di apertura e con `\right` i delimitatori di chiusura. Usando pdftex (che, ricordiamo, include anche le estensioni di etex) come interprete, esiste anche il comando `\middle` per un delimitatore intermedio che viene esteso in verticale nello stesso modo di quando si usa `\left` o `\right` ma con una spaziatura a destra e a sinistra coerente con il significato di delimitatore intermedio.

Solo lualatex e xelatex

Per i programmi xelatex e lualatex i font matematici si impostano caricando il file di estensione *unicode-math* e caricando una famiglia di font matematici OpenType. Il sistema TEX viene distribuito con una certa collezione di questi font matematici OpenType i cui nomi contengono tutti il nome di un font testuale con il suffisso ␣Math. Così si avrà la famiglia XITS Math, Asana Math, Latin Modern Math, eccetera: il sistema TEX e il sistema operativo mettono a disposizione un'altra quindicina di font OpenType matematici. Le impostazioni nel preambolo sono semplicemente le seguenti:

```
\usepackage{amsmath} % quando è necessario
\usepackage{unicode-math}
\setmathfont{⟨famiglia di font matematici OpenType⟩}
```

Si noti che se si vuole usare il file di estensione *amsmath* lo si deve fare prima di caricare *unicode-math*, ma *non si deve mai caricare* il file *amssymb*, perché tutti i simboli matematici sono già forniti dal file *unicode-math* e dal font matematico prescelto. Usando questi programmi l'uso dei font matematici è molto più semplice rispetto a quando si usa pdflatex; quanto descritto qui di seguito riguarda solo quest'ultimo programma.

Solo pdflatex

In matematica la scelta dei font è molto più complessa. Innanzi tutto è possibile, di default, scrivere una formula in caratteri neri, solo se si segue una delle regole seguenti.

1. Prima di entrare in modo matematico si specifica la dichiarazione `\boldmath`; chiuso l'ambiente matematico bisogna poi specificare il comando `\unboldmath`, altrimenti la dichiarazione di comporre in nero rimane valida 'per sempre'; in alternativa la formula da comporre in nero può essere racchiusa, assieme alla dichiarazione `\boldmath`, dentro ad un gruppo, per esempio:

   ```
   {% inizio del gruppo
   \boldmath
   \begin{equation}
   ...
   \end{equation}
   }% fine del gruppo
   ```

2. Si usa il pacchetto *amsmath* con il pacchetto accessorio *amssymb* che mette a disposizione il comando `\boldsymbol` per comporre in nero un simbolo matematico isolatamente; oppure si carica il pacchetto *bm* che mette a disposizione un comando simile, `\bm`, per comporre in nero simboli o sottoespressioni matematiche isolate o, volendo, una intera espressione.

I comandi disponibili fanno riferimento agli alfabeti matematici caricati dalla classe o dai pacchetti richiamati nel preambolo; di default pdflatex carica i font della collezione Computer Modern a 128 caratteri predisposti per la matematica, tabelle 18.7-18.9; questi font obbediscono in generale a codifiche specifiche non usabili nel testo se non con comandi particolari. Per esempio è possibile usare nel testo le cifre arabe *minuscole* 0123456789 attraverso l'uso del comando `\oldstylenums`, sebbene i loro segni derivino dal font destinato alle lettere matematiche.

In particolare ogni classe e/o ogni pacchetto che carichi i font matematici di default, o alternativi a quelli di default, carica quattro famiglie[6] distinte.

[6] Il nome *famiglia* qui è usato in modo un po' diverso da solito; per TeX una famiglia di font matematici è un gruppo di font con lo stesso disegno, ma usabili in tre corpi distinti, quello normale, quello per i pedici e gli apici di primo livello e quello per i pedici e gli apici di secondo

1. La famiglia per gli operatori (tabella 18.3); essa può coincidere con il font tondo con grazie usato per il testo, ma viene usato in modo matematico: non accetta gli accenti testuali e non accetta gli spazi. Esso serve per comporre i più comuni simboli matematici come $+, -, =$ e pochi altri, le cifre arabe *maiuscole* diritte, ma essenzialmente serve per comporre il nome delle funzioni matematiche come 'sin', 'cos', 'log', 'lim', eccetera.
2. La famiglia per le lettere (tabella 18.7); essa contiene l'alfabeto latino e l'alfabeto greco corsivi usati per le variabili matematiche; contiene anche numerosi altri simboli binari o di relazione e le cifre arabe *minuscole*. Il corsivo matematico latino non riconosce le legature, perché ogni lettera rappresenta una variabile e in matematica le variabili moltiplicate fra di loro sono generalmente scritte una dopo l'altra senza segni interposti. Si può scrivere $ff = f^2$ e le due f non devono legarsi nella solita legatura testuale *ff*. Usare questi font per scrivere V_{eff} per indicare la tensione *efficace* produce una composizione orrenda. Si è già detto come scrivere correttamente pedici e/o apici testuali usati come apposizioni delle variabili matematiche.
3. La famiglia per i simboli (tabella 18.8); contiene l'alfabeto maiuscolo *calligrafico* e moltissimi altri segni per gli operatori binari e/o di relazione.
4. La famiglia dei simboli estensibili (tabella 18.9), cioè i segni di radice e di integrale, e le parentesi di vari tipi che cambiano dimensioni a seconda del contenuto su cui devono operare.

Siccome le famiglie di font matematici in senso stretto che pdflatex può usare sono al massimo sedici, non è il caso di caricare altre famiglie senza tenere conto di questo limite; quando si carica direttamente o indirettamente il pacchetto *amsfonts* vengono definite altre due famiglie di segni matematici (tabelle 14.1 e 14.2). È raro raggiungere il limite di 16 famiglie, ma a chi scrive è capitato.

Per inserire variabili composte con altri alfabeti, mediante i comandi `\mathsf` o `\mathtt` o `\mathit`, o simili, pdflatex ricorre a dei comandi che usano i font testuali in modo matematico (senza accenti e senza spazi) ma senza caricare effettivamente altre famiglie di font matematici. I dettagli sono abbastanza delicati, per cui si rimanda alla guida `fntguide.pdf` [1] nella cartella `.../doc/latex/base/`, che dedica una lunga sezione ai comandi per gestire i font matematici. Per vedere in azione quei comandi, si può esaminare il file `fontmath.ltx` nella cartella `.../tex/latex/base/` dove sono usati tutti gli speciali comandi per caricare i font matematici di default.

Il consiglio che si può dare in merito alla gestione dei caratteri matematici è quello di cercare di avere sempre i font testuali e quelli matematici che si accordino fra di loro armoniosamente; come minimo devono avere lo stesso occhio delle lettere minuscole e devono avere la stessa nerezza. I vecchi pacchetti

livello. Questo è uno dei motivi per i quali la gestione dei font matematici è decisamente più elaborata di quella per i font testuali. Con i font matematici OpenType, non si parla nemmeno di famiglie matematiche o gruppi matematici, perché con i font OpenType matematici basta un solo font per contenere tutti i glifi che servono.

txfonts e *pxfonts* e quelli nuovi *newtxtext* con *newtxmath* e *newpxtext* con *newpxmath* caricano correttamente sia i font testuali sia quelli per la matematica. Se invece, per esempio, si volesse usare il font Utopia per il testo, basterebbe usare il pacchetto *utopia*; questo pacchetto, però, imposta solo i font testuali, conservando i font di default per la matematica. Il risultato che si otterrebbe è pessimo: l'occhio dei font Utopia e la loro nerezza contrasta vistosamente con l'occhio e la nerezza dei font Computer Modern, per cui l'uso di quel pacchetto sarebbe accettabile solo per comporre un documento *privo* completamente di matematica.

Il pacchetto *mathdesign*, specificando l'opportuna opzione, permette invece di caricare font testuali e matematici che si accordano fra di loro armoniosamente. Dando il comando

```
\usepackage[utopia]{mathdesign}
```

si possono usare infatti i font Utopia anche per testi contenenti della matematica, perché i font matematici sono scelti in modo da accordarsi con i font per il testo.

Il pacchetto *mathdesign* accetta come opzioni *utopia* (Adobe Utopia), *charter* (Bitstream Charter), *garamond* (URW Garamond), e ne sono previsti alcuni altri. Bisogna usare qualche piccola cautela (per esempio, non si deve contemporaneamente caricare il pacchetto *amsmath* con i suoi font, se non altro perché *mathdesign* ha già ridefinito tutti i comandi) e per questo è necessario leggere con attenzione il file di documentazione `mathdesign-doc.pdf` in `.../doc/latex/mathdesign/`. Anche il pacchetto *fourier* consente di usare il font Utopia con i caratteri matematici congruenti; il file di documentazione `fourier-doc-en.pdf` è in `.../doc/fonts/fourier/`. Anche il pacchetto *kpfonts* permette di usare una ottima imitazione dei font Palladio testuale a cui vengono affiancati dei font matematici disegnati apposta. Il pacchetto *libertinus* permette di usare i font testuali e matematici di una variante Libertinus dei font Linux Libertine. Un pregio unico di questi font è che usa font matematici codificati a 8 bit, quindi con 256 segni in ogni polizza, per cui si possono usare tranquillamente più famiglie (sempre al massimo 16, ma di default se ne usano di meno per la matematica ordinaria) rispetto a quanto si può ottenere con i font matematici "normali".

Con alcune classi di default vengono usati font testuali e matematici che possono venire facilmente modificati e sostituiti con altri. In particolare talvolta viene usato il pacchetto *euler*; il file della documentazione `euler.pdf` si trova in `.../doc/latex/euler/` e va letto attentamente; fra l'altro vi si dice che questo tipo di font matematico si accorda abbastanza bene con diversi font vettoriali, ma bisogna fare attenzione sulla sequenza delle chiamate dei vari pacchetti. Il pacchetto accetta anche diverse opzioni che vanno usate con attenzione. Tuttavia, benché i singoli caratteri siano bellissimi e siano una imitazione (disegnata dal grande Hermann Zapf) dei segni che un matematico con bella calligrafia scriverebbe sulla lavagna, essi sono sostanzialmente diritti e non hanno una forma corsiva o comunque inclinata; per questo motivo essi sono da preferire per comporre un documento di tipo prettamente matematico e non risultano adatti ad un documento scritto da un fisico o da un tecnologo. Naturalmente è una

questione di gusti, ma è opportuno ricordare che le norme ISO per la scrittura della matematica per la fisica e le scienze sperimentali sono piuttosto cogenti e hanno una loro precisa ragion d'essere.

	’00	’01	’02	’03	’04	’05	’06	’07	’10	’11	’12	’13	’14	’15	’16	’17	
’000	` 0	´ 1	ˆ 2	˜ 3	¨ 4	˝ 5	˚ 6	ˇ 7	˘ 8	¯ 9	˙ 10	¸ 11	˛ 12	’ 13	14	15	"00
’020	16	17	" 18	19	20	— 21	— 22	23	← 24	→ 25	‿ 26	‿ 27	‿ 28	‿ 29	30	31	"10
’040	ƀ 32	33	34	35	$ 36	37	38	’ 39	40	41	* 42	43	, 44	= 45	. 46	/ 47	"20
’060	o 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	58	59	⟨ 60	— 61	⟩ 62	63	"30
’100	64	65	66	67	68	69	70	71	72	73	74	75	76	℧ 77	78	◯ 79	"40
’120	80	81	82	83	84	85	86	Ω 87	88	89	90	⟦ 91	92	⟧ 93	↑ 94	↓ 95	"50
’140	` 96	97	⋆ 98	o|o 99	† 100	101	102	103	104	105	106	107	108	∞ 109	♩ 110	111	"60
’160	112	113	114	115	116	117	118	119	120	121	122	123	124	125	~ 126	= 127	"70
’200	◡ 128	◡ 129	″ 130	‟ 131	† 132	‡ 133	‖ 134	‰ 135	• 136	°C 137	$ 138	¢ 139	f 140	₡ 141	₩ 142	₦ 143	"80
’220	₲ 144	₱ 145	£ 146	₨ 147	? 148	¡ 149	₫ 150	™ 151	‰ 152	¶ 153	฿ 154	№ 155	℅ 156	℮ 157	○ 158	℠ 159	"90
’240	{ 160	} 161	¢ 162	£ 163	¤ 164	¥ 165	¦ 166	§ 167	¨ 168	Ⓒ 169	ª 170	⊃ 171	¬ 172	Ⓟ 173	Ⓡ 174	¯ 175	"A0
’260	° 176	± 177	2 178	3 179	´ 180	µ 181	¶ 182	· 183	※ 184	1 185	º 186	√ 187	1/4 188	1/2 189	3/4 190	€ 191	"B0
’300	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	"C0
’320	208	209	210	211	212	213	× 214	215	216	217	218	219	220	221	222	223	"D0
’340	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	"E0
’360	240	241	242	243	244	245	÷ 246	247	248	249	250	251	252	253	254	255	"F0
	"00	"01	"02	"03	"04	"05	"06	"07	"08	"09	"0A	"0B	"0C	"0D	"0E	"0F	

Parametri

inclinazione	0,00000	x-height	4,30450 pt
spazio interparola	3,33251 pt	larghezza del quadrato	9,99755 pt
allungamento interparola	1,66625 pt	spazio extra	1,11083 pt
accorciamento interparola	1,11083 pt	corpo nominale	10,00000 pt

Tabella 18.10: Il Text Companion Font con codifica TS1

18.6 Il Text Companion Font

LaTeX consente di usare un numero enorme di simboli, specialmente in matematica; ma da solo LaTeX non offre i simboli che si usano nel testo, dai segni delle unità monetarie ad altri segni che possono trovare posto in scritti di vario genere.

Così quando furono prodotti i primi font latini con la codifica T1, fu sentita immediatamente la necessità di un certo numero di segni o comandi non coperti con i simboli di nessun altro font, e fu predisposto il Text Companion Font, i cui simboli sono riportati nella tabella 18.10; ritorna per l'ennesima volta il segno del mho, ℧, il cui unico pregio è quello di poter essere scritto in modo testo con il comando `\textmho`. Ahimè, trattandosi di un segno bandito dalle norme ISO,

questo segno appare per la terza volta, ma inutilmente, perché non può essere usato.[7]

Il Text Companion Font veniva caricato e reso utilizzabile invocando il pacchetto *textcomp*. Le recenti modifiche (vedi il capitolo **??**) al nucleo di LaTeX rendono superfluo caricare questo pacchetto, perché tutti i comandi, definiti da *textcomp*, vi sono migrati dentro. Anche se si usa il pacchetto *mathdesign* è superfluo caricare il pacchetto *textcomp*, perché tutti i simboli disponibili con i font Utopia, Charter e Garamond sono già tutti definiti.

18.7 Gli alfabeti diversi da quello latino

Solo pdflatex

Per scrivere in altre lingue con pdflatex bisogna avere invocato il pacchetto `babel` avendo indicato fra le opzioni le lingue che si vogliono usare; l'ultima lingua indicata nell'elenco delle opzioni diventa la lingua di default per l'intero documento. Dalla versione 3.9 di *babel*, la lingua principale può venire marcata con l'attributo *main* e in questo caso può apparire nella lista delle lingue in una posizione qualsiasi.

Quando si invoca una qualsiasi lingua che si scriva con l'alfabeto cirillico (russo, bulgaro, ucraino, eccetera) il pacchetto `babel` imposta direttamente anche l'uso dell'alfabeto cirillico. Una versione a 128 caratteri è riportata nella tabella 18.4, però con alcune di quelle lingue può venire scelta un'altra codifica, magari a 256 caratteri.

18.7.1 Caratteri cirillici

Solo pdflatex

Come già detto i font cirillici sono solitamente preinstallati in ogni distribuzione non minimale del sistema TeX. Le codifiche cirilliche, oltre alla già citata *OT2* a 128 caratteri, sono tre diverse versioni a 256 caratteri: *T2A*, *T2B* e *T2C*, e infine la codifica *X2*.

In che cosa differiscono queste codifiche? Nel fatto che la codifica a 128 caratteri richiede espressamente di essere attivata quando si usa il cirillico e permette di scrivere *esclusivamente* in russo. Le codifiche *T2x* hanno i primi 128 caratteri coincidenti con quelli della codifica *T1*, quindi sono caratteri latini, mentre i secondi 128 caratteri sono quelli cirillici validi per scrivere in russo, anche con ortografie 'prerivoluzionarie'. Ma la variante *A* serve per scrivere oltre che in caratteri latini anche con il cirillico usato nei paesi dell'Europa orientale (Ucraina, Bulgaria, Russia e altre). La variante *B* serve per comporre con i segni cirillici usati nelle repubbliche della Siberia Orientale. La variante *C* serve per scrivere in abcazio.

È chiaro che queste tre codifiche permettono di scrivere contemporaneamente nelle lingue dell'Europa occidentale e in una o l'altra delle lingue dell'Europa

[7] In realtà è ancora usato negli Stati Uniti dove le norme ISO hanno meno seguito che in Europa. In Europa, invece, non dovrebbe più esserci nessuno che lo usi.

	’00	’01	’02	’03	’04	’05	’06	’07	’10	’11	’12	’13	’14	’15	’16	’17	
’000	` 0	´ 1	ˆ 2	˜ 3	¨ 4	˝ 5	° 6	ˇ 7	˘ 8	¯ 9	˙ 10	¸ 11	˛ 12	I 13	⟨ 14	⟩ 15	"00
’020	“ 16	” 17	‘ 18	’ 19	„ 20	– 21	— 22	23	◦ 24	‵ 25	´ 26	¨ 27	δ 28	δ 29	Ӈ 30	ӈ 31	"10
’040	␣ 32	! 33	" 34	# 35	$ 36	% 37	& 38	’ 39	(40	) 41	* 42	+ 43	, 44	- 45	· 46	/ 47	"20
’060	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	: 58	; 59	< 60	= 61	> 62	? 63	"30
’100	@ 64	Æ 65	Ъ 66	Ћ 67	Ҽ 68	Ҿ 69	К 70	Ќ 71	Ӆ 72	I 73	J 74	Љ 75	М 76	Њ 77	Ԛ 78	Ԝ 79	"40
’120	Ҏ 80	Q 81	Ҭ 82	S 83	Ҵ 84	Ц 85	Ч 86	W 87	Ѣ 88	Ӂ 89	V 90	[91	\ 92	] 93	^ 94	_ 95	"50
’140	‘ 96	æ 97	ђ 98	ћ 99	ҽ 100	ҿ 101	к 102	ќ 103	ӆ 104	i 105	j 106	љ 107	м 108	њ 109	ԛ 110	ԝ 111	"60
’160	ҏ 112	q 113	ҭ 114	s 115	ҵ 116	ц 117	ч 118	w 119	ѣ 120	ӂ 121	v 122	{ 123		124	} 125	~ 126	- 127	"70
’200	Ґ 128	Ғ 129	Ӷ 130	Ҕ 131	Һ 132	Ӝ 133	Ҙ 134	Ӟ 135	Ї 136	Қ 137	Ҟ 138	Ҝ 139	Ӆ 140	Ӊ 141	Ҥ 142	Ӈ 143	"80
’220	Ө 144	Ҫ 145	Ў 146	Ү 147	Ұ 148	Х 149	Ҳ 150	Ҷ 151	Ҷ 152	Ҽ 153	Ә 154	Ҿ 155	Ё 156	№ 157	¤ 158	§ 159	"90
’240	ґ 160	ғ 161	ӷ 162	ҕ 163	h 164	ж 165	ҙ 166	ӟ 167	ї 168	қ 169	ҟ 170	ҝ 171	ӆ 172	ӊ 173	ҥ 174	ӈ 175	"A0
’260	ө 176	ҫ 177	ў 178	ү 179	ұ 180	х 181	ҳ 182	ҷ 183	ч 184	ҽ 185	ә 186	ҿ 187	ё 188	„ 189	« 190	» 191	"B0
’300	А 192	Б 193	В 194	Г 195	Д 196	Е 197	Ж 198	З 199	И 200	Й 201	К 202	Л 203	М 204	Н 205	О 206	П 207	"C0
’320	Р 208	С 209	Т 210	У 211	Ф 212	Х 213	Ц 214	Ч 215	Ш 216	Щ 217	Ъ 218	Ы 219	Ь 220	Э 221	Ю 222	Я 223	"D0
’340	а 224	б 225	в 226	г 227	д 228	е 229	ж 230	з 231	и 232	й 233	к 234	л 235	м 236	н 237	о 238	п 239	"E0
’360	р 240	с 241	т 242	у 243	ф 244	х 245	ц 246	ч 247	ш 248	щ 249	ъ 250	ы 251	ь 252	э 253	ю 254	я 255	"F0
	"00	"01	"02	"03	"04	"05	"06	"07	"08	"09	"0A	"0B	"0C	"0D	"0E	"0F	

Font parameters

inclinazione	0.0pt	x-height	4.3045pt
spazio interparola	3.33252pt	larghezza del quadrato	9.99756pt
allungamento interparola	1.66626pt	spazio extra	1.11084pt
accorciamento interparola	1.11084pt	corpo nominale	10.00000 pt

Tabella 18.11: Il font cirillico esteso con la codifica *X2*

orientale e dell’ex Unione Sovietica. Non tutte le collezioni di font, però, contengono sia il latino sia il cirillico dell’una o dell’altra specie e/o sono disponibili sia in forma vettoriale, come sarebbe desiderabile, sia in forma a matrice di punti.

Per scrivere testi cirillici senza i vincoli delle polizze e delle codifiche appena viste si può usare la codifica *X2*, la cui matrice di segni è mostrata nella tabella 18.11. Si noti che la distribuzione normale del sistema TEX, almeno fino alla distribuzione di TEX Live 2009, conteneva questi font adatti alla codifica *X2* solo a matrici di punti, non in formato scalabile. Se si fosse dovuto avere bisogno di questi font, non sarebbe stato un problema trasformarli in formato scalabile usando la procedura esemplificata nel paragrafo 18.8.4; con la distribuzione TEX Live 2010 non è più necessario ricorrere a questa trasformazione.

Va notata infine una cosa: i font cirillici con codifica *OT2* sono disponibili sia a matrici di punti sia in formato scalabile; questi font sono adatti a scrivere parole russe semplicemente cambiando codifica e impedendo la sillabazione; una tastiera latina è perfettamente in grado di immettere il testo russo ricorrendo alle sole lettere latine, un po’ come è stato descritto nel paragrafo 15.2 per scrivere in greco: in altre parole la disposizione dei segni cirillici nella tabella 18.4 e un set di legature permette di usare i segni latini e la traslitterazione latina del russo secondo la norma anglosassone, cosicché non è affatto difficile inserire qualche parola russa all’interno del testo scritto in caratteri latini. Per esempio, dopo

aver definito un comando `\textcyrillic` della forma[8]:

```
\newcommand\cryllictext{\hyphenrules{nohyphenation}%
\fontencoding{OT2}\fontfamily{cmr}\selectfont}
\DeclareTextFontCommand{\textcyrillic}{\cyrillictext}
```

si può scrivere `\textcyrillic{Khrushchev}` dentro un testo scritto in lettere latine per ottenere Хрущев. Con le altre codifiche si possono fare cose analoghe; in particolare è possibile usare la codifica di entrata *utf8* e, disponendo di un editor adeguato per l'input dei caratteri cirillici, si può scrivere direttamente in cirillico con la certezza che quel testo viene composto in caratteri cirillici e il testo 'latino' in caratteri latini. Si può approfondire l'argomento nel bell'articolo di Enrico Gregorio che espone tutte le procedure nei minimi dettagli e mostra anche le soluzioni a qualche piccolo problema che si può manifestare, [28].

Per scrivere in russo, bulgaro, ucraino, eccetera, è meglio specificare la lingua come opzione a *babel*; così si dispone direttamente della sillabazione specifica della lingua, delle parole codificate per indicare l'equivalente di 'Capitolo', 'Figura', 'Tabella', eccetera, e per disporre dei comandi specifici per introdurre certi segni cirillici nel testo. Per la lingua russa, sono utilizzabili tutte le codifiche disponibili con il sistema TeX, ma quella suggerita è costituita dalla codifica *T2A*, sebbene quella di default, sembra di capire, sia la codifica *X2*. In ogni caso ognuna ha i propri vantaggi; la codifica suggerita è particolarmente utile per gli utenti russi che operano con tastiere russe, nelle quali con un semplice click si passa dai caratteri latini a quelli cirillici (e viceversa). Cosicché l'eventuale cambio di lingua avviene al livello della tastiera. Infatti sopra si è spiegato che i primi 128 caratteri della codifica *T2A* coincidono con i caratteri latini secondo la codifica ASCII (a 7 bit), mentre i secondi 128 caratteri contengono i caratteri cirillici validi per il russo; quindi comporre in modo misto non coinvolge il cambio del font di uscita. Ovviamente i caratteri ASCII non sono raccomandabili altro che per scrivere in inglese, forse l'unica lingua moderna occidentale priva di accenti.

lualatex e xelatex

Usando i programmi xelatex o lualatex basta impostare una qualunque lingua slava o asiatica scritta in cirillico come lingua principale o secondaria e definire una famiglia di font col nome *cyrillicfont* per scegliere un font OpenType che contenga il cirillico, e qualunque lingua da scriversi con quell'alfabeto viene composta con il font selezionato per il cirillico; gli ambienti *russian*, *bulgarian*, *ukranian*, eccetera sono già forniti da polyglossia in corrispondenza con le lingue specificate. Tutto ciò è molto più semplice rispetto a quando si usa pdflatex.

18.7.2 Caratteri greci

Solo pdflatex

Quando usando pdflatex si invoca invece la lingua greca senza ulteriori specifica-

[8] Nel testo [28] lo stesso comando è definito in maniera molto più raffinata, nel senso che prende in considerazione la possibilità che la lingua *nohyphenation* non sia disponibile; qui il comando è definito supponendo di disporre di una distribuzione sufficientemente moderna del sistema TeX.

zioni, viene anche impostato l'alfabeto greco riportato nella tabella 18.12 ma predisposto per la scrittura monotonica moderna, cioè con gli accenti ridotti al solo accento acuto nelle parole polisillabiche e dove sono eliminati gli spiriti; con queste semplificazioni diventa più frequente l'uso della dieresi per indicare gruppi di vocali che non formano dittonghi. Se si specifica invece l'attributo/modificatore `polutoniko` o l'attributo/modificatore `ancient` per la lingua greca diventano disponibili tutti i segni indicati nella tabella 18.12. Per usare il greco bisogna leggere attentamente la documentazione `babel` relativa a questa lingua per usare correttamente la transcodifica della tastiera USA (o anche italiana) nei caratteri greci; si noti che per la moltitudine di accenti e altri segni diacritici presenti con molte vocali, sono previste numerose legature; così la sequenza `>~a|` produce il segno ᾷ.

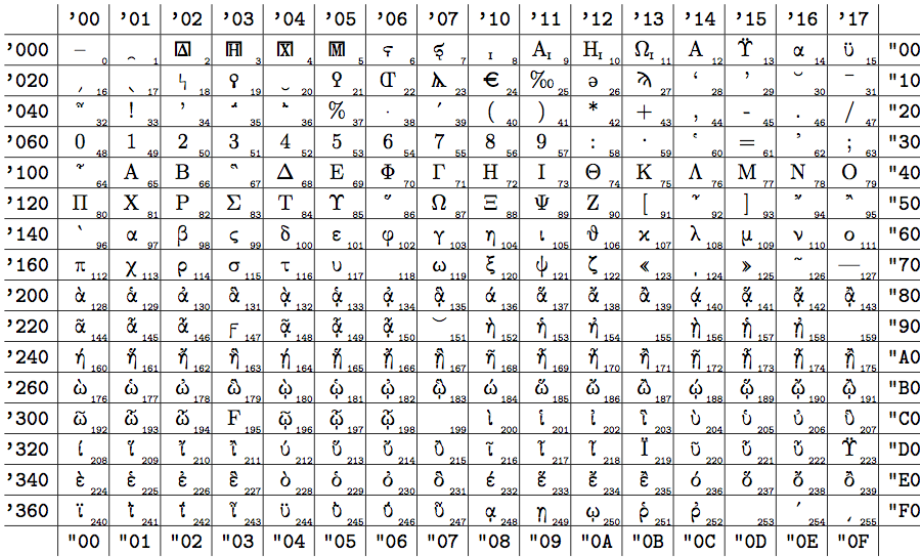
Pur essendo disponibili tutti i comandi di *babel* per comporre brani di testo in una lingua differente da quella principale, per il greco esistono diversi pacchetti che consentono di fare cose simili anche senza avere invocato la lingua greca. Un pacchettino molto utile (che però richiede la specificazione della lingua greca,) è *begingreek* che definisce l'ambiente *greek* per scrivere un brano di testo più o meno lungo in greco, oppure il comando `\greektxt` per scrivere una parola o una breve locuzione; entrambi l'ambiente e il comando accettano un argomento facoltativo per specificare la famiglia di font greci che si preferisce usare; per esempio si può scrivere nel file sorgente

```
\greektxt[artemisia]{t\'eqnh}
```

per ottenere τέχνη, quando senza l'argomento facoltativo si otterrebbe τέχνη. Per ulteriori informazioni circa l'uso normale dei font greci si veda il paragrafo 15.2.

Per produrre testi greci finemente commentati con le notazioni filologiche del settore si può estendere la funzionalità di LaTeX con il pacchetto *teubner* che consente di introdurre una grande varietà di segni specifici, di usare una serie di font greci inclinati particolarmente leggibili, sviluppati preso la tipografia Teubner di Lipsia e in onore della quale è stato scelto il nome del pacchetto. Tuttavia va fatto osservare che pacchetti di vario genere possono collidere; in particolare il pacchetto *teubner* inizialmente collideva con i pacchetti *amsmath* e compagni; questo comportamento è stato corretto nel febbraio 2008 grazie alle indicazioni di un utente. È vero che difficilmente in un lavoro di filologia classica greca c'è bisogno di usare matematica avanzata, tuttavia questo problema esiste e in questi casi l'utente che si scontra con problemi derivanti dal conflitto di pacchetti dovrebbe avere 'il senso civico' di avvisare i curatori dei pacchetti, come avviene regolarmente con tutto il software libero, il cui progresso avviene proprio grazie a questa interazione fra programmatori e utenti.

Recentemente il pacchetto *teubner* è stato arricchito di altre funzionalità che permettono di comporre i testi filologici in modo tipograficamente migliore. In particolare diversi segni, che era stato necessario creare con soluzioni non molto azzeccate, sono stati corretti sfruttando le proprietà grafiche del motore di composizione pdflatex; allo stesso tempo le crenature dei segni accentati che con le macro precedenti richiedevano non poca ingegnosità per essere applicate



Parametri			
inclinazione	0,00000	x-height	4,30450 pt
spazio interparola	3,33251 pt	larghezza del quadrato	9,99755 pt
allungamento interparola	1,66625 pt	spazio extra	1,11083 pt
accorciamento interparola	1,11083 pt	corpo nominale	10,00000 pt

Tabella 18.12: Il font greco a 256 caratteri con codifica LGR

correttamente, ora possono essere affrontate con macro che sfruttano meglio i comandi interni di pdflatex. Speriamo dunque che i filologi si possano trovare meglio con la nuova versione.

Il pacchetto *teubner* per il momento non funziona ancora per lavorare con xelatex e i font OpentType.

Solo xelatex

Usando xelatex non ci sono differenze rispetto all'uso di altre lingue; basta, dopo aver caricato *polyglossia* specificare la lingua greca come lingua principale o come lingua secondaria, ricordando di specificare la variante fra *mono*, *poly* o *ancient*, tenendo conto che la mancanza della variante implica l'uso dell'ortografia e della sillabazione monotonica. Conviene anche definire dopo avere caricato *fontspec* una famiglia di font OpenType che contenga il greco; specificandone il nome mediante l'istruzione `greekfont`, cosicché quando si vuole scrivere in greco, in particolare con l'ambiente greek definito da *polyglossia*, viene automaticamente selezionato quel font.

Vale la pena segnalare che la lingua greca può essere specificata con varianti diverse lungo lo sviluppo del documento da comporre; questa probabilmente è una funzionalità interessante, per esempio, per chi scrive in greco moderno per riportare brani di greco antico.

Solo lualatex

Usando lualatex si procede come quando si usa xelatex con una sola sostanziale differenza; infatti lualatex, per impostazione predefinita, carica per ora un solo file di pattern di sillabazione per una data lingua, perché questo programma carica questi file in tempo reale sulla base delle lingue usate in un dato documento; invece xelatex richiede che tutti i pattern di sillabazione siano precaricati nel file di formato. Ci sono vantaggi e svantaggi nei due approcci, ma in questo caso lualatex è svantaggiato per lavorare con il greco e le sue tre varianti che implicano tre differenti file di pattern di sillabazione. Per questo lualatex carica soltanto i pattern per il greco monotonico; nessuno impedisce di scrivere il testo sorgente anche o solo in greco politonico o in greco antico, ma la sillabazione sarà inevitabilmente carente o errata. Dal 2018 questo problema non dovrebbe essere stato eliminato.

Ovviamente non si può ancora usare il pacchetto *teubner* per l'impossibilità di questo di lavorare con i font OpenType e per il problema della sillabazione monotonica con vecchie versioni del sistema TEX. Sembra che ora si stia lavorando per estendere il pacchetto *teubner* per renderlo usabile anche con questi programmi di composizione.

18.7.3 Scrivere con altri alfabeti

È interessante vedere come il sistema TEX disponga di una miriade di caratteri anche per lingue insolite e consenta di comporre in lingue che fanno uso di questi alfabeti.

Nella figura 18.7 vi è rappresentato il Credo cristiano come stabilito nel Concilio di Nicea; esso è composto in latino, in greco e in copto nella varietà bohairica, quella che viene usata nei testi liturgici della chiesa copta. Il latino accentato è la normale versione del latino ecclesiastico per il quale esiste il pacchetto *ecclesiastic* (o il modificatore/variante *ecclesiastic* per il modulo *latin* per *babel* e *polyglossia*). Per il greco si è già detto. Per il copto esiste il pacchetto *coptic* che però consente di comporre quella lingua nella varietà sahidica; il pacchetto *bohairic* esiste ma oggi (2017) non è ancora disponibile per la generalità degli utenti. Le due versioni differiscono solo per la presenza delle lettere accentate, che per la liturgia copta svolgono più o meno la stessa funzione delle lettere accentate usate per il latino ecclesiastico.

Non si ritiene opportuno di riportare il codice sorgente, perché sarebbe molto complicato e di scarsissimo interesse per la maggior parte dei lettori. Inoltre si potrebbero portare esempi anche con scritti in arabo o in ebraico; si potrebbero portare esempi con brevi testi in cinese o giapponese o coreano; in lingue del tutto esotiche come la lingua inuhit degli eschimesi, e tanto altro. Ogni lingua dispone o può disporre di font adatti alla sua composizione, e questa è una delle forze del sistema TEX e dei suoi vari motori di composizione.

Si tenga presente che il sistema TEX con i suoi diversi motori di composizione permette di comporre testi in un centinaio di lingue. Per ciascuna di solito dispone di font adeguati e di moduli per *babel* o *polyglossia* che provvedono ai font necessari per comporre in ciascuna di quelle lingue. Ovviamente ciò

Il Credo di Nicea in tre lingue

Credo in unum Deum, Patrem omnipoténtem Factórem cæli et terræ, visibílium ómnium et invisibílium.

Et in unum Dóminum Iesum Christum, Fílium Dei Unigénitum, et ex Patre natum ante ómnia sæcula. Deum de Deo, lumen de lúmine, Deum verum de Deo vero, génitum, non factum, consubstantiálem Patri: per quem ómnia facta sunt.

Qui propter nos hómines et propter nostram salútem descéndit de cælis. Et incarnátus est de Spíritu Sancto ex María Vírgine, et homo factus est.

Crucifíxus étiam pro nobis sub Póntio Piláto; passus, et sepúltus est, et resurréxit tértia die, secúndum Scriptúras, et ascéndit in cælum, sedet ad déxteram Patris. Et íterum ventúrus est cum glória, iudicáre vivos et mórtuos, cuius regni non erit finis.

Et in Spíritum Sanctum, Dóminum et vivificántem: qui ex Patre (Filióque) procédit. Qui cum Patre et Fílio simul adorátur et conglorificátur: qui locútus est per prophétas. Et unam, sanctam, cathólicam et apostólicam Ecclésiam. Confíteor unum baptísma in remissiínem peccatorum. Et expecto resurrectionem mortuorum, et vitam ventúri sæculi. Amen.

Πιστεύομεν εἰς ἕνα Θεόν, Πατέρα, Παντοκράτορα, ποιητὴν οὐρανοῦ καὶ γῆς, ὁρατῶν τε πάντων καὶ ἀοράτων.

Καὶ εἰς ἕνα Κύριον Ἰησοῦν Χριστόν, τὸν Υἱὸν τοῦ Θεοῦ τὸν μονογενῆ, τὸν ἐκ τοῦ Πατρὸς γεννηθέντα πρὸ πάντων τῶν αἰώνων· φῶς ἐκ φωτός, Θεὸν ἀληθινὸν ἐκ Θεοῦ ἀληθινοῦ, γεννηθέντα οὐ ποιηθέντα, ὁμοούσιον τῷ Πατρί, δι᾽ οὗ τὰ πάντα ἐγένετο.

Τὸν δι᾽ ἡμᾶς τοὺς ἀνθρώπους καὶ διὰ τὴν ἡμετέραν σωτηρίαν κατελθόντα ἐκ τῶν οὐρανῶν καὶ σαρκωθέντα ἐκ Πνεύματος Ἁγίου καὶ Μαρίας τῆς Παρθένου καὶ ἐνανθρωπήσαντα.

Σταυρωθέντα τε ὑπὲρ ἡμῶν ἐπὶ Ποντίου Πιλάτου, καὶ παθόντα καὶ ταφέντα. Καὶ ἀναστάντα τῇ τρίτῃ ἡμέρα κατὰ τὰς Γραφάς. Καὶ ἀνελθόντα εἰς τοὺς οὐρανοὺς καὶ καθεζόμενον ἐκ δεξιῶν τοῦ Πατρός. Καὶ πάλιν ἐρχόμενον μετὰ δόξης κρῖναι ζῶντας καὶ νεκρούς, οὗ τῆς βασιλείας οὐκ ἔσται τέλος.

Καὶ εἰς τὸ Πνεῦμα τὸ Ἅγιον, τὸ κύριον, τὸ ζωοποιόν, τὸ ἐκ τοῦ Πατρὸς ἐκπορευόμενον, τὸ σὺν Πατρὶ καὶ Υἱῷ συμπροσκυνούμενον καὶ συνδοξαζόμενον, τὸ λαλῆσαν διὰ τῶν προφητῶν. Εἰς μίαν, Ἁγίαν, Καθολικὴν καὶ Ἀποστολικὴν Ἐκκλησίαν. Ὁμολογῶ ἓν βάπτισμα εἰς ἄφεσιν ἁμαρτιῶν. Προσδοκῶ ἀνάστασιν νεκρῶν. Καὶ ζωὴν τοῦ μέλλοντος αἰῶνος. Ἀμήν.

Ⲡⲉⲛⲛⲁϩ ⲉⲟⲩⲛⲟⲩϯ ⲛⲟⲩⲱⲧ: Ⲫⲛⲟⲩϯ Ⲫⲓⲱⲧ ⲡⲓⲡⲁⲛⲧⲟⲕⲣⲁⲧⲱⲣ: ⲫⲏⲧⲁϥϣⲓⲇ ⲛ̀ⲧ̀ⲫⲉ ⲛⲉⲙ ⲛ̀ⲕⲁϩⲓ: ⲛⲏⲉⲧⲟⲩⲛⲁϩ ⲉ̀ⲣⲱⲟⲩ ⲛⲉⲙ ⲛⲏⲉⲧⲉ ⲛ̀ⲥⲉⲛⲁϩ ⲉ̀ⲣⲱⲟⲩ ⲁⲛ.

Ⲡⲉⲛⲛⲁϩ ⲉⲟⲩⲃⲟⲓⲥ ⲛⲟⲩⲱⲧ Ⲓⲏⲥⲟⲩⲥ Ⲡⲓⲭ̀ⲣⲓⲥⲧⲟⲥ Ⲡ̀ϣⲏⲣⲓ ⲛ̀Ⲫⲛⲟⲩϯ ⲡⲓⲙⲟⲛⲟⲅⲉⲛⲏⲥ: ⲡⲓⲙⲓⲥⲓ ⲉ̀ⲃⲟⲗϩⲉⲛ Ⲫⲓⲱⲧ ϩⲁϫⲱⲟⲩ ⲛ̀ⲛⲓⲉⲱⲛ ⲧⲏⲣⲟⲩ: ⲟⲩⲟⲩⲱⲓⲛⲓ ⲉ̀ⲃⲟⲗϩⲉⲛ ⲟⲩⲟⲩⲱⲓⲛⲓ: ⲟⲩⲛⲟⲩϯ ⲛ̀ⲧⲁⲫ̀ⲱⲙⲓ ⲉ̀ⲃⲟⲗϩⲉⲛ ⲟⲩⲛⲟⲩϯ ⲛ̀ⲧⲁⲫ̀ⲱⲙⲓ: ⲟⲩⲙⲓⲥⲓ ⲡⲉ ⲟⲩⲑⲁⲙⲓⲇ ⲁⲛ ⲡⲉ: ⲟⲩⲙⲟⲟⲩⲥⲓⲟⲥ ⲡⲉ ⲛⲉⲙ Ⲫⲓⲱⲧ: ⲫⲏⲧⲁ ϩⲱⲃ ⲛⲓⲃⲉⲛ ⲱⲱⲡⲓ ⲉ̀ⲃⲟⲗϩⲓⲧⲟⲧⲕ.

Ⲫⲁⲓ ⲉ̀ⲧⲉ ⲉⲑⲃⲏⲧⲉⲛ ⲗⲁⲛⲟⲛ ϩⲁ ⲛⲓⲣⲱⲙⲓ ⲛⲉⲙ ⲉⲑⲃⲉ ⲡⲉⲛⲟⲩϫⲁⲓ: ⲁϥⲓ̀ ⲉ̀ⲡⲉⲥⲏⲧ ⲉ̀ⲃⲟⲗϩⲉⲛ ⲧ̀ⲫⲉ: ⲁϥⲃⲓⲥⲁⲣⲝ ⲉ̀ⲃⲟⲗϩⲉⲛ Ⲡⲓⲡ̀ⲛⲉⲩⲙⲁ ⲉⲑⲟⲩⲁⲃ ⲛⲉⲙ ⲉ̀ⲃⲟⲗϩⲉⲛ Ⲩⲁⲣⲓⲁ ⲧ̀ⲡⲁⲣⲑⲉⲛⲟⲥ ⲟⲩⲟⲩ ⲁϥⲉ̀ⲣⲣⲱⲙⲓ.

Ⲟⲩⲟⲩ ⲁⲩⲉⲣⲥ̀ⲧⲁⲩⲣⲱⲛⲓⲛ ⲁ̀ⲙⲟⲩ ⲉ̀ⲣⲡⲓ ⲉ̀ϫⲱⲛ ⲛⲁϩⲣⲉⲛ Ⲡⲟⲛⲧⲓⲟⲥ Ⲡⲓⲗⲁⲧⲟⲥ: ⲁϥⲱⲉⲡⲉⲙⲕⲁϩ ⲟⲩⲟⲩ ⲁⲩⲕⲟⲥⲙ. Ⲟⲩⲟⲩ ⲁϥⲧⲱⲛϥ ⲉ̀ⲃⲟⲗϩⲉⲛ ⲛⲏⲉⲑⲙⲱⲟⲩⲧ ϩⲉⲛ ⲡⲓⲉ̀ϩⲟⲟⲩ ⲁ̀ⲙⲁϩⲱⲟⲩⲧ ⲕⲁⲧⲁ ⲛⲓ̀ϩⲣⲁⲫⲏ. Ⲁϥϣⲉⲛⲁϥ ⲉ̀ⲡ̀ϣⲱⲓ ⲉ̀ⲛⲓⲫⲏⲙⲓ: ⲁϥⲉⲙⲓ ⲥⲁⲟⲩⲓ̀ⲛⲁⲓ ⲁ̀ⲡⲉϥⲱⲧ. Ⲕⲉ ⲡⲁⲗⲓⲛ ⲉ̀ⲛⲛⲟⲩ ϩⲉⲛ ⲡⲉϥ̀ⲱⲟⲩ ⲉ̀ϯϩⲁⲡ ⲉ̀ⲛⲛⲉⲧⲟⲛϩ ⲛⲉⲙ ⲛⲏⲉⲑⲙⲱⲟⲩⲧ: ϥⲏⲧⲉ ⲧⲉϥⲙⲉⲧⲟⲩⲣⲟ ⲟⲩⲁⲑⲙⲟⲩⲧⲛⲕ ⲧⲉ.

Ⲥⲉ ⲧⲉⲛⲛⲁϩ ⲉⲡⲓⲡ̀ⲛⲉⲩⲙⲁ ⲉⲑⲟⲩⲁⲃ: Ⲡ̀ⲃⲟⲓⲥ ⲛ̀ⲣⲉϥϯ ⲛ̀ϩⲱⲛϩ: ϥⲏⲉⲛⲛⲟⲩ ⲉ̀ⲃⲟⲗϩⲉⲛ Ⲫⲓⲱⲧ: ⲥⲉⲟⲩⲱⲱⲡ ⲁ̀ⲙⲟⲩ ⲥⲉϯ̀ⲱⲟⲩ ⲛⲁϥ ⲛⲉⲙ Ⲫⲓⲱⲧ ⲛⲉⲙ Ⲡ̀ϣⲏⲣⲓ: ϥⲏⲧⲁϥⲥⲁⲥⲓ ϩⲉⲛ ⲛⲓⲡ̀ⲣⲟϥⲏⲧⲏⲥ. Ⲉ̀ⲟⲩⲛ̀ ⲛⲁⲗⲏⲓⲁ ⲛ̀ⲕⲁⲑⲟⲗⲓⲕⲏ ⲛⲁ̀ⲡⲟⲥⲧⲟⲗⲓⲕⲏ ⲛⲉⲕⲕⲗⲏⲥⲓⲁ. Ⲧⲉⲛⲉⲣⲑⲟⲙⲟⲗⲟⲅⲓⲛ ⲛⲟⲩⲱⲙⲥ ⲛⲟⲩⲱⲧ ⲉ̀ⲡ̀ϫⲱ ⲉ̀ⲃⲟⲗ ⲛ̀ⲧⲉ ⲛⲓⲛⲟⲃⲓ. Ⲧⲉⲛⲍⲟⲩⲱⲧ ⲉ̀ⲃⲟⲗ ϩⲁ ⲧ̀ϩⲏ ⲛ̀ⲧⲁⲛⲁⲥⲧⲁⲥⲓⲥ ⲛ̀ⲧⲉ ⲛⲓⲣⲉϥⲙⲱⲟⲩⲧ: ⲛⲉⲙ ⲡⲓⲱⲛϩ ⲛ̀ⲧⲉ ⲡⲓⲉ̀ⲱⲛ ⲉⲑⲛⲛⲟⲩ: ⲁ̀ⲙⲏⲛ.

Figura 18.7: Il Credo di Nicea in tre lingue che fanno uso di alfabeti diversi.

non impedisce agli utenti di scegliere font diversi da quelli preimpostati; questa operazione di solito non è facilissima con pdflatex ma è quasi banale con xelatex e lualatex.

18.8 La gestione dei font

Gestire la moltitudine di font sul proprio elaboratore può talvolta risultare faticoso; bisogna tenere presente che, tradizionalmente, i font classici di LaTeX sono per lo più distribuiti sotto forma di sorgenti scritti con il linguaggio di programmazione METAFONT, e che tutti i programmi delle distribuzioni del sistema TeX sono in grado di lanciare METAFONT, il programma, in modo da generare al volo i font in formato bitmapped adatti al tipo di schermo e di stampante che si sono configurati durante l'installazione del sistema.

Ma questi font bitmapped sono buoni per la stampa, ma non lo sono per la visualizzazione a schermo, come si è già fatto notare. Inoltre, benché non sia impossibile usarli con xelatex e lualatex essi erano pensati inizialmente per tex, poi per latex; ma oggi con pdflatex, xelatex e lualatex si preferiscono di gran lunga i font vettoriali. Lo si è già detto, ma bisogna ripeterlo: pdflatex gestisce solamente i font a 256 caratteri (codificati con indirizzi a 8 bit), mentre xelatex e lualatex gestiscono anche e preferibilmente i font OpenType e TrueType codificati in UNICODE, con indirizzi di diversi byte.

Nel seguito si cercherà di tenere distinte le due classi di font, codificati con un byte, o con molti byte, perché la loro gestione è diversa. Si tengano perciò sott'occhio le note marginali con le quali si marcano quanto è valido solo per i font codificati con un byte validi per pdflatex e invece ciò che è valido per i font gestibili solo con xelatex e lualatex.

Ci si ricordi che mentre su carta i font bitmapped creati per la densità di stampa della stampante in uso danno luogo ad un risultato perfetto, quegli stessi font non sono il meglio che si possa usare per leggere il testo sullo schermo, specialmente se si desidera leggere il testo con ingrandimenti diversi.

Sarebbe meglio disporre dei font in formato vettoriale, ma questi non sono facili da ottenere dai file sorgente METAFONT, quindi è preferibile scaricarli dalla rete 'già fatti'. Inoltre c'è la miriade di font che si possono comperare o che si trovano come freeware, sia in formato Adobe Type 1 (PostScript), sia in formato TrueType e in formato OpenType. La loro installazione non è banale per i seguenti motivi.

LaTeX per lavorare non ha bisogno dei disegni dei font, ma delle loro caratteristiche geometriche; queste sono raccolte in file con l'estensione `.tfm` (TeX Font Metrics); se si usa latex sarà poi il driver che trasforma il file DVI in forma leggibile quello che avrà bisogno dei disegni che, in formato bitmapped, hanno estensioni del tipo `.`⟨*dpi*⟩`pk`, in formato PostScript `.pfb` (PostScript Font in Binary format), in formato TrueType `.ttf` (TrueType Font), in formato OpenType `.otf` (OpenType Font). Se invece si usano gli altri programmi di

composizione, essi hanno bisogno simultaneamente sia dei file metrici sia dei file con le descrizioni vettoriali dei contorni o, in mancanza, dei file bitmapped.

I font bitmapped di origine METAFONT sono codificati solo con un byte e vengono generati insieme con il file `.tfm`, quindi non c'è bisogno d'altro.

I font PostScript ottenuti dal mercato o dalla rete (non ottenuti trasformando i font bitmapped, che ne sono già dotati) sono disponibili con il file metrico che ha formato diverso dal `.tfm` ed ha estensione `.afm` (Adobe Font Metrics).

I font TrueType e i font OpenType non dispongono di un file metrico separato, perché le informazioni metriche sono già contenute nel file stesso che contiene i disegni dei font.

In questa situazione bisogna leggere attentamente le guide che accompagnano il sistema TeX della propria distribuzione e seguirle scrupolosamente; i programmi per le debite conversioni di formato dei file metrici fanno già parte della distribuzione, ma le operazioni da fare devono esser svolte dall'utente/amministratore del PC con grande cura.

La raccomandazione di leggere la documentazione in questo caso è più che una raccomandazione: è un obbligo!

Però questa difficoltà è compensata dal fatto che il sistema TeX, contrariamente a quel che si mormora da parte di coloro che non leggono la documentazione, non è affatto vincolato ai suoi font METAFONT, ma può lavorare con qualunque font esista al mondo, certamente, se non tutti, ameno un numero sterminato.

Infine va ricordato che gli ultimi nati della famiglia dei programmi di composizione del sistema TeX, xelatex e lualatex, consentono di lavorare con i font con la codifica UNICODE e sanno trovare da soli le informazioni metriche di cui necessitano sia nei file `.tfm` o `.afm` sia all'interno dei file `.ttf` e `.otf`, quindi l'uso dei font e degli alfabeti più disparati (compresi quelli ad ideogrammi) diventa decisamente semplificata, specialmente se si dispone di uno *shell editor* che possa scrivere il testo sorgente facendo uso direttamente della codifica UNICODE. Anche ConTeXt-MK-IV (che oggi usa come motore di composizione luatex) è capace di lavorare con i font OpenType con codifica UNICODE.

Va fatta una precisazione: il sistema TeX viene distribuito dalla rete oppure mediante dischi di installazione, ma spesso l'utente installa solo la versione base arricchita dai pacchetti di estensione di cui necessita via via che usa questo sistema di composizione; sarebbe sempre meglio installare la versione completa, ma l'utente non installa tutto perché il sistema e l'intera collezione di estensioni è enorme e quindi piuttosto ingombrante sul disco. Sui PC e laptop più recenti questo non è un vero problema, perché 3 GiB o 4 GiB di occupazione di memoria sono una inezia nei dischi rigidi e nei dischi a stato solido di oggi.

L'installazione parziale è particolarmente infelice per quanto riguarda i pacchetti di estensione che arricchiscono le potenzialità compositive del sistema mediante una grande collezione di font; i diversi formati, però, fanno sì che siano necessarie delle conversioni, perché non tutti i programmi usano lo stesso formato.

Inoltre, come si è avuto occasione di commentare più volte, le codifiche dei font possono essere diverse, spesso lo sono; ma TeX conosce solo le sue codifiche,

quindi, talvolta è necessario anche provvedere al cambiamento di codifica. Per questo scopo si usano sia i 'font virtuali' sia i 'vettori di codifica'. Il discorso, come si vede, può diventare assai complicato, e non si procede oltre.

Tuttavia, per rendersi conto del perché certe cose non sono fatte come uno si aspetterebbe, bisogna ricordare che i file DVI possono essere mostrati su schermo da programmi come YAP o xdvi; e questi quando devono rappresentare graficamente i singoli segni dei font ricorrono di default alle loro immagini bitmapped.

Altri programmi facenti parte del sistema TEX, come per esempio dvips (che trasforma i file dal formato DVI al formato PostScript) oppure dvipdfmx (che trasforma i file dal formato DVI al formato PDF) preferiscono usare i font vettoriali e ricorrono a quelli bitmapped solo in caso di necessità. Hanno perciò bisogno di sapere quali font vettoriali siano disponibili e dove si trovino, perché questa informazione non è contenuta nel file DVI. Il programma pdflatex, invece, ha bisogno di conoscere subito quali font siano disponibili, in che formato e dove si trovino questi font, perché nel file sorgente non c'è nessun accenno a questa informazione. I programmi xelatex e lualatex sono più autonomi e non hanno bisogno di queste informazioni. Ma, attenzione, non dimentichiamo che quanto è disponibile su una macchina non è necessariamente disponibile su un'altra, perché il suo amministratore non ha caricato gli stessi pacchetti e/o non ha comperato gli stessi prodotti commerciali.

Ecco allora che sono necessari tre file per svolgere le funzioni appena descritte; essi si chiamano `psfonts.map`, `dvipdfm.map` e `pdftex.map`; sono collocati in tre cartelle diverse dell'albero di sistema o dell'albero personale delle cartelle di TEX; essi contengono le opportune informazioni rispettivamente per dvips, o per dvipdfmx, o per pdftex (l'interprete di pdflatex). Solo pdflatex e latex

Durante la lavorazione di questo testo le persone che hanno contribuito si sono rese conto che la loro configurazione, benché fortemente personalizzata e piuttosto ricca di estensioni, tuttavia rendeva difficile il lavoro di gruppo, e qualche volta lo rendeva impossibile. Per questo inizialmente si è scelto di usare i font Computer Modern con codifica OT1, anche se non è la scelta ottimale per le lingue con gli accenti come l'italiano, ma è sicuramente presente su ogni installazione. In realtà da diversi anni per le successive versioni di questa Guida si usano i font Latin Modern con la codifica T1, ma il master file dice esattamente che cosa fare se questi font non fossero installati. Finora nessun collaboratore si è lamentato di questi font ed è un bene, perché significa che persino le distribuzioni base fortemente personalizzate sono in grado di essere uniformi con questi font vettoriali, forse persino un poco più belli dei font Computer Modern, ma certamente più adatti alla lingua italiana che fa uso relativamente spesso di diversi accenti.

18.8.1 Font OpenType

I font OpenType codificati con più byte, perché usano la codifica UNICODE, distribuiti con il sistema TEX si trovano nella cartella `.../texmf-dist/fonts/` lualatex o xelatex

`opentype/`. Alcune installazioni sono tali per cui il sistema operativo è immediatamente conscio della loro presenza che si aggiunge alla disponibilità di altri font OpenType presenti sul disco fisso grazie al sistema operativo stesso, o all'installazione di alcuni applicativi, come certe suite di programmi per ufficio.

Per altri sistemi operativi, come varie versioni di Linux, si rende nota la presenza dei font OpenType installati con il sistema TeX semplicemente rinfrescando la 'cache' dei font; come si faccia dipende dalla particolare versione di Linux, ma tipicamente bisogna dare una tantum (vale a dire ogni volta che si installa o si reinstalla il sistema TeX) un comando da linea di comando che informa il sistema operativo della necessita di aggiornare la 'cache' dei font, e il sistema operativo si mette al lavoro per eseguire questa operazione; essa dura qualche decina di secondi, ma deve essere eseguita, al massimo, solo una volta all'anno.

Per il sistema operativo Mac OS X delle macchine Apple, bisogna selezionare tutti i font contenuti nella citata cartella `.../opentype/` e aprirli con l'applicativo FontBook (o LibroFont), il quale chiederà via via conferma, ma installerà tutti i font della cartella, verificandone la validità e aggiungendoli non solo alla 'cache' dei font, ma anche rendendoli visibili nella cartella `~/Library/Fonts/`.

Ciò fatto tutti i font UNICODE già installati con il sistema operativo o con altri applicativi, sono immediatamente disponibili per l'uso con xelatex e lualatex. Si noterà però che ogni volta che si aggiorna un file OpenType o il sistema operativo, benché la 'cache' dei font sia già completa, il programma lualatex diventa un po' lento e non sembra lavorare bene; in realtà esso si aggiorna la sua personale 'cache', ma superata questa fase iniziale, si rimette a funzionare con la sua solita velocità di elaborazione, e non subisce più questi ritardi se non, appunto, quando avvengono quegli aggiornamenti.

Quali font sono allora disponibili su ciascuna delle nostre macchine? Come si fa ad averne un elenco? Chi scrive può dirlo solo per le macchine Apple, perché lavora con una di queste e non dispone di altre macchine né reali né virtuali. I dettagli sono quindi diversi da macchina a macchina, come lo sono i programmi da usare, ma sostanzialmente le azioni da eseguire sono le stesse su tutte le macchine.

Se quindi io apro la cartella `~/Library/Fonts/` sulla mia macchina e seleziono la visione di tutti i font OpenType (non solo quelli di sistema o quelli che mi sono installato personalmente compresi quelli del sistema TeX), posso vedere un elenco di circa 400 famiglie di font OpenType; ogni famiglia contiene generalmente quattro varietà, Regular, Bold, Italics, Bolditalics, ma alcune, poche famiglie ne contengono solo una, mentre altre ne possono contenere una dozzina. Va notato che i font con l'estensione `.otf` sono i 'normali' font OpenType, mentre quelli con l'estensione `.ttf` si riferiscono a font TrueType codificati UNICODE. Questo dipende dal fatto che i font TrueType hanno i contorni descritti da curve di Bézier di secondo grado (archi di parabole) mentre i veri font OpenType hanno i contorni descritti da curve di Bézier di terzo grado (archi di curve cubiche); queste sono finezze che all'utente normale non interessano, quindi non ci si preoccupi se alcuni file di font hanno un'estensione e altri ne hanno un'altra.

Alcuni font hanno nomi arcinoti: Latin Modern, Computer Modern, CM-super, Times, Palatino, Arial, Verdana, Helvetica, Courier, Garamond, Baskerville, Bodoni, Utopia, Palladio, XITS e STIX, Libertine, Libertinus, e via di questo passo. Fra i font appartenenti alla distribuzione del sistema TEX, oltre ai primi tre nominati sopra, ci sono gli Antyqwa Torunska, i font della collezione TeX Gyre, che sono una rivisitazione di font commerciali ben noti riveduti e corretti dal gruppo dei Soci del GUST, gruppo degli utenti polacchi, i font messi a disposizione dalla Società dei Tipografi Ellenici GFSDidot, GFSBodoni, GFSArtemisia ed altri; insomma senza andare a cercare font rarissimi, l'utente del sistema TEX ha a disposizione centinaia di font, che gli consentono di scrivere in qualsiasi alfabeto quasi qualunque lingua parlata e, specialmente, scritta nel mondo. Come si è detto altrove, le lingue che può gestire il sistema TEX nelle sue varianti ammontano a un centinaio.

Tutti i font citati dispongono dei caratteri latini estesi, quasi tutti contengono anche il cirillico e il greco estesi; un buon numero contiene alcuni alfabeti retrogradi, scritti da destra a sinistra, ebraico, arabo, farsi, eccetera; altri contengono anche gli ideogrammi cinesi e i sillabari giapponesi e coreano; inoltre alfabeti meno conosciuti come il georgiano, il devanagari, il thai, eccetera.

Come si fa a usare tutti questi font? È molto semplice e si sono mostrati già diversi esempi; per riassumere: bisogna caricare il file di estensione *fontspec* specificandogli solo opzioni che valgano per tutto il documento; chi scrive usa sempre le legature TEX, quindi specifica sempre la chiamata nel modo seguente:

```
\usepackage{fontspec}[Ligatures=TeX]
```

Questo pacchetto mette a disposizione degli utenti i comandi per impostare le tre famiglie principali e il comando per caricare un font al quale si richiedono caratteristiche speciali:

```
\setmainfont{⟨famiglia con grazie⟩}[⟨opzioni⟩]
\setsansfont{⟨famiglia senza grazie⟩}[⟨opzioni⟩]
\setmonofont{⟨famiglia monospaziata⟩}[⟨opzioni⟩]

\fontspec{⟨font particolare⟩}[⟨opzioni⟩]
\newfontfamily{⟨comando⟩}{⟨font particolare⟩}[⟨opzioni⟩]

\usepackage{amsmath} % facoltativo ma consigliabile sempre
\usepackage{unicode-math}
\setmathfont{⟨font matematico⟩}[⟨opzioni⟩]
```

I nomi degli argomenti da specificare sono autoesplicativi, ma con l'ovvio commento che le ⟨*opzioni*⟩ specificate in ogni riga non sono dello stesso genere. Invece ⟨*comando*⟩ è un comando formato dal solito backslash seguito da una stringa letterale. Se dopo avere caricato *fontspec* e *polyglossia* e dopo aver specificato i font normali per le tre famiglie principali e le lingue da usare, si definisce una nuova famiglia con il nome del ⟨*comando*⟩ ottenuta agglutinando il

nome di una lingua con la parola font, \⟨*lingua*⟩`font`, allora questa famiglia verrà usata ogni volta che si scrive testo in quella ⟨*lingua*⟩. Questa è una funzionalità comodissima.

Si noti che nella sintassi appena mostrata viene anche indicato come gestire i font per la matematica, con l'annotazione che se si vuole usare il file di estensione `amsmath`, lo si può fare, ma lo si deve caricare prima del file di estensione `unicode-math` e della scelta del ⟨*font matematico*⟩. Si noti infine che i font OpenType possono contenere migliaia di glifi, per cui usando lualatex e xelatex non ci si deve preoccupare dei *math group*, o famiglie matematiche, né del loro numero limitato a 16, come avviene per pdflatex; ciascun font matematico di tipo OpenType contiene ogni alfabeto possibile, ogni serie e ogni forma possibile e ogni serie, e gli operatori formati da un solo glifo, come $\times, \in, \infty, \ldots$ che si possono usare anche con la matematica estesa mediante i font della American Mathematical Society; questo implica, perciò, che *non si deve caricare il file di estensione* `amssymb` *né, meno che mai, il file* `amsfonts`.

I simboli disponibili in tutte le forme e serie sono talmente tanti che in un tentativo di elencare tutti i simboli gestibili con `unicode-math` scrivendo accanto a ciascuno qualche parola di spiegazione, si sono composte circa 80 pagine A4. Siccome non è opportuno replicare le 80 pagine in questo testo, si rinvia alla documentazione del pacchetto `unicode-math` che, oltre alla lista dei simboli, contiene anche alcune indicazioni d'uso per certe macro che "costruiscono" alcuni simboli speciali.

L'unica difficoltà nascosta in quei semplici comandi consiste nel conoscere il nome della famiglia e i nomi dei font particolari. A questo punto bisogna leggere le istruzioni contenute nella documentazione di `fontspec` con il solito comando da linea di comando `texdoc fontspec`. Si scopre allora che questo pacchetto può fare moltissime cose molto specializzate il cui solo elenco qui sarebbe troppo lungo.

Tuttavia il problema persiste; `fontspec` non dice come si può trovare il nome della famiglia, ma solo come usarlo.

Il sistema TeX, già in prima installazione, contiene un piccolo programma da linea di comando che si chiama otfinfo; la sua documentazione si legge mediante il solito comando da linea di comando `texdoc otfinfo`; è opportuno leggere quella documentazione, dalla quale si evince che esso accetta molte opzioni, oltre al nome del file che contiene la descrizione del font di cui fornire le proprietà, ivi incluso il nome della famiglia. Ma se non si specificano le opzioni giuste si rischia di ottenere troppa informazione che coinvolge molte schermate nelle quali ci si può perdere. È meglio essere selettivi.

La sintassi del comando otfinfo nel terminale o prompt dei comandi è la seguente:

```
...$ otfinfo ⟨opzioni⟩ ⟨file⟩
```

dove `...$` è quanto scrive il sistema operativo nel terminale quando aspetta l'introduzione di un comando da parte dell'utente; nelle macchine Windows

al posto del segno del dollaro c'è il segno `>`; inoltre ⟨*file*⟩ è il nome di un file contenente le informazioni relative al font che si intende analizzare: si tratta di un indirizzo completo a partire dalla radice del disco; se si volessero le informazioni relative al font Arial sul Mac, dovrei scrivere il comando

```
...$ otfinfo ⟨opzioni⟩ /Library/Fonts/Microsoft/Arial.ttf
```

Già il fatto di dover trovare nel disco il nome del file con il suo percorso a partire dalla radice del disco permette di ottenere qualche informazione sul font, per esempio che questo è stato installato insieme alla suite Microsoft Office™; si vede dall'estensione che si tratta di un file TrueType.

Se come ⟨*opzioni*⟩ specifico solo `-s` il programma mi stampa sullo schermo l'elenco degli alfabeti che contiene; dunque:

```
...-MacBook-Pro: $ otfinfo -s /Library/Fonts/Microsoft/Arial.ttf
arab Arabic
arab.URD Arabic/Urdu
cyrl Cyrillic
grek Greek
hebr Hebrew
latn Latin
latn.LTH Latin/Lithuanian
```

Ecco dunque che veniamo a sapere che con questo font si può comporre nelle lingue che usano l'alfabeto latino, l'arabo, anche nella varietà Urdu, l'alfabeto cirillico; si può scrivere anche in ebraico, in greco e in lituano che usa una particolare varietà dell'alfabeto latino.

Le opzioni disponibili sono le seguenti.

-s mostra gli *script*, gli alfabeti presenti nel font.

-f mostra le varianti che sono presenti nel font e che possono essere selezionate dall'utente; per esempio: creare il maiuscoletto a partire dal maiuscolo; usare stili diversi per il minuscolo e il maiuscolo; spazieggiare le maiuscole; distinguere lo zero 0 dalla O maiuscola barrando lo zero; eccetera.

-z mostra le informazioni disponibili in merito ai corpi ottici.

-p mostra il nome PostScript del font, talvolta necessario per usare comandi di basso livello sottostanti a quelli definiti da *fontspec*.

-a mostra il nome della famiglia.

-v mostra il numero di versione del font.

-i mostra le informazioni generali relative al font: la forma; il nome della famiglia; il nome PostScript del font; la versione; l'identificatore unico secondo lo standard delle ditte produttrici; l'autore; il venditore, almeno il suo URL; il trade mark in termini verbali; il copyright; la licenza d'uso, almeno il suo URL.

Questa è forse una delle opzioni più interessanti; se la invochiamo per il font Arial, otteniamo quanto segue.

```
...-MacBook-Pro:~ $ otfinfo -i /Library/Fonts/Microsoft/Arial.ttf
Family:              Arial
Subfamily:           Regular
Full name:           Arial
PostScript name:     ArialMT
Version:             Version 5.06
Unique ID:           Monotype:Arial Regular:Version 5.06 (Microsoft)
Designer:            Monotype Type Drawing Office - Robin Nicholas,
                     Patricia Saunders 1982
Manufacturer:        The Monotype Corporation
Trademark:           Arial is a trademark of The Monotype Corporation
                     in the United States and/or other countries.
Copyright:           © 2008 The Monotype Corporation. All Rights Reserved.
                     License Description: You may use this font to display
                     and print content as permitted by the license terms
                     for the product in which this font is included.
                     You may only (i) embed this font in content as
                     permitted by the embedding restrictions included
                     in this font; and (ii) temporarily download this
                     font to a printer or other output device to help print
                     content.
Vendor ID:           TMC
```

Se la invochiamo per il font Latin Modern Math otteniamo quanto segue.

```
...-MacBook-Pro:~ $ otfinfo -i ~/Library/Fonts/latinmodern-math.otf
Family:              Latin Modern Math
Subfamily:           Regular
Full name:           LatinModernMath-Regular
PostScript name:     LatinModernMath-Regular
Preferred family:    Latin Modern Math
Preferred subfamily: Regular
Version:             Version 1.958
Unique ID:           1.958
Trademark:           Please refer to the Copyright section for the font
                     trademark attribution notices.
Copyright:           Copyright 2012 for Latin Modern Math OTF by
                     B. Jackowski, P. Strzelczyk and P. Pianowski
                     (on behalf of TeX users groups). This work is
                     released under the GUST Font License --
                     see http://tug.org/fonts/licenses/GUST-FONT-LICENSE.txt
                     for details.
Vendor ID:           PfEd
```

-g mostra i *nomi* di tutti i glifi contenuti nel file; attenzione ad usare questa opzione perché può scrivere migliaia di righe; se interessa conservare questa lista bisogna reindirizzare l'uscita del programma verso un file; l'operazione può essere differente a seconda del sistema operativo.

-u mostra sia il *nome* sia l'*indirizzo* UNICODE di ogni glifo contenuto nel font; si veda la descrizione dell'opzione precedente.

-t mostra le dimensioni e il nome di ogni tabella UNICODE contenuta nel font.

-T ⟨*tabella*⟩ mostra il contenuto della particolare ⟨*tabella*⟩ OpenType specificata.

--script = ⟨*script*⟩[⟨*.lang*⟩] usato insieme all'opzione -f permette di mostrare le particolarità associate solo all'alfabeto indicato con ⟨*script*⟩, facoltativamente esteso con la specificazione della lingua; per esempio volendo le

specifiche del font latino per il lituano bisognerebbe specificare `--script = latn.LTH -f`.

-V mostra l'operato del programma in modo "verboso".

-q ordina al programma di lavorare in modo quieto, senza scrivere informazioni non relative alle proprietà del font.

-h ordina al programma di mostrare un riassunto di quanto esposto qui per esteso.

--version mostra il numero di versione del programma.

Ecco, sembra complicato, ma è semplicissimo, come si è mostrato con i due soli esempi fatti.

Questa lunga spiegazione andrebbe completata con una attenta lettura di *fontspec*; in ogni caso il tutto si riduce a poche cose come mostra i seguente preambolo preso da un altro documento, visto che questo testo è stato composto con pdflatex.

```
\documentclass[12pt,twoside]{toptesi}
\ProvidesFile{toptesi-it.tex}[2020 v....]
...
    \usepackage{fontspec}
    \usepackage{polyglossia}
    \setmainlanguage[babelshorthands]{italian}
    \setotherlanguage[variant=ancient]{greek}

    \setmainfont[Ligatures=TeX]{TeX Gyre Termes}
    \setsansfont[Ligatures=TeX, Scale=MatchLowercase]{TeX Gyre Heros}
    \setmonofont{UM Typewriter}
    \newfontfamily{\greekfont}{GFS Didot}
    \usepackage{amsmath}
    \usepackage{unicode-math}
    \setmathfont{XITS Math}
...
\usepackage{booktabs,array,tabularx,fancyvrb,xcolor,imakeidx}

\usepackage{hyperref}
    \hypersetup{%
        pdfpagemode={UseOutlines},
        bookmarksopen,
        pdfstartview={FitH},
        colorlinks,
        linkcolor={blue},
        citecolor={blue},
        urlcolor={blue}
    }
\makeindex[intoc,columns=2]
...
```

Si riconoscono in questo preambolo molte delle cose spiegate anche in capitoli precedenti; in particolare si vede come sono state definite le tre famiglie principali

per i font e come sono stati associati i font Greci Didot alla lingua greca.

Come si vede queste impostazioni sono un poco più dettagliate rispetto a quelle che si userebbero con il programma pdflatex, ma sono molto più flessibili che non quelle destinate all'uso dei font codificati con un byte; la cosa è del tutto ovvia: i font OpenType permettono un numero molto maggiore di personalizzazioni, per cui anche i comandi per le personalizzazioni sono più dettagliati. Vale la pena osservare che se non si specificano font diversi per le tre famiglie principali, i font usati per impostazione predefinita sono i Latin Modern nella loro forma OpenType.

18.8.2 Font Type 1

Solo pdflatex

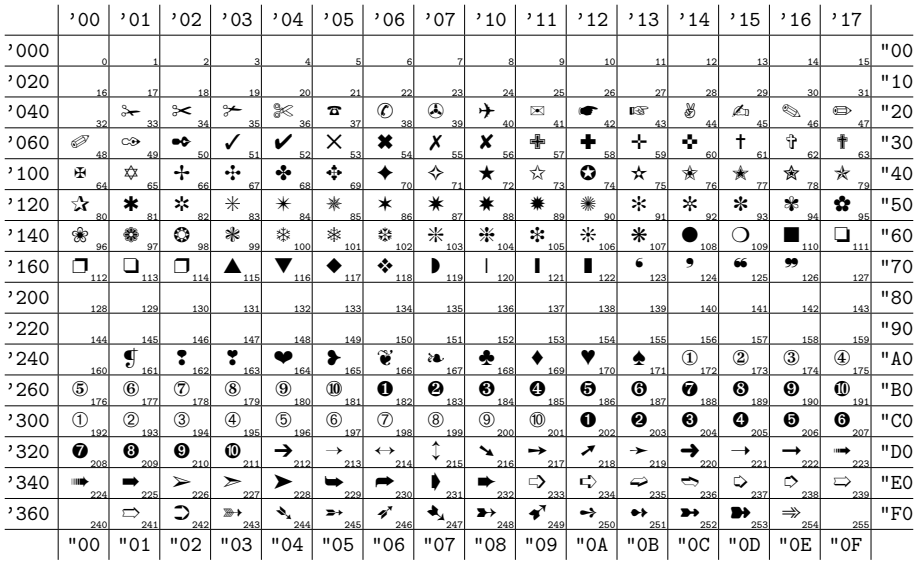
I font Type 1 sono codificati con un byte, e sono i soli font vettoriali che si possono usare con pdflatex. Sebbene i font Type 1 siano usabili anche con i programmi lualatex e xelatex, per servirsi contemporaneamente di questi font e di quelli OpenType bisogna ricorrere a certe astuzie, che possono essere lette nella documentazione di *fontspec*. Qui dunque ci occuperemo solo dell'uso dei font Type 1 mediante il programma pdflatex.

18.8.2.1 Altri font già disponibili

Prima di descrivere la delicata operazione dell'installazione di nuovi font Type 1, vale la pena di descrivere i pacchetti che fanno riferimento a font già disponibili in ogni distribuzione del sistema TeX. Talvolta è necessario scaricare da CTAN ed installare pacchetti già predisposti con tutto il necessario, ma molto spesso i pacchetti fanno parte della dotazione ordinaria del sistema TeX quando ne sia stata fatta una installazione completa.

Fra questi pacchetti si sono già segnalati quelli corrispondenti ai font Times eXtended e Palatino eXtended. L'aggettivo 'extended' (esteso) si riferisce al fatto che le collezioni dei segni usabili sono più ampie (estese) rispetto all'insieme di segni forniti ordinariamente dai font delle collezioni CM, EC e TS arricchite dai segni delle collezioni fornite dall'American Mathematical Society. Il pacchetto *newtx* mette a disposizione due file di estensione: *newtxtext* e *newtxmath* che possono venire usati separatamente. Questa nuova versione corregge alcuni piccoli difetti di accostamento nella composizione della matematica che invece erano presenti con i font Times usati dal pacchetto *txfonts*. Lo stesso è avvenuto con i font Palatino eXtended.

Il documento *Using common PostScript fonts with LaTeX*, leggibile dando da terminale il comando `texdoc psnfss2e`, descrive tutti i font PostScript con i relativi pacchetti e le rispettive opzioni già predisposti per usare i font Palatino (font diversi e con prestazioni inferiori a quelle dei pacchetti *pxfonts* e *newpx*), i Times (anche questi con prestazioni inferiori a quelle fornite dai pacchetti *txfonts* e *newtx*) Helvetica, Avant Garde, Courier, Zapf Chancery, Bookman, New Century Schoolbook, Charter, Utopia. Il documento descrive anche i limiti



Parametri			
inclinazione	0,00000	x-height	3,99998 pt
spazio interparola	2,77999 pt	larghezza del quadrato	10,00000 pt
allungamento interparola	3,00000 pt	spazio extra	0,00000 pt
accorciamento interparola	1,00000 pt	corpo nominale	10,00000 pt

Figura 18.8: I *dingbats* creati da Hermann Zapf

di ogni pacchetto e segnala i pacchetti obsoleti da *non* usare, se non per comporre vecchi documenti che ricorrevano ad alcuni di questi font.

Il documento segnala anche alcuni pacchetti, per esempio *Pifonts*, da caricare per disporre dei simboli e dei 'dingbats' forniti dalle polizze che offrono solo questo genere di segni. Diventa così abbastanza semplice disporre di tutti quei *pittogrammi* usati nel mondo occidentale per segnalare luoghi specifici (telefono, luoghi di culto, aeroporti, ...), delle manine che indicano e attraggono l'attenzione su un oggetto o una parola, motivi di decorazione, cifre da 0 a 10 variamente stilizzate, molti tipi di frecce, eccetera. La figura 18.8 presenta una di queste polizze di segni speciali.

Va segnalato che con alcuni di questi font, e per altri non indicati, il pacchetto *mathdesign* consente di associare i font matematici e le collezioni di simboli in modo che molte delle limitazioni e degli inconvenienti segnalati dal documento `psnfss2e.pdf` possono venire eliminati. Questo pacchetto consente di estendere i font Utopia, Charter e Garamond; esso consente quindi di personalizzare fortemente l'aspetto grafico dei propri lavori di composizione, senza però rinunciare a nessuna delle prerogative di LaTeX. Chi scrive, però, ha raramente usato questi altri font, non perché egli li consideri inadatti, ma perché si è decisamente trovato

meglio con i font Latin Modern, anche se i suoi scritti sembrano 'monòtoni' per mancanza di varietà di font.

Chi scrive ha usato talvolta l'uno o l'altro dei pacchetti *fourier* e *kpfonts*; il primo offre una installazione completa per il testo e la matematica, compresi i simboli della American Mathematical Society, e molti altri simboli e altri alfabeti. Si basa sui font Utopia, molto belli e leggibili, visto che a pari corpo hanno un occhio più grande dei Latin Modern. Analogamente il secondo pacchetto offre pressappoco le stesse prestazioni del primo, ma usando come font principale un'ottima approssimazione del font Palladio; chi scrive apprezza molto entrambi questi pacchetti.

In questo testo sono stati usati tra gli altri i font della famiglia Text Companion vettoriali; questa collezione di font si affianca ai font di default per mettere a disposizione un altro centinaio di lettere e segni analfabetici molto utili in diverse circostanze.

Il sistema TeX contiene già preinstallati i font della collezione `CM-super`, completa di file metrici, di font virtuali, di font vettoriali e di tutte le varianti con le codifiche *T1* (caratteri latini), *T2* (caratteri cirillici) e *TS1* (simboli della collezione Text Companion); grazie ai font virtuali, condivide quei segni che sono identici fra i caratteri latini e cirillici. Essi figurano già nelle mappe relative ai caratteri vettoriali e non è necessario preoccupassi della loro installazione.

Sono preinstallati anche i font Latin Modern (LM) che ora sono distribuiti anche nella forma OpenType e sono predefiniti quando si usano xelatex e lualatex; essi offrono una migliore collezione di segni vettoriali, apparentemente migliori dei CM-super secondo molti utenti. Certamente, se non si ha bisogno di usare il cirillico, la collezione Latin Modern è molto più parca per quel che riguarda lo spazio sul disco.

I font della collezione Latin Modern comprendono anche i file di ricodifica conformi alle norme Adobe per i font PostScript e permettono di servirsi di codifiche diverse dalle solite. È possibile usarli anche con codifiche diverse da *T1*, ma non c'è bisogno di fare nulla per la loro installazione.

Tra le codifiche con cui possono essere usati i font Latin Modern, c'è anche la codifica *LY1*; è una codifica interessante, perché consente di evitare il caricamento dei font Text Companion, a meno che non si debbano usare simboli molto particolari; per esempio per usare il simbolo dell'euro il comando `\texteuro` richiederebbe l'uso di font Text Companion se per il testo si usa la codifica T1, mentre se si usa la codifica LY1, non ce n'è bisogno; lo stesso vale per altri pochi segni; è meglio verificare sulla tabella 18.13 per vedere che cosa è disponibile e che cosa manca.

Ricordo quanto già segnalato: gli ultimi aggiornamenti (vedi il capitolo **??**) del nucleodi LaTeX non richiedono più l'invocazione dell'apposito pacchetto per usare il Text Companion font.

	’00	’01	’02	’03	’04	’05	’06	’07	’10	’11	’12	’13	’14	’15	’16	’17	
’000	0	€ 1	2	3	/ 4	· 5	˝ 6	˛ 7	fl 8	9	10	ff 11	fi 12	13	ffi 14	ffl 15	"00
’020	ı 16	J 17	` 18	´ 19	ˇ 20	˘ 21	¯ 22	° 23	˛ 24	ß 25	æ 26	œ 27	ø 28	Æ 29	Œ 30	Ø 31	"10
’040	32	! 33	" 34	# 35	$ 36	% 37	& 38	’ 39	(40	) 41	* 42	+ 43	, 44	- 45	. 46	/ 47	"20
’060	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	: 58	; 59	< 60	= 61	> 62	? 63	"30
’100	@ 64	A 65	B 66	C 67	D 68	E 69	F 70	G 71	H 72	I 73	J 74	K 75	L 76	M 77	N 78	O 79	"40
’120	P 80	Q 81	R 82	S 83	T 84	U 85	V 86	W 87	X 88	Y 89	Z 90	[91	\ 92	] 93	^ 94	_ 95	"50
’140	‘ 96	a 97	b 98	c 99	d 100	e 101	f 102	g 103	h 104	i 105	j 106	k 107	l 108	m 109	n 110	o 111	"60
’160	p 112	q 113	r 114	s 115	t 116	u 117	v 118	w 119	x 120	y 121	z 122	{ 123	| 124	} 125	~ 126	¨ 127	"70
’200	Ł 128	' 129	‚ 130	f 131	„ 132	... 133	† 134	‡ 135	ˆ 136	‰ 137	Š 138	‹ 139	Œ 140	Ž 141	ˆ 142	— 143	"80
’220	ł 144	‘ 145	’ 146	“ 147	” 148	• 149	– 150	— 151	˜ 152	TM 153	š 154	› 155	œ 156	ž 157	~ 158	Ÿ 159	"90
’240	160	¡ 161	¢ 162	£ 163	¤ 164	¥ 165	¦ 166	§ 167	¨ 168	© 169	ª 170	« 171	¬ 172	- 173	® 174	¯ 175	"A0
’260	° 176	± 177	2 178	3 179	´ 180	µ 181	¶ 182	· 183	¸ 184	1 185	º 186	» 187	¼ 188	½ 189	¾ 190	¿ 191	"B0
’300	À 192	Á 193	Â 194	Ã 195	Ä 196	Å 197	Æ 198	Ç 199	È 200	É 201	Ê 202	Ë 203	Ì 204	Í 205	Î 206	Ï 207	"C0
’320	Đ 208	Ñ 209	Ò 210	Ó 211	Ô 212	Õ 213	Ö 214	× 215	Ø 216	Ù 217	Ú 218	Û 219	Ü 220	Ý 221	Þ 222	ß 223	"D0
’340	à 224	á 225	â 226	ã 227	ä 228	å 229	æ 230	ç 231	è 232	é 233	ê 234	ë 235	ì 236	í 237	î 238	ï 239	"E0
’360	ð 240	ñ 241	ò 242	ó 243	ô 244	õ 245	ö 246	÷ 247	ø 248	ù 249	ú 250	û 251	ü 252	ý 253	þ 254	ÿ 255	"F0
	"00	"01	"02	"03	"04	"05	"06	"07	"08	"09	"0A	"0B	"0C	"0D	"0E	"0F	

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

Parametri

inclinazione	0,00000	x-height	4,30549 pt
spazio interparola	3,33332 pt	larghezza del quadrato	10,00000 pt
allungamento interparola	1,66665 pt	spazio extra	1,11111 pt
accorciamento interparola	1,11111 pt	corpo nominale	10,00000 pt
alfabeto maiuscolo	186,18060 pt	alfabeto minuscolo	127,58364 pt

Tabella 18.13: Il font latino con codifica LY1

18.8.2.2 Installazione di altri font vettoriali

Se si va ad aprire il file `pdftex.map` si scopre che contiene 12 445 righe (almeno sul calcolatore di chi scrive, che in aggiunta ai font normalmente distribuiti con l’installazione TEX Live completa, ha aggiunto una ventina di font personali) fra le quali poche righe di commento; si può stimare, visto che ogni riga si riferisce ad un diverso font, che su questo calcolatore la distribuzione TEX Live completa mette a disposizione circa 12 000 diversi font vettoriali; sembra impossibile che con tale varietà di scelta si debbano installare altri font, ma succede.

Qui si esporranno alcune situazioni che si presentano effettivamente, non molto spesso, ma si presentano. Installare nuovi font vettoriali è difficile specialmente per le numerose piccole attenzioni che bisogna avere ad ogni passo della procedura. Nel seguito mi riferirò specialmente alla distribuzione TEX Live, ma non trascurerò MiKTEX, visto che alcune operazioni da compiere con quest’ultimo sono diverse.

Vale la pena premettere la scaletta delle operazioni da compiere per installare ciascun nuovo font vettoriale; ogni passo verrà illustrato nei paragrafi successivi.

1. Creare i file metrici `.tfm` e, se occorre, il file virtuali `.vf`:
   - per font Type 1 generati con FontForge;
   - per font Type 1 generati a partire da font creati con METAFONT;
   - per i font Type 1 acquistati o scaricati dalla Rete;
   - per i font TrueType;
   - per i font OpenType.
2. Creare i file di descrizione dei font `.fd` e magari anche il pacchetto di estensione `.sty`.
3. Creare i file di mappa `.map`:
   - con TeX Live;
   - con MiKTeX.
4. Aggiornare i file generali delle mappe:
   - per un solo utente;
   - per tutti gli utenti della macchina
     - con TeX Live;
     - con MiKTeX.
5. Verificare di disporre dei programmi per eseguire i compiti precedenti.

La lista sembra lunga, e lo è, ma se si ha la pazienza di seguire con attenzione i vari passi nell'ordine in cui sono esposti, non dovrebbe essere difficile installare nuovi font vettoriali. In ogni caso procederò per esempi; quindi è possibile che alcuni punti della procedura non vengano esposti; in questi casi il lettore è indirizzato verso la documentazione prevalentemente contenuta in quella di TeX Live (dare il comando `texdoc texlive` dal terminale), oltre che nel messaggio di aiuto del programma `updmap` (dare il comando `updmap -help` dal terminale); oppure consultando la documentazione di MiKTeX.

Che cosa siano alcuni dei file nominati nella lista, apparirà chiaro da quanto esposto qui di seguito.

18.8.2.3 Operazioni preliminari

Nei paragrafi che seguono si parlerà di diversi tipi di file; è opportuno che ogni tipo di file risieda in una cartella opportuna, possibilmente innestata in una struttura conforme al TDS (TeX Directory System); l'installazione di TeX Live già prevede questa struttura. MiKTeX vi si avvicina abbastanza, ma sembra che la sua struttura di cartelle non sia completamente conforme alle direttive TDS. Questa struttura è descritta con abbastanza dettaglio nella documentazione di TeX Live, leggibile con `texdoc texlive`. Partendo dalla radice dell'unico albero o dall'insieme di alberi della distribuzione installata, dovrebbe contenere diverse ramificazioni:

- `fonts/tfm/` destinata a contenere i file metrici;
- `fonts/vf/` destinata a contenere il file dei font virtuali;

- `fonts/type1/` destinata a contenere i file dei font Type 1;
- `fonts/truetype/` destinata a contenere i file dei font TrueType
- `fonts/opentype/` destinata a contenere i file dei font OpenType;
- `fonts/map/` destinata a contenere i file delle mappe;
- `fonts/enc/` destinata a contenere i file di codifica dei font virtuali;
- `fonts/pk/` destinata a contenere i file dei font bitmapped;
- `web2c/` destinata a contenere i file di configurazione delle mappe dei font;
- `fonts/source/` destinata a contenere i file sorgente per il programma METAFONT;
- `tex/latex/` destinata a contenere i file di descrizione dei font e, se esistono, i file di estensione per l'uso dei corrispondenti font.

Spesso bisogna replicare questa struttura nell'albero personale e, se c'è, anche nell'albero locale; questo solitamente comincia con una cartella che si chiama `texmf-local`. Invece l'albero personale è una cosa personale dell'utente e non viene creato con l'installazione del sistema TEX; tocca all'utente crearselo con la struttura indicata. Nelle macchine Linux generalmente questo albero personale è radicato in `~/texmf`; nelle macchine Mac esso va radicato in `~/Library/texmf/`; nelle macchine Windows dipende dalla versione: sulle macchine WinVista e successive potrebbe essere radicato in `C:\Users\`⟨*utente*⟩`\texmf`.

In linea generale quello che si installa e non fa parte in senso stretto della distribuzione TEX Live o MiKTEX è bene che sia installato nell'albero personale o, in TEX Live, nell'albero locale. Ciò che viene messo nell'albero locale richiede che sia stato creato e che venga aggiornato il database dei nomi dei file; nell'albero personale questo aggiornamento non è necessario con TEX Live. Il motivo di questa preferenza è che gli alberi pertinenti alla distribuzione in senso stretto vengono riscritti ad ogni installazione di una nuova versione della propria distribuzione del sistema TEX; a parte gli eventuali privilegi per scrivere nelle cartelle della distribuzione, mettere in queste cartelle qualcosa che non fa parte in senso stretto della distribuzione vuol dire perderlo completamente ad ogni aggiornamento di una nuova versione. Sarebbe un "infortunio" particolarmente sgradevole.

Con TEX Live ciò che sta nell'albero locale è accessibile a tutti gli utenti della macchina *in lettura*; quindi per *scrivere* qualcosa nelle cartelle di questo albero bisogna avere le prerogative dell'amministratore. Invece nell'albero personale scrive e legge solo il proprietario. Sulle macchine Windows il concetto di amministratore è ancora abbastanza vago, ma sta diventando un po' più definito con le ultime versioni del sistema operativo, da Vista in poi. Da molte versioni gli utenti sono divisi in amministratori, utenti con permessi estesi, utenti normali, ospiti, eccetera, ma di fatto le differenze per quel che riguarda il sistema TEX sono abbastanza modeste. Con i sistemi operativi di tipo UNIX, il concetto di amministratore è sempre stato molto preciso come lo sono i modi di accesso (lettura, scrittura, esecuzione) per i singoli file.

Nel seguito si supporrà che ogni tipo di file sia collocato nella cartella che gli pertiene, possibilmente con un nome fortemente mnemonico, collocata lungo

la ramificazione giusta dei vari alberi descritti sopra, e non lasciato in cartelle qualsiasi. I programmi del sistema TeX sanno dove trovare i file sui quali devono operare solo se sono nelle ramificazioni giuste.

18.8.3 Installare un font Type 1 creato con FontForge

È capitato a chi scrive di creare font con il programma FontForge; questo è un programma gratuito disponibile per tutte e tre le piattaforme principali che conviene installare anche per altri scopi; si tratta di un editor di font vettoriali con il quale è possibile creare font partendo da zero, come anche convertire in formato Type 1 font bitmapped generati con METAFONT; riesce anche a creare font vettoriali partendo da immagini bitmapped di glifi fotografati o copiati da una schermata. In uscita può produrre font vettoriali in diversi formati, il più comune dei quali è appunto il formato Type 1, ma è in grado di creare anche font OpenType.

Lo scrivente ha dovuto creare il logo della Università delle Tre Età e lo ha fatto con FontForge creando direttamente il font `uni3.pfb` e il corrispondente file `uni3.tfm`. Questo font non ha una codifica definita perché contiene solo un glifo nella posizione corrispondente alla lettera 'U'. Per questo motivo per scrivere questo glifo non occorre né un file di descrizione del font (benché non sia inutile) né un file contenente qualche definizione per comporre questo solo glifo; basta avere definito nel preambolo o in un file di macro personali quanto segue:

```
\newfont{\unitre}{uni3}
\newcommand*\LogoUniTre{{\unitre U}}
```

Il primo comando non è altro che un comando del vecchio LaTeX 209 che, dopo aver controllato che la macro sia definibile, associa il nome della macro `\unitre` al font appena creato; il secondo comando permette di comporre il logo della Università delle Tre Età.

Ma questo non basta; perché pdfLaTeX possa usare questo font Type 1 bisogna dirgli che il font esiste e quale font vero corrisponde al nome del file metrico. Questo si fa in due tempi; prima si scrive un file `.map` che descrive quella associazione, per esempio `uni3.map`; poi si inserisce questo file di mappa nel file generale con il quale pdfLaTeX accede ai font realmente esistenti. Il file di mappa per questo font, con il nome, per esempio, `uni3.map`[9] ha questo contenuto:

```
uni3␣uni3␣<uni3.pfb
```

La prima istanza del nome 'uni3' si riferisce al nome senza estensione del file metrico; la seconda istanza si riferisce al nome del font contenuto dentro il file `.pfb`, nome che si può agevolmente leggere per mezzo di FontForge; il terzo elemento preceduto dal segno `<` è il nome per esteso, completo di estensione del file vettoriale che contiene la descrizione grafica dei glifi contenuti nel font.

[9] Il nome può essere scelto a piacere; basta usarlo coerentemente. Ma il contenuto del file di mappa qui indicato può anche essere inserito in un file di mappa generico che raccoglie le informazioni necessarie anche per altri font.

Per il secondo passo, quello di inserire questa informazione nei file generali di mappa, di veda più avanti il paragrafo 18.8.8. Dopo aver eseguito questo secondo passo, pdflatex (e gli altri programmi che usano i font vettoriali: dvips e dvipdfmx) possono usare il font descritto in questa sezione.

18.8.4 Installare un font Type 1 creato con METAFONT

Supponiamo di disporre dei file sorgente `mf` per creare dei font con METAFONT; che questi file facciano parte della distribuzione (come, per esempio, i file della collezione dei font EC), oppure che si siano caricati dalla rete, oppure che l'utente abbia creato da solo per disegnare certi suoi font non fa nessuna differenza, purché questi file siano nella ramificazione di cartelle giusta.

Eseguendo METAFONT vengono generati i file `.tfm` e i file delle bitmap dei font; questi sono file con l'estensione formata agglutinando la densità di pixel al pollice con la desinenza `pk`; quindi il font bitmapped cmr10 creato per lavorare con una stampante da 300 dpi ha il suo file che si chiama `cmr10.300pk`. Questi file 'pk' non rappresentano font vettoriali, e se servono a qualcosa, servono solo se si desidera comporre per ottenere il file composto in formato DVI senza volerlo trasformare in altri formati; se lo si trasformasse, i programmi di trasformazione userebbero i font vettoriali se ne esistono con lo stesso nome.

Bisogna trasformare il font bitmapped appena creato con METAFONT in un font vettoriale; a questo scopo si può usare lo script python mftrace. Innanzi tutto se non lo si è già fatto bisogna installarselo dalla rete; inoltre, mentre l'interprete python è già presente sulle macchine UNIX (Linux e Mac) questo interprete generalmente non fa parte dei sistemi operativi Windows, quindi bisogna installarselo; in alternativa è possibile installarsi il sistema CygWin che forma sulle macchine Windows un ambiente virtuale che simula una macchina UNIX; in questo caso il programma mftrace è bene che sia installato nella "macchina virtuale" CygWin, nella quale si installeranno anche gli altri programmi indicati in questo paragrafo.

Supponendo che il file da trasformare si chiami `miofont.mf`, si lancia poi il programma mftrace con il comando

```
mftrace -f PFB -e encfile --potrace --simplify miofont
```

dove `-f PFB` specifica il formato di uscita ed è obbligatorio se si vuole un formato diverso da PFA che è il formato di uscita predefinito; `-e encfile` serve per specificare facoltativamente il nome del file che definisce l'ordine nel quale sono elencati i vari glifi all'interno del file di uscita; questi file si trovano nell'apposita ramificazione indicata precedentemente; `--potrace` specifica che si vuole usare il programma potrace per tracciare il contorno di ogni glifo; `--simplify` serve per semplificare la descrizione del contorno espressa mediante l'elenco dei nodi e dei punti guida delle spline di Bézier, visto che l'operazione di tracciamento potrebbe avere indicato troppi nodi e punti di controllo in relazione ad una certa approssimazione consentita; ovviamente `miofont` è il nome del font che si vuole trasformare in vettoriale ma di cui si dispone solo della versione per

METAFONT. Naturalmente si possono specificare altre opzioni; le istruzioni per eseguire questo programma si trovano soltanto nello stesso sito da cui lo si può scaricare `http://lilypond.org/mftrace/`.

A grandi linee mftrace opera così: genera nuovamente il file `pk` del font in questione con un'altissima densità di pixel al pollice, come se dovesse creare un font bitmapped per caratteri cubitali; si capisce che maggiore è la dimensione dell'immagine bitmapped di ogni glifo, minore è l'influenza della seghettatura del suo contorno. Traccia i contorni dei vari glifi mediante spline di Bézier del terzo ordine e memorizza il risultato in un file vettoriale intermedio che abbia il formato giusto per essere dato in pasto al programma FontForge (che quindi deve già essere installato); FontForge provvede a semplificare l'insieme dei nodi e dei punti di controllo dei contorni dei glifi e genera i font in formato `.pfb`. Alla conclusione mftrace cancella tutti i file ausiliari su cui ha lavorato, lasciando nella cartella di lavoro solo il file `.pfb`.

Il file `.tfm` non è stato toccato, quindi si trova già al posto giusto, ma bisogna ancora spostare il file `.pfb` nell'opportuna cartella della ramificazione giusta.

Non resta che creare il file di mappa e procedere, quindi, come spiegato alla fine del paragrafo 18.8.3.

18.8.5 Installare font Type 1

I font Type 1 che si possono acquistare o scaricare dalla rete sono dotati di un file `.pfb` (raramente di un file in formato ASCII `.pfa`) e di un file metrico nel formato `.afm` (Adobe Font Metrics). Il file `.pfa` nel formato leggibile dagli uomini è piuttosto ingombrante; è meglio trasformarlo nel formato `.pfb`; per questo scopo è meglio affidarsi al programma FontForge.

Tuttavia non basta; i glifi all'interno del file `.pfb` potrebbero essere disposti in un ordine diverso da come se lo aspetta il sistema TEX; per ovviare a questo e per ottenere il file metrico per il sistema TEX e, facoltativamente, il font virtuale, basta eseguire il programma afm2tfm specificandogli le opzioni giuste e il file di codifica giusto; per i dettagli bisogna consultare la documentazione; con `texdoc afm2tfm` si ottiene una documentazione scarna e obsoleta, mentre con il comando `afm2tfm -help` si ottiene una documentazione succinta ma sufficiente allo scopo. Si noti che se si vuole generare anche il font virtuale, questo esce in formato leggibile `.vpl` (Virtual Property List); per convertirlo in formato binario `.vf`, quello che viene effettivamente usato dai programmi di trasformazione o di composizione, bisogna trasformarlo ancora nel formato desiderato mediante il programmino vptovf.

Non resta che creare il file di mappa come indicato alla fine del paragrafo 18.8.3.

Questa semplice sequenza di operazioni che può includere anche il trasferimento dei vari file nelle cartelle giuste della diramazione giusta, può venire eseguita mediante uno script bash per le macchine UNIX o bat per le macchine Windows. Alternativamente si possono usare programmi o script come fontinst (un po' vecchiotto) o autoinst alla cui documentazione si rinvia il letto-

re: `texdoc fontinst` e `texdoc autoinst`. Questi programmi, specialmente il secondo, possono provvedere anche a spostare i vari file nelle cartelle giuste di una struttura TDS.

18.8.6 Installare font TrueType

Per i font TrueType (codificati con un solo byte) le cose si sviluppano in modo del tutto simile a quello seguito per i font Type 1: si esegue il programmino ttf2tfm che estrae il font metrico per il sistema TEX direttamente dai font TrueType, non da un file separato, e crea anche il file virtuale.

Bisogna ancora creare il file di mappa come indicato alla fine del paragrafo 18.8.3; un'unica avvertenza: nel file di mappa bisogna indicare nel nome del file reale l'estensione `.ttf` invece dell'estensione `.pfb`; sembra ovvio, ma è meglio ricordarlo.

18.8.7 Installare font OpenType

In linea di principio l'installazione di un font OpenType, alcuni dei quali hanno l'estensione `.ttf` come i font TrueType ma hanno i glifi codificati con più di un byte, da usare con pdflatex non differisce granché da quella di un font TrueType; in pratica la cosa è molto più complessa, perché i font OpenType contengono moltissimi glifi, e dispongono di proprietà specifiche per usare certi glifi al posto di altri; per esempio, spesso dispongono di cifre maiuscole (tutte alte uguali e come le lettere maiuscole) e di cifre minuscole (le cifre di altezze diverse, alcune alte come le lettere minuscole, alcune con ascendenti, altre con discendenti; si confronti infatti 0123456789 con 0123456789).

L'operazione si esegue in modo relativamente semplice mediante il programma autoinst che consente, almeno con una installazione conforme alle direttive TDS, di creare i file virtuali necessari con le transcodifiche necessarie, insieme ai file metrici per l'uso con pdflatex, spostandoli tutti nelle cartelle giuste; crea anche i file di mappa. La documentazione leggibile con `texdoc autoinst` è abbastanza chiara, ma bisogna avere familiarità con diversi termini tecnico-informatici.

Riuscendo però a compiere questa operazione, non resta che creare i file generali di mappa, come viene descritto qui di seguito.

18.8.8 Aggiornamento dei file generali di mappa

I file generali di mappa sono le raccolte di tutti i file di mappa per tutti i font che l'utente vuole usare; sono scritti nei linguaggi che i programmi pdftex, dvips e dvipdfm sanno interpretare; sono rispettivamente i file `pdftex.map`, `psfonts.map`, `dvipdfm.map`, oltre ad altri file per altri programmi di trasformazione.

Senza questi file i programmi di composizione non possono riconoscere la disponibilità dei font vettoriali, quindi non possono lavorare o non possono produrre risultati corretti.

18.8.8.1 TeX Live

Dalla versione 2012 di TeX Live le modalità per generare le mappa generale sono cambiate; in un certo senso sono più semplici di quanto non lo fossero con le versioni precedenti.

In pratica consistono nell'eseguire il programma updmap-sys oppure il programma updmap. Il secondo genera i file generali di mappa per l'uso del solo utente che dà il comando, e memorizza i file ottenuti nell'albero personale. Il primo comando, che richiede i privilegi di amministratore, genera i file generali di mappa nell'albero locale che è accessibile a tutti gli utenti della macchina in sola lettura; da questo deriva che per eseguire il programma, l'utente deve acquisire i privilegi dell'amministratore; sulle macchine Windows dovrebbe eseguire il login con lo username di un utente che abbia i privilegi dell'amministratore, se per caso il suo username non è già configurato per disporre di tali privilegi; questo succede spessissimo, visto che la maggior parte dei personal computer sono installati dall'utente che l'ha comprato e che in quanto tale è il primo utente, quindi l'amministratore; coloro che, dopo l'acquisto, fanno eseguire il primo avviamento da una terza persona, generalmente non dispongono dei privilegi dell'amministratore; stranamente sono molto numerose le persone che si fanno fare questa operazione da terzi e ignorano che non possono più amministrare il loro PC. In questo caso saranno "costretti" a creare i file generali di mappa solo per se stessi; probabilmente non noteranno nessuna differenza se sono gli unici utenti del loro PC. Ma una differenza c'è; infatti ogni nuova installazione e ogni aggiornamento di TeX Live non aggiornano mai l'albero personale, e quindi non vengono ricreati i file di mappa aggiornati e l'utente rimane spiazzato, perché non riesce più ad usare i suoi font personali; basta che si ricordi di eseguire updmap e si ritrova il proprio sistema in grado di usare i font personali; ma bisogna riconoscere che per lui/lei è una seccatura; la soluzione è quella di essere amministratori del proprio PC, senza ricorrere a terze persone.

Ma su altre macchine, dove il concetto di amministratore è più definito, è meglio creare i file generali di mappa come amministratore rendendo quindi accessibili questi file, in sola lettura, a chiunque abbia il diritto di usare la macchina.

Per acquisire i diritti dell'amministratore, o del *super user*, che è quasi lo stesso, l'utente deve scrivere il comando `sudo` prima di `updmap-sys`:

```
sudo updmap-sys
```

Dato questo comando, il sistema operativo chiede di immettere la parola chiave dell'amministratore, e se è corretta, esegue il successivo comando.

Tutto ciò è semplicissimo, ma come fa updmap-sys a sapere quali file di mappa deve usare? Ci sono sparsi nei vari rami degli alberi del sistema TeX e nell'albero personale, diversi file `updmap.cfg`; il programma updmap-sys usa tutti questi file di configurazione, o meglio tutti quelli che il programma della distribuzione TeX Live kpsewhich riesce a trovare; se si dà questo comando, sul mio Mac in questo momento, nella forma:

```
kpsewhich --all updmap.cfg
```

esso risponde:

```
~/Library/texmf/web2c/updmap.cfg
/usr/local/texlive/texmf-local/web2c/updmap.cfg
/usr/local/texlive/2019/texmf-dist/web2c/updmap.cfg
```

e questi sono i tre file di configurazione che updmap-sys concatena e usa tutte le informazioni di ciascun font che sono nominati all'interno di ciascun file. In particolare i tre file di configurazione indicati vengono letti e concatenati nell'ordine in cui sono mostrati sopra. Quindi prima quello mostrato nell'albero personale (HOME), poi quello dell'albero locale (LOCAL), poi quello in cui si trova la maggior parte della distribuzione TEX Live (DIST). Se esaminando questi file di configurazione un file di mappa fosse ripetuto più volte, il programma conserva solo la prima istanza; analogamente se un font fosse ripetuto più volte in diversi file vengono usate solo le informazioni della prima istanza.

Il file di configurazione nell'albero DIST contiene diverse impostazioni generali e la maggior parte di tutte le mappe della distribuzione; il file di configurazione dell'albero LOCAL, invece, contiene le informazioni sulle mappe dei font aggiunti da chi amministra la macchina. In questo calcolatore il file locale di configurazione contiene solo la riga

```
Map local.map
```

A sua volta la mappa `local.map` che si trova nella diramazione `fonts/map/` dell'albero LOCAL contiene qualcosa come:

```
NebioloGreek          NebioloGreek          <NebioloGreek.pfb
athanasius3           athanasius3           <athanasius3.pfb
athanasius4           athanasius4           <athanasius4.pfb
athanasius5           athanasius5           <athanasius5.pfb
athanasius6           athanasius6           <athanasius6.pfb
cbbohairicmedium      cbbohairicmedium      <cbbohairicmedium.pfb
...                   ...                   ...
uni3                  uni3                  <uni3.pfb
```

Ecco quindi il meccanismo vero: per i font installati localmente, in aggiunta a quelli della distribuzione di TEX Live, basta indicarne le specifiche come illustrato alla fine del paragrafo 18.8.3 o in un file singolo o un file collettivo, dal nome arbitrario (qui chiamato `local.map`), e questo o questi file di mappa per font particolari usati solo localmente devono venire trascritti con il prefisso `Map␣` nel file di configurazione `updmap.cfg` dell'albero LOCAL. I file di configurazione presenti negli altri alberi della distribuzione TEX Live non devono venire toccati per nessun motivo.

18.8.8.2 MiKTeX

Per MiKTeX le cose sembrano più semplici, ma sono anche loro abbastanza delicate, perché nella distribuzione MiKTeX per piattaforme Windows non esiste l'albero locale (non si confonda l'albero locale con l'albero personale) e non è ben chiaro chi sia l'amministratore; ciò premesso, creato un file, per esempio `local.map` come indicato sopra per TeX Live, ma memorizzato nell'albero personale nel ramo `texmf/web2c/`, dopo aver aggiornato il database dei nomi dei file, l'utente può dare il comando:

```
initexmf --edit-config-file updmap
```

che consente di editare il file `updmap.cfg` aggiungendoci in fondo la riga:

```
Map local.map
```

Salvato il file, si dà poi il comando:

```
initexmf --makemaps
```

che aggiorna i file generali di mappa.

Chi scrive da diversi anni non dispone più di una versione di MiKTeX aggiornata, quindi non sa se la prima delle due operazioni sia fragile rispetto all'installazione di una nuova versione di MiKTeX; è possibile; ma da un lato gli upgrade di singoli pacchetti in MiKTeX non implicano la riscrittura di tutto l'albero della sua distribuzione, bensì solo delle cartelle coinvolte da quei pacchetti; dall'altro le nuove versioni di MiKTeX non si succedono annualmente come le nuove versioni di TeX Live. Quindi il problema della sovrascrittura, ammesso che si ponga, è molto dilazionato nel tempo.

18.8.9 I file di descrizione dei font

I file di descrizione dei font, con estensione `.fd` non sono obbligatori, ma sono utili. Si è già visto nel paragrafo 18.8.3 e si è commentato a questo proposito che in quel caso il file di descrizione del font, benché non inutile, non era veramente indispensabile. Tuttavia per casi più complessi è quanto mai opportuno disporre di un file di descrizione di una collezione di font, altrimenti per usarli bisogna ricorrere a comandi nativi o a comandi di basso livello.

Innanzi tutto si richiama il fatto che la selezione dei font con il New Font Selection Scheme (NFSS) che ha rivoluzionato LaTeX 2_ε rispetto al precedente LaTeX 209, ma che ora, dopo più di venti anni di esistenza, non sembra più tanto nuovo, ha definito per ogni font 6 caratteristiche che possono venire selezionate le une indipendentemente dalle altre (vedi però anche il capitolo **??**).

1. La codifica; ne abbiamo già descritte varie con le loro funzioni e caratteristiche.

2. La famiglia: per ogni collezione di font abbiamo visto che si possono avere almeno tre famiglie, quella dei font con grazie, quella dei font senza grazie e quella dei font monospaziati; alcune collezioni di font possono avere anche altre famiglie.
3. La serie: abbiamo visto che ogni font può essere dotato di glifi più o meno "neri" e più o meno evidenziati, per cui disponiamo sempre della serie media e della serie nera estesa, ma potremmo avere la serie condensata, condensata nera, nera senza estensione, eccetera.
4. La forma: abbiamo già descritto la forma diritta, quella inclinata, quella corsiva, quella corsiva diritta, quella maiuscoletta, ma alcune collezioni di font contengono anche altre forme.
5. Il corpo: abbiamo già spiegato che cosa sia il corpo e sappiamo che con il sistema TEX abbiamo la possibilità di usare nello stesso documento corpi piccolissimi, come quello di 5 pt, e corpi piuttosto grandi come quello di 25 pt; in certi casi si possono avere anche corpi decisamente maggiori e la classe *memoir* può addirittura comporre documenti (poster) usando un corpo 'normale' da 60 pt.
6. Lo scartamento: sappiamo che tradizionalmente lo scartamento (erroneamente chiamato anche interlinea), cioè la distanza fra due linee di base successive, è del 20% maggiore del corpo, ma in certe circostanze è opportuno aumentare o diminuire questo scartamento per ottenere effetti speciali. In certi casi lo scartamento potrebbe essere anche leggermente minore del corpo.

Come fa il programma di composizione, nel nostro caso pdflatex, a sapere di quali caratteristiche è dotato il font corrente? Oppure, come fa pdflatex a sapere di quali caratteristiche disponiamo? Lo fa leggendo i file di descrizione dei font.

Esiste un file distinto per ciascuna codifica e ciascuna famiglia di ogni collezione di font. Anzi, il nome del file, con estensione `.fd`, ha il nome in lettere minuscole formato agglutinando la sigla della codifica e il nome della famiglia; qui, in questo capoverso, si sta componendo con la codifica *T1* e con la famiglia dei font con grazie Latin Modern, la cui sigla è `lmr`, per cui le informazioni sui font disponibili con questa codifica e questa famiglia sono contenuti nel file `t1lmr.fd`. Vale la pena vedere che cosa contiene questo file, ma vale anche la pena segnalare subito che i dettagli per l'esatta creazione di questo genere di file sono contenuti nella guida sui font leggibile dal terminale attraverso il comando `texdoc fntguide`. Comunque il file in questione[10] è il seguente.

```
% This file belongs to the Latin Modern package. The work is
% released under the GUST Font License. See the
% MANIFEST-Latin-Modern.txt and README-Latin-Modern.txt files
% for the details. For the most recent version of this license
% see http://www.gust.org.pl/fonts/licenses/GUST-FONT-LICENSE.txt
% or http://tug.org/fonts/licenses/GUST-FONT-LICENSE.txt
```

[10] Le prime 6 righe sono state leggermente riformattate per far rientrare il testo dentro la giustezza di questa guida.

```
\ProvidesFile{t1lmr.fd}[2009/10/30 v1.6 Font defs for Latin Modern]
\DeclareFontFamily{T1}{lmr}{}
\DeclareFontShape{T1}{lmr}{m}{n}%
     {<-5.5>    ec-lmr5     <5.5-6.5> ec-lmr6
      <6.5-7.5> ec-lmr7     <7.5-8.5> ec-lmr8
      <8.5-9.5> ec-lmr9     <9.5-11>  ec-lmr10
      <11-15>   ec-lmr12
      <15-> ec-lmr17
      }{}
\DeclareFontShape{T1}{lmr}{m}{sl}%
     {<-8.5>    ec-lmro8    <8.5-9.5> ec-lmro9
      <9.5-11>  ec-lmro10   <11-15>   ec-lmro12
      <15-> ec-lmro17
      }{}
\DeclareFontShape{T1}{lmr}{m}{it}%
     {<-7.5>    ec-lmri7
      <7.5-8.5> ec-lmri8    <8.5-9.5> ec-lmri9
      <9.5-11>  ec-lmri10   <11->   ec-lmri12
      }{}
\DeclareFontShape{T1}{lmr}{m}{sc}%
     {<-> ec-lmcsc10}{}
\DeclareFontShape{T1}{lmr}{m}{ui}%
     {<-> ec-lmu10}{}
%
% Is this the right 'shape'?:
\DeclareFontShape{T1}{lmr}{m}{scsl}%
     {<-> ec-lmcsco10}{}
%%%%%%% bold series
\DeclareFontShape{T1}{lmr}{b}{n}
     {<-> ec-lmb10}{}
\DeclareFontShape{T1}{lmr}{b}{sl}
     {<-> ec-lmbo10}{}
%%%%%%% bold extended series
\DeclareFontShape{T1}{lmr}{bx}{n}
     {<-5.5>   ec-lmbx5      <5.5-6.5> ec-lmbx6
      <6.5-7.5> ec-lmbx7      <7.5-8.5> ec-lmbx8
      <8.5-9.5> ec-lmbx9      <9.5-11>  ec-lmbx10
      <11->   ec-lmbx12
      }{}
\DeclareFontShape{T1}{lmr}{bx}{it}
     {<-> ec-lmbxi10}{}
\DeclareFontShape{T1}{lmr}{bx}{sl}
     {<-> ec-lmbxo10}{}
%%%%%%% Font/shape undefined, therefore substituted
\DeclareFontShape{T1}{lmr}{b}{it}
```

```
     {<->sub * lmr/b/sl}{}
\endinput
%%
%% End of file `t1lmr.fd'.
```

Dopo le prime righe di informazioni sulle licenze, il codice comincia con la identificazione del file, la sua data e la sua versione. Poi specifica che quanto segue riguarda la "FontFamily" identificata dalla codifica *T1* e dalla famiglia `lmr`. Poi definisce una dozzina di forme abbinate alle loro serie, specificandone i corpi disponibili. La prima dichiarazione definisce i font di serie media (`m`) e di forma normale (`n`), definisce, cioè, il tondo medio. Il quinto argomento elenca i corpi disponibili e i nomi dei file dei font con i quali rappresentare quei corpi. Ogni font reale (*ec-lmr5, ec-lmr6, . . . , ec-lmr17*) può generare corpi diversi contenuti in un intervallo: per esempio l'intervallo indicato con `<9.5-11>` indica tutti i corpi da 9,5 pt incluso a 11 pt escluso. I font più piccoli sono caratterizzati da un intervallo aperto `<-5.5>` che manca dell'estremo inferiore, quindi il font *ec-lmr5* serve per rappresentare i caratteri di corpo piccolissimo (idealmente fino a 'zero') fino al corpo 5,5 pt escluso. I font più grandi sono identificati da un altro intervallo aperto `<15->` per cui il font slshape ec-lmr17 serve per rappresentare tutti i corpi da 15 pt compreso fino a corpi molto grandi (idealmente infinitamente grandi). In questo momento il testo viene composto con il font *T1/lmr/m/n/10* che seleziona mediante la macro `\T1/lmr/m/n/10` il cui significato è `select font ec-lmr10`. Come si vede, questa macro strana, il cui nome contiene anche delle barre e delle cifre, è costruita con il nome della codifica, della famiglia, della serie, della forma e del corpo e serve proprio a selezionare il font *ec-lmr10*; sono tutte informazioni ricavate dal file di descrizione del font corrente.

Si noti la forma `scsl`: essa corrisponde al maiuscoletto inclinato, non direttamente accessibile con i comandi standard di LaTeX, almeno quelli per l'utente; questi per altro potrebbe usare i comandi di basso livello per ottenere IL MAIUSCOLETTO INCLINATO: il comando da usare è:

```
\fontshape{scsl}\selectfont
```

Un'altra forma non direttamente accessibile è la forma corsiva diritta, `ui`, che in questo testo è stato usato diverse volte: *corsivo diritto* con un comando definito apposta, ma sempre accessibile mediante i comandi di basso livello:

```
\fontshape{ui}\selectfont
```

Tra le serie c'è anche il nero non esteso disponibile almeno per le forme normale e inclinata (`sl`) si confronti infatti il **nero non esteso** con il **nero esteso** – in questo caso la differenza non è molto visibile, ma su un testo più lungo è decisamente percepibile. Il nero non esteso è accessibile con i comandi di basso livello:

```
\fontseries{b}\selectfont
```

L'ultima forma abbinata al nero non esteso riguarda il corsivo: ma la collezione non contiene nessun font reale che possa rappresentare il corsivo nero non esteso, quindi lo sostituisce con il font *lmr/b/sl*, cioè con il tondo inclinato nero non esteso. La sostituzione è indicata con la parola chiave `sub` che quando viene attivata, oltre ad eseguire la sostituzione, scrive anche un messaggio nel file `.log`. Esisterebbe anche la parola chiave `ssub` che esegue la sostituzione silenziosamente, cioè senza riportare nulla nel file `.log`.

Come si vede, dunque, quando si usano i comandi di alto o di basso livello per scegliere un font, il programma di composizione va a leggere[11] il file di descrizione per la codifica corrente (o quella specificata e per la famiglia corrente o quella specificata esplicitamente) e poi esso esamina ogni dichiarazione di forma fino a trovare il corpo giusto oppure la sostituzione predefinita.

Un utente che voglia caricare una collezione, o anche una sola famiglia di font, è bene che controlli se esistono già i file di descrizione; in mancanza dovrebbe crearseli se non vuole ridursi ad usare i comandi nativi, come si fa ancora oggi quando si usa Plain TeX.

Il lettore non si spaventi, se dovesse crearsi dei file di descrizione: quelli relativi ai font Latin Modern sono relativamente complessi perché dispone di famiglie molto ricche di forme e di serie; non solo, ma dispone anche di corpi ottici con cui cambiare leggermente il disegno dei glifi omonimi a seconda del corpo in modo da migliorarne la leggibilità per i corpi minori e non annerire troppo i corpi maggiori. Se ne è già discusso (vedi tabella 18.2). Tant'è che il file di descrizione per i font TX è molto più semplice:

```
\ProvidesFile{t1txr.fd}
        [2000/12/15 v3.1]

\DeclareFontFamily{T1}{txr}{}
\DeclareFontShape{T1}{txr}{m}{n}{% medium
     <->t1xr%
}{}
\DeclareFontShape{T1}{txr}{m}{sc}{% cap & small cap
     <->t1xsc%
}{}
\DeclareFontShape{T1}{txr}{m}{sl}{% slanted
     <->t1xsl%
}{}
\DeclareFontShape{T1}{txr}{m}{it}{% italic
     <->t1xi%
}{}
\DeclareFontShape{T1}{txr}{m}{ui}{% unslanted italic
     <->ssub * txr/m/it%
}{}
```

[11] In realtà questa lettura viene fatta una volta sola e il suo contenuto resta memorizzato all'interno del programma.

```
\DeclareFontShape{T1}{txr}{bx}{n}{% bold extended
     <->t1xb%
}{}
\DeclareFontShape{T1}{txr}{bx}{sc}{% bold extended cap & small cap
     <->t1xbsc%
}{}
\DeclareFontShape{T1}{txr}{bx}{sl}{% bold extended slanted
     <->t1xbsl%
}{}
\DeclareFontShape{T1}{txr}{bx}{it}{% bold extended italic
     <->t1xbi%
}{}
\DeclareFontShape{T1}{txr}{bx}{ui}{% bold extended unslanted italic
     <->ssub * txr/bx/it%
}{}
\DeclareFontShape{T1}{txr}{b}{n}{% bold
     <->ssub * txr/bx/n%
}{}
\DeclareFontShape{T1}{txr}{b}{sc}{% bold cap & small cap
     <->ssub * txr/bx/sc%
}{}
\DeclareFontShape{T1}{txr}{b}{sl}{% bold slanted
     <->ssub * txr/bx/sl%
}{}
\DeclareFontShape{T1}{txr}{b}{it}{% bold italic
     <->ssub * txr/bx/it%
}{}
\DeclareFontShape{T1}{txr}{b}{ui}{% bold unslanted italic
     <->ssub * txr/bx/ui%
}{}
\endinput
```

Si nota, infatti, che tutti gli intervalli dei corpi sono aperti da entrambi i lati, nel senso che qualunque corpo, da piccolissimo a grandissimo, viene ricavato rimpicciolendo o ingrandendo i segni messi a disposizione da un unico font reale. Inoltre sono presenti molte sostituzioni silenziose. In sostanza la disponibilità di forme e di serie con in font TX è molto minore di quella che si può avere con i font Latin Modern.

18.8.10 Ricapitolazione

Da quanto esposto si vede dunque che l'installazione di nuovi font estranei alle distribuzioni ufficiali di TEX Live e di MiKTEX non è una operazione che si risolva con quattro click di mouse. Richiede attenzione e precisione, ma non si tratta in generale di operazioni difficili: delicate sì, ma non difficili. Avendo capito quale sia il meccanismo delle mappe e dei font accessibili ai vari programmi di

composizione o di trasformazione, il tutto diventa più facile e la successione delle operazioni diventa del tutto naturale.

Ciò non toglie che non possano sorgere problemi con i sistemi operativi che vengono usati sulle diverse macchine, sistemi operativi che hanno l'abitudine di cambiare assai spesso, talvolta senza retrocompatibilità; chi scrive aggiunge nuovi font abbastanza spesso, certo non tutti i giorni ma almeno una volta all'anno, e trova che il meccanismo operativo oggigiorno sulla distribuzione TeX Live sia ottimo, e non incontra più i problemi che invece doveva affrontare negli anni passati.

Non è quindi il caso di spaventarsi; si possono aggiungere altri font ai moltissimi già distribuiti con le varie installazioni disponibili; per un solo font può essere una questione di un quarto d'ora; per una collezione di font le operazioni possono susseguirsi molto più rapidamente.

18.8.11 Riassunto delle operazioni di gestione dei font e della loro connessione con l'intero sistema TeX

Solo pdflatex

Vale la pena di riassumere graficamente il modo di operare del sistema TeX quando si usi latex o pdflatex. Molte delle informazioni sono sparse in diversi altri capitoli, ma qui, dove si parla di font, sembra il posto giusto per collegare tutto assieme.

Nelle figure 18.9–18.13 i simboli grafici hanno i significati seguenti:

Font I file che si riferiscono ai font in senso stretto sono rappresentati da cerchi gialli.

Programmi di sistema Sono rappresentati da rettangoli rosso chiaro.

Programmi esterni Sono rappresentati da rettangoli azzurri.

File accessori per la gestione dei font Sono rappresentati da cerchi verdi.

File finali Sono rappresentati da cerchi marroncino chiaro; 'file finale' indica il file che contiene un documento già composto.

Risorse esterne Sono gruppi di file di programmi e/o di font da reperire gratuitamente in rete, o onerosamente presso i rivenditori specifici. Sono rappresentati da ellissi blu chiaro.

Uscite L'uscita su schermo, su carta, su lastre per fotolitotipografia, o su altri supporti sono rappresentati mediante ellissi arancione.

Nella figura 18.9 sono rappresentati due dei processi di composizione descritti in questo testo, quello attraverso latex e quello attraverso pdflatex. Si noti la semplicità del processo attraverso latex, a confronto con la relativa complicazione attraverso il processo pdflatex[12]. Il primo richiede solo il file sorgente e i file metrici dei font usati, e in uscita presenta il risultato 'finale' mediante un file `.dvi`. Questo file è finale solo in senso logico, perché, comunque, per essere utilizzato ha ancora bisogno di trasformazioni, specialmente se vi sono state incluse delle immagini ottenute da file esterni.

[12] Questo schema vale anche per XeLaTeXe LuaLaTeX; sono le figure successive che valgono solo per latex o pdflatex.

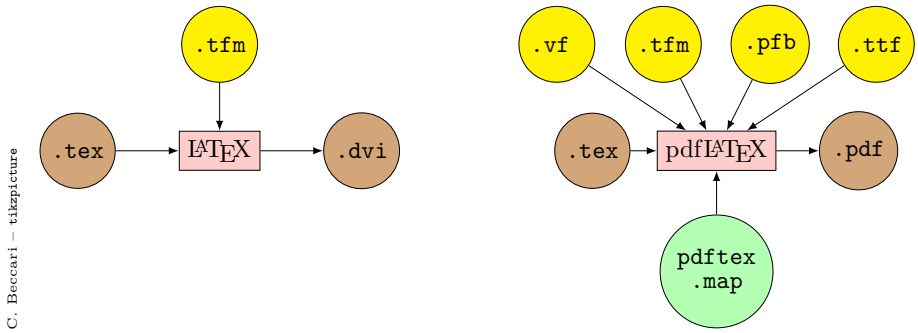


Figura 18.9: Schema grafico del processo di composizione

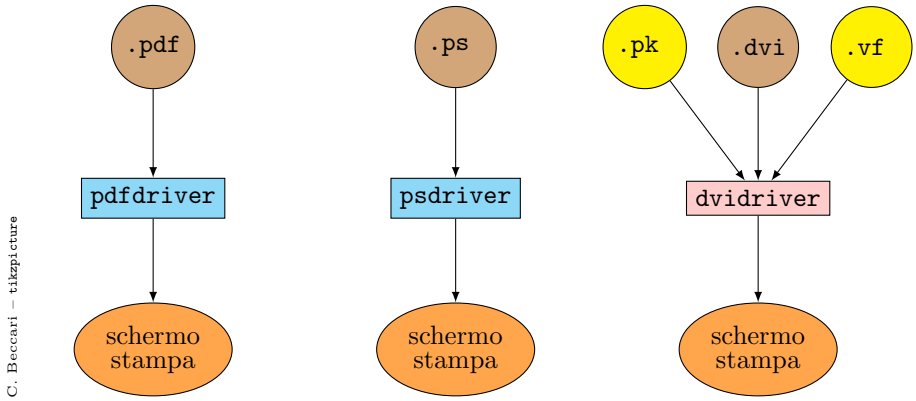


Figura 18.10: Presentazione all'esterno dei risultati della composizione

Al contrario il processo di composizione mediante pdflatex è completo e il file finale si può considerare definitivo; per fare ciò, però, esso ha bisogno di tutti i file concernenti i font, specialmente quelli di tipo vettoriale.

Il file `.dvi` può essere visualizzato e/o stampato direttamente mediante software che solitamente fa già parte del corredo del sistema TeX; sarà il visualizzatore YAP di MiKTeX, oppure xdvi di Linux oppure altri programmi simili. Lo stesso può dirsi del file `.pdf` in uscita da pdflatex, così come del file `.ps` in uscita dal convertitore dvips, nel caso che sia richiesto questo formato per l'uso successivo. Nella figura 18.10 sono rappresentati graficamente i processi di presentazione mediante i vari programmi specifici e sono messi in evidenza i file di cui necessitano, con particolare rilievo per quelli che riguardano i font.

Generalmente però il file `.dvi` ha bisogno di essere trasformato o in file `.pdf` o in file `.ps`; quest'ultimo a sua volta può richiedere la trasformazione ulteriore in file `.pdf`; il processo di conversione è rappresentato graficamente nella figura 18.11, ancora evidenziando l'importanza dei file concernenti i font.

Come si vede da queste figure, i file direttamente o indirettamente legati

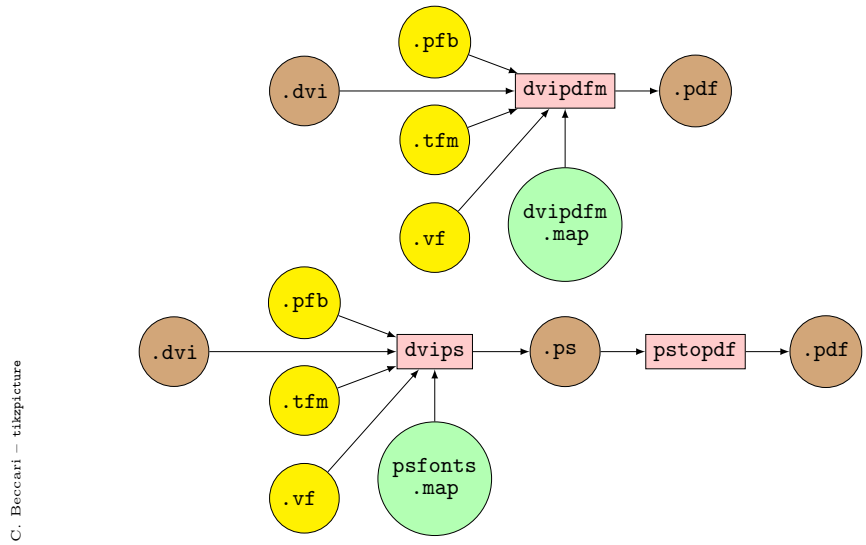


Figura 18.11: Processi di conversione dei vari formati di uscita dai programmi di composizione

ai font sono tanti e spesso bisogna crearli apposta con il supporto delle utility fornite con il sistema TEX o con l'ausilio di programmi esterni. La figura 18.12 mostra in forma schematica le risorse esterne da cui trarre i font necessari e come trasformare questi ultimi in modo da renderli usabili con il sistema TEX. Si vede chiaramente che i font in formato METAFONT sono i più facili da usare, il che è logico, visto che TEX e METAFONT sono programmi 'fratelli' entrambi concepiti da Donald Knuth e sono sempre distribuiti assieme e completi di una buona collezione di file sorgente di tipo `.mf`. Ma i font vettoriali distribuiti secondo lo standard Adobe (PostScript Type 1), oppure i font TrueType, nati da un progetto comune della Adobe e della Microsoft, richiedono ulteriori trattamenti, se non altro perché non sono dotati dei file `.tfm`, che bisogna quindi generare; spesso hanno anche codifiche diverse, certamente diverse da quelle che TEX si aspetta, e quindi sono necessarie conversioni a diversi livelli. I programmi che operano queste conversioni e il loro modo di procedere sono illustrati schematicamente nella figura 18.12; in questa figura è indicata anche la conversione dei font `.mf` in font di tipo vettoriale (mediante il programma mftrace); i font vettoriali sono utilissimi, specialmente per la visualizzazione su schermo dove è necessario poter ingrandire o rimpicciolire la schermata, e quindi il suo contenuto, secondo le necessità di lettura.

In particolare le mappe parziali dei font disponibili sulla macchina specifica sulla quale viene eseguita la composizione vanno unite in modo coerente fino a formare le mappe generali di cui necessitano i programmi di conversione dei file finali. Il procedimento è indicato schematicamente nella figura 18.13; il programma indicato con il nome 'editor' è un qualunque programma di gestione

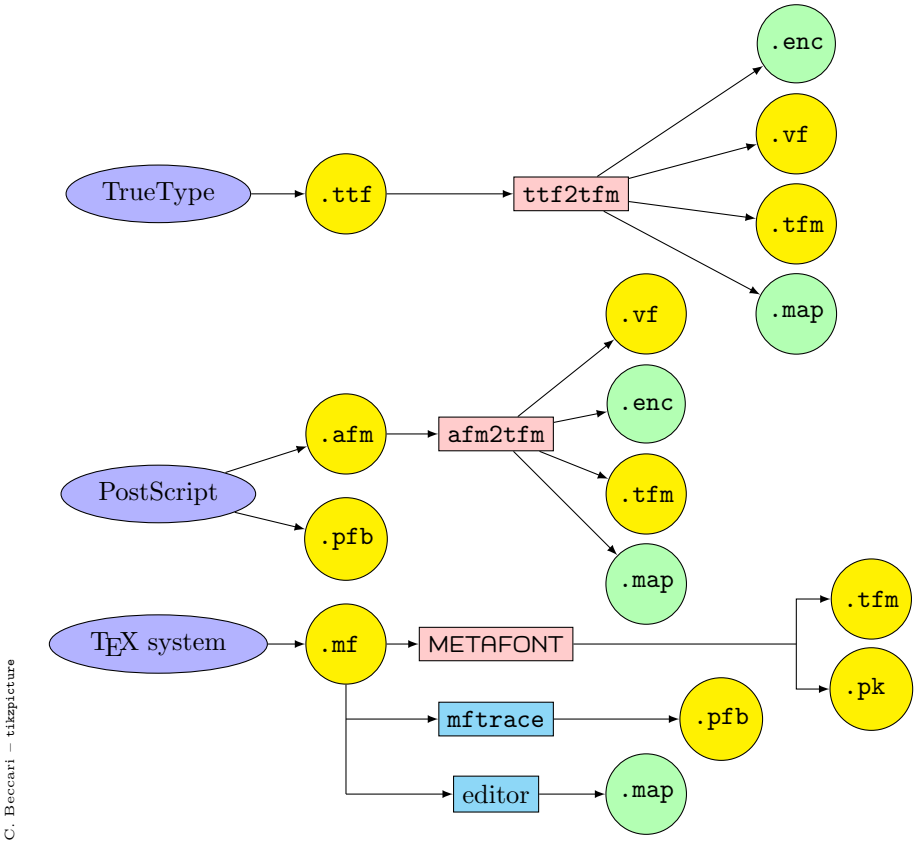


Figura 18.12: Trasformazioni di formato dei font e generazione dei file ausiliari

di file testuali ASCII; potrà essere Notepad dei sistemi Windows, oppure vim dei sistemi Linux, ma va benissimo anche lo *shell editor* usato per gestire i file sorgente da comporre con il sistema TEX; il file `updmap.cfg` in uscita dall'editor è quello che crea l'utente per i suoi font privati; gli altri file con lo stesso nome sono quelli del sistema TEX e non vanno toccati assolutamente dall'utente.

18.9 Conclusioni

Tutte queste operazioni, descritte forse con troppo dettaglio, possono impaurire il neofita; tuttavia ci si può consolare pensando che l'installazione di un nuovo font non è una operazione da eseguirsi tre volte al giorno; d'altra parte se così fosse, queste operazioni finirebbero per venire eseguite in modo del tutto automatico e non apparirebbero così intricate.

Ci si potrebbe domandare perché il sistema TEX, che è dotato di un gran numero di font accessori, non sia dotato anche di un programma che consenta di

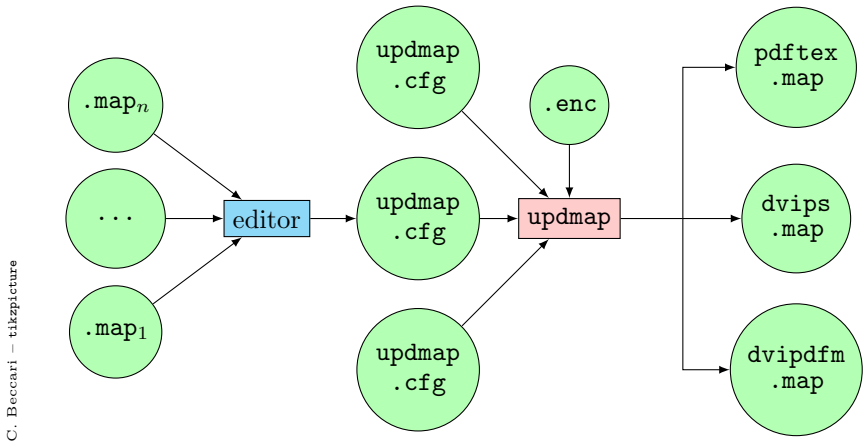


Figura 18.13: Gestione delle mappe dei font

eseguire tutte le operazioni indicate in questi ultimi paragrafi in modo automatico. Buona domanda. Una risposta possibile è che per ogni font o collezione di font bisogna agire in modo diverso da ogni altro font o collezione. I nomi, le codifiche, il contenuto dei file che contengono le informazioni che descrivono i vari font, potrebbero essere forse oggetto di un programma o di uno script in linguaggio bash, oppure python, oppure perl, oppure java, oppure ..., ma le possibilità di errore sono così numerose che il lavoro del programmatore diventerebbe troppo oneroso, specialmente in relazione al fatto che l'installazione di nuovi font non avviene, come si è detto, tre volte al giorno.

Chi scrive generalmente reinstallava i propri font non standard una volta all'anno, quando aggiornava il suo sistema; talvolta aggiungendo un nuovo font, di cui creava a mano la mappa e di cui aggiungeva il nome ad un file personale `local.map`, già elencato nel file di configurazione locale `updmap.map`, e tutto il resto si ripeteva di anno in anno senza bisogno di avere un comportamento diverso per ogni nuovo font.

Oggi le cose sono molto più semplici. Fermo restando che il sistema TeX non può sapere se l'utente ha creato nuovi font, l'utente ha ancora il dovere di elencare i suoi nuovi font nella sua mappa personale. Ma fatto questo ogni nuova installazione annuale del sistema TeX scopre da sola l'esistenza di diversi file di configurazione e li concatena in un unica configurazione per usarla onde aggiornare la mappa generale dei font.

Ad ogni aggiornamento dell'installazione (che anche altrove si è raccomandato di eseguire ogni 7-10 giorni, massimo due settimane) ogni volta che il sistema si arricchisce di nuovi font rigenera la mappa, eseguendo sempre la concatenazione dei vari file di configurazione, per cui la manutenzione delle mappe è diventata una cosa di nessun peso.

Capitolo 19

LaTeX: nuovi comandi

19.1 Introduzione

Uno dei punti di forza del linguaggio di programmazione che sostiene il sistema TeX è costituito dal fatto di poter definire nuovi comandi, nuove istruzioni.

Gli interpreti tex, o pdftex, o xetex, o luatex che interpretano i comandi dei vari tipi di mark-up dei file sorgente sono in grado di eseguire alcune centinaia di comandi base, comandi 'nativi' o 'primitivi'; primitivi non nel senso che siano rozzi, ma che sono quelli a cui tutti gli altri si riferiscono; i comandi derivati sono per lo più definiti nei vari file di formato; `plain.tex` per il mark-up di Plain TeX; `latex.ltx` per il mark-up di LaTeX, eccetera.

Questi file di formato *definiscono* dei nuovi comandi, detti *macro*, attraverso i comandi primitivi o attraverso quelli derivati e già definiti.

Quando l'interprete incontra una macro, la deve sviluppare, cioè deve ricercarne in memoria la traduzione in base alla definizione specifica di quella macro; se questa definizione contiene le azioni da eseguire specificate solamente mediante comandi primitivi, l'interprete li esegue, altrimenti sequenzialmente ricerca i comandi derivati presenti nella definizione, ripetendo il processo finché lo sviluppo contiene solo comandi primitivi che finalmente esegue.

Ogni macro può essere definita in modo che possa lavorare su *argomenti*, cioè parti variabili che cambiano di volta in volta che si invoca la macro. Gli argomenti possono essere *delimitati* o *non delimitati*. All'interno dei formati si fa un grande uso di argomenti delimitati, ma i comandi per gli utenti, specialmente con LaTeX, fanno un uso minimo di argomenti delimitati. Vedi però il capitolo **??** dove si parla dei recenti aggiornamenti e dell'introduzione delle funzionalità del pacchetto *xparse* nel nucleo di LaTeX; grazie a queste funzionalità, a cui ci si riferirà con alcuni esempi in questo capitolo, si possono definire comandi che accettano argomenti variamente delimitati con i quali si possono fare cose incredibili prima di quegli aggiornamenti.

Gli argomenti non delimitati sono costituiti da un solo *token*, per esempio una sola lettera o una sola cifra, e non richiedono di venire racchiusi dentro nessuna struttura che ne limiti la lunghezza. Altrimenti devono essere racchiusi fra parentesi graffe. Nei capitoli precedenti si sono visti quasi sempre argomenti non delimitati indicati nelle istruzioni sintattiche con `{`⟨*argomento*⟩`}`. LaTeX usa raramente gli argomenti delimitati, ma li delimita solo mediante parentesi quadre; è quanto si è indicato con `[`⟨*argomento*⟩`]`; nell'ambiente *picture* sono delimitate le coordinate, racchiuse fra parentesi tonde e indicate nelle costruzioni sintattiche con `(`⟨*x*⟩`,`⟨*y*⟩`)`. Ma per il resto è difficile trovare altri comandi destinati al compositore che facciano uso di argomenti delimitati.

Qui si illustreranno succintamente le definizioni di nuovi comandi, le ridefinizioni di comandi già esistenti; le definizioni di nuovi ambienti e le ridefinizioni di ambienti già esistenti.

Come preannunciato sopra, grazie al nuovo linguaggio L3, LaTeX 3, sono disponibili file di estensione, come *xparse* (documentazione `texdoc xparse`), che permettono agli utenti di definire facilmente i comandi con argomenti delimitati e con diversi tipi di argomenti facoltativi. Merita approfondire leggendo la documentazione di questo pacchetto. Si noti che dal 2020 le funzionalità di questo pacchetto sono integrate nel nucleo di LaTeX, per cui esse sono disponibili senza bisogno di caricarlo.

19.2 Le definizioni di comandi nuovi

Per definire un nuovo comando si usa la seguente sintassi:

```
\newcommand{⟨macro⟩}[⟨num-argomenti⟩][⟨default⟩]{⟨definizione⟩}
\newcommand*{⟨macro⟩}[⟨num-argomenti⟩][⟨default⟩]{⟨definizione⟩}
\providecommand{⟨macro⟩}[⟨num-argomenti⟩][⟨default⟩]{⟨definizione⟩}
\providecommand*{⟨macro⟩}[⟨num-argomenti⟩][⟨default⟩]{⟨definizione⟩}
\DeclareRobustCommand{⟨macro⟩}[⟨num-argomenti⟩][⟨default⟩]{⟨definizione⟩}
\DeclareRobustCommand*{⟨macro⟩}[⟨num-argomenti⟩][⟨default⟩] {⟨definizione⟩}
```

dove ⟨*num-argomenti*⟩ è il numero degli argomenti su cui opera la ⟨*macro*⟩ dei quali il primo può essere facoltativo; esso è facoltativo se viene espressa anche la seconda opzione incluse le sue parentesi quadre; se nell'invocazione della macro che accetta un argomento facoltativo non lo si specifica, esso riceve il valore di ⟨*default*⟩. Nella definizione della ⟨*macro*⟩ gli argomenti, al massimo in numero di 9, sono indicati con il loro numero progressivo `#1`, `#2`, eccetera. La ⟨*macro*⟩, cioè il suo nome, può essere costituita solo:

- dal segno di backslash `\` seguito da una stringa formata solamente da lettere dell'alfabeto latino maiuscole o minuscole, ricordando che LaTeX distingue le une dalle altre; oppure
- dal segno di backslash `\` seguito da un solo segno non alfabetico; oppure

- da un carattere attivo; un carattere attivo è un segno che ha ricevuto in precedenza l'etichetta di carattere attivo; LaTeX dichiara carattere attivo il segno `~` al quale ha dato il significato di *spazio non separabile* (in corrispondenza del quale non può cadere la fine di una riga); `babel` con l'opzione di quasi tutte le lingue che usano caratteri accentati, fra cui l'italiano, definisce attivo anche il carattere `"`.

Il comando `\newcommand`, con o senza asterisco, controlla se la ⟨*macro*⟩ sia già stata definita e, nel caso, emette un messaggio d'errore. Il nuovo comando potrebbe risultare fragile (la fragilità di comandi è un concetto delicato, descritto nel paragrafo 19.6 e descritto anche nel capitolo 29), mentre `\DeclareRobustCommand` crea una definizione robusta, cioè "non fragile"; vedi il paragrafo 19.6. I comandi senza asterisco creano definizioni i cui argomenti possono essere formati da più capoversi, mentre i comandi con l'asterisco creano comandi i cui argomenti possono essere formati anche da diverse parole, ma tali da non costituire un capoverso; nel caso un argomento contenga esplicitamente o implicitamente un comando di "fine capoverso", l'esecuzione del comando con un simile argomento genera un errore.Infine i comandi `\providecommand`, con o senza asterisco definiscono una macro solo se essa non è mai stata definita prima.

Si raccomanda agli utenti di non "giocherellare" con i caratteri attivi; qui li si è nominati perché è giusto sapere che esistono e che cosa sono, ma è meglio lasciarli stare. È meglio lasciare stare anche i comandi che usino solo o anche caratteri non alfabetici, come per esempio il carattere `@`.[1]

19.2.1 Definizione del logo di MacTeX

Come primo esempio si riporta quanto è stato definito nel preambolo di questo documento, la definizione del comando per scrivere il logo di MacTeX (vedi i comandi robusti nel paragrafo 19.6):

```
\DeclareRobustCommand*\MacTeX{Mac\-\TeX}
```

Come si vede non ci sono argomenti né obbligatori né facoltativi per la definizione della macro `\MacTeX`; la ⟨*definizione*⟩ si avvale di altri comandi: `\-` per consentire una cesura dopo la prima parte del logo, e `\TeX` per comporre la seconda parte formata dal logo di TeX.

19.2.2 Definizione del comando `\cs`

Come secondo esempio si porta la definizione del comando `\cs` che è servito tante volte per scrivere i nomi dei comandi con carattere a spaziatura fissa e

[1] Nei file di formato compaiono una quantità di definizioni di comandi che contengono il carattere `@` come se fosse una lettera; all'interno del formato questo carattere è definito come lettera alfabetica, ma fuori del formato, quando il compositore usa LaTeX, esso è ripristinato alla categoria di 'non-lettera'; la cosa è fatta di proposito affinché i compositori non si mettano a giocare con i caratteri non alfabetici.

preceduti dal segno di backslash senza che LaTeX li eseguisse, ma li considerasse solo delle stringhe letterali:

```
\newcommand*{\cs}[1]{\texttt{\char92#1}}
```

Il comando accetta un solo argomento obbligatorio e la sua sintassi diventa perciò

`\cs{`⟨*nome della macro senza backslash*⟩`}`

La definizione della macro[2], oltre a contenere il comando `\texttt` per comporre il suo argomento con caratteri a spaziatura fissa, contiene anche il comando primitivo `\char` che si aspetta un numero decimale, oppure esadecimale, oppure ottale; dalle tabelle dei caratteri si vede che è più semplice riferirsi all'indirizzo decimale, invece che a quelli ottale o esadecimale, e si vede che il segno di backslash è nella casella con indirizzo 92; usando il comando `\char` e l'indirizzo nella tabella dei caratteri, il segno identificato viene trascritto tale e quale, prescindendo dal significato che esso potrebbe avere per LaTeX, cioè quello di iniziatore di nomi di macro: `\cs` è diverso da `\char92cs` il primo è un comando, il secondo è il nome di un comando.[3]

19.2.3 Il comando \\

Nel preambolo di questo documento non sono stati definiti comandi con argomento facoltativo; si prenderà un esempio da altri documenti; prendiamo una possibile definizione del comando `\\`:

```
\newcommand*{\\}[1][0pt]{\newline\vspace{#1}}
```

Questa non è la definizione completa che compare all'interno del file di formato, ma rende l'idea. Il comando `\\` accetta un argomento; siccome ne è specificato il valore di default, questo primo ed unico argomento è facoltativo; se lo si vuole indicare lo si deve scrivere fra parentesi quadre. Se non lo si indica affatto, viene usato il suo valore di default che viene passato tale e quale alla macro `\vspace` che esegue una spaziatura verticale pari all'ammontare specificato con l'argomento esplicito o di default; la spaziatura verticale di default è di 0 pt, cioè una spaziatura nulla.

[2] In realtà la definizione indicata è solo una versione preliminare; quando si è deciso di comporre anche l'indice analitico, si è cambiata la definizione includendovi il comando `\index` corredato di un opportuno argomento; la necessità di usare il segno `@`, che ha un significato particolare per makeindex, ci ha obbligati ad una definizione decisamente più complessa, la cui spiegazione esula da questo testo. Tuttavia è importante notare come la modifica della sola definizione di `\cs` ha permesso di inviare all'indice analitico tutti i comandi descritti, senza bisogno di inserire il comando `\index` ogni volta che li si citava.

[3] Invece di `\char92` si sarebbe potuto scrivere `\textbackslash`; ma questo comando agisce solo in modo testo, mentre `\char92` agisce anche in modo matematico. È vero che non lo si è mai usato in modo matematico, ma non si sa mai...

19.2.4 i comandi di definizione asteriscati

Il lettore attento avrà notato che in due dei tre esempi un asterisco segue il comando `\newcommand`; questo asterisco facoltativo esprime una variante del comando che è consigliabile usare ogni volta che il comando debba ricevere degli argomenti; come già detto, ma lo si evidenzia di nuovo, senza asterisco il comando può ricevere argomenti arbitrariamente lunghi, fatti anche di numerosi capoversi; con l'asterisco il comando può ricevere argomenti formati solo da una stringa di caratteri che non contenga né esplicitamente né implicitamente una terminazione di capoverso; questa come si ricorda, è costituita da una riga del file sorgente completamente vuota oppure dal comando esplicito `\par`.

Quando si definiscono comandi che accettano solo una o poche parole, certamente non più di un capoverso, è prudente inserire l'asterisco, perché nel caso che ci si dimentichi di scrivere la graffa di chiusura, viene emesso subito un messaggio d'errore che consente al compositore di intervenire per correggerlo. È bene ricordare che l'omissione di una graffa chiusa rappresenta l'errore più frequente quando si compone il testo sorgente.

19.2.5 I capilettera

Nel capitolo 20 si parlerà dei *capilettera*; vi si descriverà il pacchetto *lettrine* che consente di inserire queste grandi lettere maiuscole all'inizio del primo capoverso di un capitolo o di un paragrafo. Nell'esemplificare l'uso dei capilettera in quel capitolo non si è fatto uso delle macro disponibili con il pacchetto *lettrine* ma si è usato il comando `\capolettera` descritto qui di seguito insieme alle motivazioni per non aver usato il pacchetto citato.

La macro `\capolettera` è definita nel file `MacroGuida.sty` che raccoglie le macro usate per comporre questo testo. Questa macro `\capolettera` non ricorre al pacchetto *lettrine* perché sul forum di GUIT è stato notato che *lettrine* funziona bene solo con i font vettoriali che dispongono solamente di un corpo da ingrandire e rimpicciolire a seconda delle esigenze. Qui si usano in prima istanza i font Latin Modern (scalabili con continuità a tratti) o, in seconda istanza, i font Computer Modern[4], che sono dotati di tutti i corpi necessari. Si è preferito quindi definire una macro apposita che svolgesse più o meno le stesse funzioni offerte dal pacchetto *lettrine*, ma non facesse riferimento implicito ad un solo corpo. La macro qui descritta produce un capolettera grande come due righe di testo e costruito in modo da essere a filo con la linea di base della seconda riga, ma in modo che la sua altezza non superi quella delle lettere in maiuscoletto che la seguono. Il suo scopo, infatti è quella di cominciare un capoverso nel modo seguente.

UN GRANDE CAPOLETTERA deve essere molto visibile come inizio di un capoverso ma richiede una transizione di forma dei caratteri che lo seguono

[4] I font Computer Modern non sono scalabili, ma sono disponibili solo in certi corpi; però, usando il pacchetto *type1ec*, *senza* specificargli l'opzione *10pt*, i font CM-Super sono scalabili con continuità a tratti.

per rendere meno brusco il passaggio dalle sue grandi dimensioni al corpo normale del font usato nel resto del capoverso. Per questo motivo la tradizione tipografica generalmente ricorre a qualche parola iniziale del capoverso, dopo il capolettera, composta in maiuscoletto.

La macro perciò richiede di specificare come primo argomento la lettera da ingrandire e, come secondo argomento, il completamento della prima parola seguito da qualche altra parola iniziale del capoverso:

```
\newlength\CLett              % Nuova dimensione

\newcommand*\capolettera[2]{% #1: lettera da ingrandire
                            % #2 testo da comporre in maiuscoletto
\setbox\z@\hbox{\scalebox{1.25}{\huge#1}}\CLett=\wd\z@%
\par\noindent
\hangindent\CLett\hangafter-2\relax
\raisebox{-\baselineskip}[\z@]\z@]{%
          \llap{\box\z@\kern1pt}}\textsc{#2}}
```

La definizione ricorre a comandi di base del linguaggio TeX, mescolati a comandi di LaTeX e del pacchetto *graphicx*. Per i dettagli si può guardare nel capitolo 29. Il capolettera si ottiene mettendo nella scatola numerata '0' (`\z@`) la lettera da ingrandire ma che viene inizialmente ottenuta con il comando `\huge` e poi viene ulteriormente ingrandita del 25%, per tenere conto che si tratta di una maiuscola (generalmente) senza discendenti.

Se ne prende la misura della larghezza per far rientrare le righe della giusta quantità; ma poi sono `\hangindent` e `\hangafter` che creano lo spazio nelle prime due righe per mettere la lettera ingrandita; `\hangindent` riceve come argomento l'ammontare del rientro e `\hangafter` riceve il numero di righe per le quali vale il rientro; il segno si riferisce al margine destro (segno più) o sinistro (segno meno) per il quale si esegue l'operazione del rientro. Alla fine del capoverso questi due parametri 'primitivi' di TeX vengono ripristinati a valori nulli, cosicché queste impostazioni valgono solo per il capoverso corrente.

Per collocare la lettera si usa il comando LaTeX `\raisebox` che mette il suo argomento in una scatola, e la solleva della quantità specificata nel primo parametro (se essa è negativa la abbassa) e tratta la scatola come se avesse altezza e profondità pari ai due ulteriori parametri facoltativi (in questo caso entrambi nulli, cosicché non si tiene conto dell'altezza e, specialmente, della profondità della lettera ingrandita). Questa lettera viene inserita dentro una scatola di larghezza nulla `\llap`, ma con il contenuto sporgente a sinistra; vi viene posto anche un poco di spazio (1 pt) per scostare un minimo la lettera ingrandita dalle rientranze delle due righe. Il secondo argomento viene poi inserito tutto composto in maiuscoletto.

Il pacchetto *lettrine* consente di corredare il comando del capolettera con diverse opzioni, cosicché lo si possa fare sporgere in alto di più o di meno, in modo che occupi lo spazio di un numero di righe a scelta, e altre finezze, ma

sembra, appunto, che funzioni correttamente solo con i font dotati di un solo carattere di corpo fisso, ma scalabile a piacere.

19.2.6 Un titolo di sezione da trattare in modo speciale

Un altro esempio istruttivo si presenta quando è necessario usare un comando di sezionamento che inserisca qualcosa di diverso dal solito sia nel suo titolo, sia in quello che viene elencato nell'indice, sia in quello che appare nelle testatine. I comandi normali del nucleo di LaTeX dispongono di un argomento facoltativo indicato con la dicitura ⟨*titolo breve*⟩:

\⟨*comando*⟩[⟨*titolo breve*⟩]{⟨*titolo lungo*⟩}

dove \⟨*comando*⟩ può essere `\part`, `\chapter`, `\section`, eccetera. Il ⟨*titolo breve*⟩ appare sia nella testatina sia nell'indice generale, ma non è possibile con questi comandi mettere due ⟨*titoli brevi*⟩ uno nell'indice e l'altro, diverso, nella testatina.

Non è il solo, ma uno dei motivi per fare questo è che se si usa il pacchetto *hyperref* i segnalibri ricevono lo stesso testo che va nell'indice, ma *hyperref* non è capace di gestire nei segnalibri alcuni comandi fragili e/o alcuni comandi che cambiano la famiglia, serie o forma dei caratteri. Questo è successo in questo testo, per esempio, nella composizione del paragrafo 11.5; dove il nome del pacchetto *biblatex* doveva apparire in minuscolo nella testatina, e doveva apparire in tondo minuscolo nei segnalibri, mentre nell'indice poteva apparire nello stile dei nomi dei pacchetti. Impossibile ottenere questi risultati con le macro predefinite.

Tuttavia è possibile farlo a mano, e se lo si fa a mano con una sequenza di operazioni collaudate, questa serie di operazioni possono essere costituire la definizione di una nuova macro. Si è accennato sopra che sarebbe possibile usare i potentissimi comandi di definizione del pacchetto *xparse*; qui si espone la strategia con un comando che si riferisce al comando `\section`, ma, *mutatis mutandis*, la stessa strategia si può applicare anche agli altri comandi.

Si è deciso di usare il comando `\section*`, che scrive solo il titolo non numerato del paragrafo e non manda nulla né all'indice, né alla testatina. Assieme a questo comando, si usano i comandi di basso livello per inviare materiale diverso all'indice e alle testatine delle pagine destre occupate dall'intero paragrafo.

Si è trovata definita la macro seguente:

```
\newcommand\Section[3]{%
% #1 := parte fissa del titolo della sezione
% #2 := comando particolare da applicare alla terza parte
% #3 := parte da trattare in modo particolare
\refstepcounter{section}%
\section*{\thesection\quad#1 #2{#3}}%
  \addcontentsline{toc}{section}{%
    \protect\numberline{\thesection}{#1
    \ifdefined\texorpdfsring\expandafter\@firstoftwo\else
```

```
10    \expandafter\@secondoftwo\fi{\texorpdfstring{#2}{}}{}{#3}}}%
11  \markright{\MakeUppercase{#1} #2{#3}}%
12 }
```

In questa macro i tre argomenti hanno i significati specificati nelle righe di commento; ma con questa macro, il titolo del paragrafo 11.5 è stato composto semplicemente con

```
\Section{Il pacchetto}{\packstyle}{biblatex}
```

È necessaria qualche spiegazione.

1. Nella riga 1 si specifica che la macro deve ricevere tre argomenti obbligatori, i cui significati sono spiegati nelle righe 2, 3 e 4.
2. Nella riga 5 si incrementa di una unità il contatore delle sezioni; infatti, benché dopo si usi il comando `\section*` per gestire il titolo della sezione, si desidera che la sezione sia numerata e che si possa fare riferimento al suo numero mediante il solito meccanismo dei riferimenti incrociati che usano `\label` e `\ref`. Il comando `\refstepcounter` incrementa di una unità il contenuto del contatore `section` e lo rende accessibile al meccanismo dei riferimenti incrociati.
3. Il comando `\section*` non numera il paragrafo di cui scrive il titolo; ma se si vuole una sezione numerata bisogna provvedere esplicitamente ad inserire il numero prima del titolo alla giusta distanza; questa, nel nostro caso, è un quadrato (`\quad`).
4. Nelle righe 7—10 si provvede a inviare all'indice quanto vi deve venire scritto; per questo scopo si usa il comando `\addcontentsline` la cui sintassi è ricordata più avanti nel capitolo 29. Il suo primo argomento specifica con la sigla `toc` il file nel quale scrivere le informazioni; il secondo argomento specifica il livello logico di quello che vi si vuole inserire, ma il terzo argomento contiene un comando `\protect` che protegge da uno sviluppo intempestivo quanto segue, in particolare il comando `\numberline` con i suoi argomenti; questo comando scrive una voce nell'indice che comincia con un numero, prosegue con un titolo che, con una fila di puntini, guida l'occhio verso il numero della pagina appoggiato al margine destro dello specchio di stampa.
5. All'interno del terzo argomento di `\numberline` c'è un costrutto insolito, perché si esegue un test mediante `\ifdefined` per verificare che il comando `\texorpdfstring` sia definito; se lo è lo usa, se non lo è non lo usa. Sembra lapalissiano, ma non lo è. Infatti il test eseguito con `\ifdefined` segue la stessa sintassi di tutti i comandi condizionali nativi di TeX. Questi hanno il difettuccio di restare a mezz'aria a seconda di cosa contengano i due rami del test; il primo, il test è vero, finisce con `\else` e il secondo, il test è falso, finisce con `\fi`. Per evitare questa apparente "debolezza" di questi comandi condizionali, i rami vero e falso vengono sostituiti da `\@firstoftwo` e `\@secondoftwo`, ma con `\expandafter` viene invertito

l'ordine di esecuzione dei comandi; cioè nel ramo vero, benché `\else` sia scritto dopo `\@firstoftwo`, esso viene eseguito prima; la stessa cosa avviene nel caso falso, dove `\fi` viene eseguito prima di `\@secondoftwo`. Con questo artificio, usato a piene mani all'interno di ogni pacchetto, classe, file di formato, eccetera, del sistema TEX, si è sicuri che l'operazione condizionale non resti a mezz'aria. Successivamente ogni "frase" condizionale `\if...\else...\fi` deve essere seguita da due gruppi, dei quali il primo contenga i comandi da eseguire nel caso vero e il secondo quelli da eseguire nel caso falso. Le due macro interne `\@firstoftwo` e `\@secondoftwo` Servono appunto per leggere i due gruppi per restituire inalterato solo il primo o solo il secondo.

6. Nella macro in esame il comando `\texorpdfstring` è fornito dal pacchetto *hyperref*; esso accetta due argomenti: il primo serve per l'indice vero e proprio, mentre il secondo serve per i segnalibri; in questo caso esso scrive nell'indice l'argomento `#2` (cioè, nell'applicazione specifica, il comando `\packstyle`), ma non scrive niente nel file ausiliario che raccoglie le informazioni per i segnalibri; il terzo argomento racchiuso fra graffe, funziona da argomento solo per il comando contenuto nell'argomento `#2` (cioè `\packstyle`), ma questo terzo argomento viene scritto senza modificazioni e senza graffe nel file dei segnalibri. Il comando `\texorpdfstring` è utilissimo, ma gli utenti spesso se ne dimenticano, e vanno incontro ad errori di compilazione talvolta difficili da correggere.
7. Visto che in questo testo il pacchetto *hyperref* è stato caricato, il comando `\texorpdfstring` è sicuramente definito, quindi tutto quel codice per verificarne l'esistenza e provvedere in merito sembra inutile; lo si è messo comunque, perché la macro potrebbe essere recuperata per la composizione di altri documenti che ne avessero bisogno, ma non è detto quegli altri documenti facciano uso di collegamenti ipertestuali e che quindi carichino il pacchetto *hyperref*.
8. Finalmente nella riga 11 si arriva al testo per le testatine; `\markright` crea il materiala da inserire nella la testatina della pagina di destra, che normalmente contiene il titolo del paragrafo corrente alla fine della pagina. Nello stile delle pagine predefinito nella classe *book*, usata per comporre questo testo, le testatine sono composte in tondo inclinato maiuscolo; ma qui non si può scrivere il nome del pacchetto *biblatex* in maiuscolo inclinato, ma si deve rendere in maiuscolo solo la prima parte della testatina; lo stile standard delle pagine provvede all'inclinazione dei caratteri.

19.2.7 Scalare il corpo

Componendo un documento con font scalabili con continuità eventualmente a tratti, si può ricorrere al comando `\setfontsize` descritto in questa sezione; Per cambiamenti di corpo globali, validi per l'intero documento si potrebbe usare il pacchetto*fontsize*, alla cui documentazione si rimanda il lettore; ma talvolta è utile disporre di un comando semplicissimo che permetta di scalare il corpo del

font in uso senza ricorrere alle dichiarazioni di corpo predefinite; ovviamente, come per quelle dichiarazioni, il comando va usato dentro un gruppo affinché il suo effetto sia locale. In questo testo lo si è usato relativamente spesso per poter ridurre la lunghezza delle righe di testo all'interno di ambienti contenenti testi speciali, come la sintassi dei comandi o negli ambienti verbatim contenenti qualche riga un poco più lunga della giustezza.

La sintassi del comando è la seguente:

```
\setfontsize{⟨corpo⟩}[⟨fattore⟩]
```

dove il ⟨*corpo*⟩ è il corpo desiderato e il ⟨*fattore*⟩ è il coefficiente che determina lo scartamento a partire dal ⟨*corpo*⟩; questo fattore ha un valore di default pari a 1.2, che corrisponde al fattore di (quasi) tutti i font usati dal sistema TeX. Per definire questo comando è comodo usare non solo la sintassi specificata dalle funzionalità del pacchetto *xparse* già integrate nel nucleo di LaTeX, ma anche alle funzionalità del pacchetto *xfp* che però va caricato espressamente nel preambolo; questo pacchetto serve per eseguire calcoli in virgola mobile con una grande precisione; molto più avanzato dei calcoli che si possono eseguire con il pacchetto *calc*.

La definizione è la seguente:

```
\NewDocumentCommand\setfontsize{m O{1.2}}{%
    \fontsize{#1}{\fpeval{#2*#1}}\selectfont}
```

Come si vede dalla lista dei descrittori degli argomenti, il primo descrittore si riferisce ad un argomento obbligatorio (m: mandatory), mentre il secondo ad un argomento facoltativo (O: optional) con un valore di default pari a 1.2. La macro `\fpeval` serve per calcolare il valore dell'espressione contenuta nel suo argomento; nel nostro caso si tratta del prodotto (`*`) del ⟨*fattore*⟩ (il secondo argomento `#2`) con il ⟨*corpo*⟩ desiderato (il primo argomento `#1`).

Il risultato che si ottiene è che questa frase è composta in corpo 9.5 mentre questa frase è composta in corpo 10.5. La composizione di questo capoverso è stata eseguita avendo specificato il comando `\setfontsize{10}[1.5]`, cioè usando lo stesso corpo usato lungo tutta la guida, ma con uno scartamento corrispondente solo al 50% in più del corpo, invece del solito 20%.

19.3 Ridefinizione di comandi già esistenti

Per ridefinire un comando già esistente, ammesso che il compositore sappia che cosa stia ridefinendo e ne capisca tutte le possibili conseguenze, si usa il comando `\renewcommand` con la sintassi seguente:

```
\renewcommand{⟨macro⟩}[⟨num-argomenti⟩][⟨default⟩]{⟨definizione⟩}
\renewcommand*{⟨macro⟩}[⟨num-argomenti⟩][⟨default⟩]{⟨definizione⟩}
```

I cui parametri hanno lo stesso significato che per `\newcommand` e al quale si può aggiungere l'asterisco come si faceva con `\newcommand`.

Il caso più frequente della ridefinizione di comandi esistenti riguarda il caso della modifica di comandi propri oppure di comandi definiti nei file di classe o di estensione.

In questi casi è *vietato* modificare i file originali, ma si procede come indicato nel paragrafo 6.7.

Va da sé che se si cerca di ridefinire un comando che non esiste, l'istruzione `\renewcommand` emette un perentorio messaggio d'errore e si rifiuta di procedere oltre.

19.3.1 Ridefinizione della parole fisse

Un uso abbastanza frequente consiste nel ridefinire le parole che compaiono durante l'esecuzione di comandi di sezionamento. Per esempio il comando `\listoffigures` esegue l'elenco delle figure come un indice a sé stante che viene introdotto da un titolo diverso a seconda della lingua usata. In italiano questo titolo è "Elenco delle figure"; questa stringa è contenuta nella definizione della macro `\listfigurename`. Per cambiare questo titolo, per esempio in "Lista delle figure", basta allora ridefinire la macro nel modo seguente:

```
\renewcommand*{\listfigurename}{Lista delle figure}
```

Naturalmente questo implica che il testo venga scritto in una sola lingua e che la ridefinizione sia eseguita dopo il comando di inizio del documento, quando la lingua di default è già stata specificata. Bisogna poi stare attenti a non cambiarla più; in particolare, se *babel* fosse chiamato con un elenco di lingue, perché il documento contiene citazioni in lingue diverse da quella di default, bisogna evitare accuratamente di usare il comando `\selectlanguage` per evitare di rieseguire la definizione dei comandi di default della lingua principale, il che renderebbe vana la ridefinizione appena illustrata. Se si deve scrivere davvero un documento in molte lingue con diversi elenchi delle figure scritti e intitolati in lingue diverse, allora *babel* consente di usare altri comandi più potenti, come `\addto`[5], che permettono di essere più efficaci nel gestire queste cose.

Questo avviso serve appunto a segnalare come anche una ridefinizione 'innocente' come quella proposta può causare risultati non previsti, anche se non sono dannosi per l'integrità di LaTeX. Il compositore che desidera usare `\renewcommand` deve sempre sapere veramente quello che sta facendo.

[5] Le parole contenute in apposite macro, come `\listfigurename`, che a loro volta sono contenute dentro una macro il cui nome è ottenuto dall'agglutinazione della parola `captions` con il nome della lingua; per l'italiano, dunque, questa macro si chiama `\captionsitalian`. Il comando `\addto`, pertanto, riceve come primo argomento la macro `\captionsitalian` e come secondo argomento la ridefinizione di `\listfigurename`. Così facendo ogni volta che si specifica la lingua italiana, la macro che contiene il titolo dell'elenco delle figure viene ridefinito di nuovo, ma questa operazione ridondante è un male minore rispetto a quanto descritto nel testo se si eseguisse la ridefinizione senza aggiungerla alle altre macro per l'italiano.

19.3.2 Ridefinizione del comando \XeLaTeX

Il comando \XelaTeX ha la prima "E" che appare capovolta o riflessa, a seconda di come la si guardi.

Usando i font OpenType non ci sono problemi ad usare questo tipo di "E", ma quei font possono essere usati solo con xelatex e lualatex.

Nello stesso tempo il nucleo di LaTeX non contiene una macro per scrivere XeLaTeX. Bisogna caricare il pacchetto *metalogo* che permette di usare una varietà di loghi, configurabili a seconda dei font usati.

Anche questo pacchetto deve distinguere come comporre il logo XeLaTeX a seconda che il programma di composizione sia pdflatex o uno degli altri due. Nel caso di pdflatex, che può usare solo font con 256 segni, l'apposito segno della "E" va in qualche modo simulato, perché nessun font Type 1 lo contiene. In quel pacchetto si usa il comando \reflectbox del pacchetto *graphicx* per scambiare la destra con la sinistra, vale a dire come riflettere la "E" in uno specchio verticale. Tutto bene finché non si usano caratteri inclinati, come il corsivo, perché la "E" inclinata e riflessa nello specchio appare inclinata nel verso sbagliato. Si confronti "E" e la sua immagine "Ǝ" se si usa un carattere diritto; con "*E*" la sua immagine è "*Ǝ*" se si usa un carattere inclinato.

La soluzione a questo problema consiste nel controllare se il carattere in uso è inclinato e nel ricorrere a una rotazione di 180° attorno al suo centro. In questo documento, composto con pdflatex il problema è stato risolto ridefinendo il comando \XeLaTeX nel modo seguente:

```
\renewcommand*\XeLaTeX{X\ifdim\fontdimen1\font=0pt\kern-0.15em\fi
  \lower.5ex\hbox{\rotatebox[origin=c]{180}{E}}}
```

In un secondo momento si è notato che il comando poteva presentare delle fragilità e si è preferito sostituire \renewcommand con \DeclareRobustCommand così da disporre di un comando sicuramente robusto; vedi il paragrafo 19.6. Nella macro il test \ifdim verifica quanto valga l'inclinazione del font corrente; per farlo bisogna ricorrere ai comandi di LaTeX descritti nella sua guida fntguide.pdf relativa ai font; essa è accessibile con il comando da terminale texdoc fntguide; la prima dimensione del font corrente indica l'inclinazione del font in punti tipografici riferiti ad un punto tipografico in orizzontale, e la si misura con quella espressione \fontdimen1\font. Se detta inclinazione è nulla bisogna arretrare il segno di una frazione di em, mentre se il font è obliquo detto arretramento non è necessaria; invece è comunque necessario abbassare la lettera ruotata, e lo di fa con il comando nativo di TeX \lower che abbassa la scatola indicata con un altro comando nativo \hbox. Procedere in questo modo serve per semplificare un poco il codice interno, ma il risultato è lo stesso di quello che si otterrebbe con il comando \raisebox{-0.5ex}{...}. Ora quando si usa il comando \XeLaTeX si ottiene XeLaTeX e in corsivo *XeLaTeX*; solo un occhio molto esercitato riconosce la piccola imperfezione dovuta al fatto che la "E" non è riflessa, ma è ruotata. D'altra parte con pdflatex non si può fare di meglio.

19.4 Ridefinizioni di comandi di sistema

La ridefinizione dei comandi di sistema è una operazione da evitare se non si sa esattamente quello che si sta facendo; si rischia di buttare all'aria tutto il funzionamento dei programmi di composizione. Si veda anche quanto si è scritto nel paragrafo 6.7.

19.4.1 I numeri romani maiuscoletti

Per eseguire la scrittura dei numeri romani minuscoli il sistema di default usa esternamente il comando `\roman`, usabile dal compositore, ma internamente il comando del compositore viene eseguito da `\@roman`; il comando comincia con il segno `@` che per il programma non è una lettera e quindi difficilmente il compositore può andare a 'disturbare' i comandi interni di sistema. Tuttavia il modo esiste, ed è descritto da Leslie Lamport in persona nel suo manuale. Bisogna usare il comando `\makeatletter`[6] prima di ridefinire un comando interno; questa istruzione cambia i codici interni che descrivono la funzione del segno @ in modo da poterlo usare come se fosse una lettera dell'alfabeto.

Questo risolve il primo problema. Ma per scrivere in maiuscoletto non basta specificare `\textsc` perché questa forma non è definita per tutte le famiglie e per tutte le serie. Bisogna accontentarsi di simulare la forma maiuscoletta con lettere maiuscole di corpo più piccolo.

La ridefinizione di `\@roman` diventa allora la seguente:

```
\makeatletter
\DeclareRobustCommand*{\simulatedSC}[1]{%
\hbox{$\relax$\fontsize{\sf@size}{\f@baselineskip}\selectfont#1}}%
\renewcommand*{\@roman}[1]{\simulatedSC{\@Roman#1}}%
```

Una spiegazione però è necessaria. I comandi `\sf@size` e `\f@baselineskip` sono i comandi interni che contengono rispettivamente il corpo degli indici primi e dello scartamento associati al font corrente; il primo dei due è definito solo dopo che sia stata scritta una formula, qui ridotta a nulla (`\relax`); essi sono documentati insieme ad altri comandi dello stesso genere nella già citata guida `fntguide` di cui si è già detto. Il comando `\fontsize` è quello di base per impostare un nuovo corpo e un nuovo scartamento; ma questi non diventano operativi se non dopo aver dato il comando esplicito `\selectfont`; infine il comando `\@Roman` è il comando interno per scrivere i numeri romani in lettere maiuscole. Ciò premesso la ridefinizione che stiamo cercando di fare richiede un comando `\simulatedSC` con il quale si ordina di scrivere con l'avanzamento di riga corrente, ma con il corpo degli indici primi; per limitare i suoi effetti e per essere sicuri che il numero romano, cadendo in fin di riga, non venga eventualmente diviso in sillabe, il tutto viene eseguito dentro una scatola orizzontale `\hbox` che successivamente

[6] Se la nuova definizione viene inclusa nel proprio file di macro personali, avente estensione `.sty` che viene richiamato con il comando `\usepackage`, non è necessario servirsi del comando `\makeatletter`.

viene usata come un unico oggetto, non come una stringa di lettere/cifre romane; usando questo comando la ridefinizione di `\@roman` viene eseguita in modo da prendere il numero romano scritto in lettere maiuscole e di stamparlo nel corpo degli indici primi. Qui si mostra come la pagina corrente possa essere scritta in numeri romani maiuscoli e in numeri romani maiuscoletti simulati in corpi diversi[7]:

Corpo 'normalsize': CDLXVIII CDLXVIII

Corpo 'Huge': CDLXVIII CDLXVIII

Ad esaminare bene il problema, tenendo conto che se si usa il pacchetto *hyperref*, che non è di bocca buona quando si definiscono o ridefiniscono comandi che lui tratta in modo particolare per creare i riferimenti ipertestuali, bisogna evitare di usare comandi personali nei punti in cui *hyperref* interviene, ci si accorge che la definizione suddetta può andare bene in contesti molto semplici, ma non in generale.

La soluzione generale non è quella di ridefinire il modo di comporre i numeri romani quando questi possono essere usati da *hyperref*, quindi tutte le volte che si maneggiano i numeri delle pagine. Bisogna inserire il comando per usare o per simulare il maiuscoletto solo nei punti in cui il numero viene effettivamente composto, quindi nelle testatine, nell'indice analitico, e nell'indice generale; inoltre il comando di simulazione del maiuscoletto potrebbe essere usato anche in altre circostanze, come, per esempio, nel comporre il nome del programma BibTeX, il cui logo prevede l'uso del maiuscoletto nelle seconde due lettere ib. Esistono infatti dei font che non solo non hanno il maiuscoletto nero, ma non hanno nemmeno il maiuscoletto *tout court*.

Allora per le testatine bisogna ridefinire i comandi per lo stile della pagina o delle pagine iniziali dove queste potrebbero essere numerate in numeri romani minuscoli; per questo testo[8] è stato necessario ridefinire lo stile `headings`, chiamandolo `frontmatter` e lo stile `plain` chiamandolo `plainfrontmatter`. Semplicemente si sono copiate le definizioni di `plain` e di `headings` e si è aggiunto il comando `\textsc` per modificare la composizione del numero della pagina, indicato dal comando `\thepage`:

```
% Stile delle pagine iniziali con il numero di pagina in maiuscoletto
\def\ps@frontmatter{%
    \ps@headings
      \def\@evenhead{\textsc{\thepage}\hfil\slshape\leftmark}%
      \def\@oddhead{\slshape\rightmark\hfil\textsc{\thepage}}%
```

[7] Caricando la lingua greca, anche se non è la lingua principale, e il pacchetto *hyperref*, quanto spiegato non funziona, perché quel modulo per il greco e quel pacchetto modificano la definizione del comando `\@roman` in modo incompatibile.

[8] In realtà tutte queste ridefinizioni erano state necessarie nelle prime edizioni di questo testo, quando nella *front matter* si usava la numerazione romana delle pagine. Successivamente, come è stato spiegato altrove, si è deciso di numerare tutte le pagine della guida con cifre arabe (ridefinendo i comandi `\frontmatter` e `\mainmatter`) e il problema è sparito del tutto.

```
    }

\def\ps@plainfrontmatter{%
     \let\@mkboth\@gobbletwo
     \let\@oddhead\@empty
     \def\@oddfoot{\reset@font\hfil\textsc{\thepage}\hfil}%
     \let\@evenhead\@empty
     \let\@evenfoot\@oddfoot
    }
```

Per l'indice analitico non si è fatto nulla di particolare sul modo di comporlo, ma si sono modificati i comandi con i quali i pochissimi lemmi presenti nelle pagine iniziali sono stati indicizzati.

Per l'indice generale la cosa è stata più complessa: il comando che genera la riga per le sezioni a livello di paragrafo o inferiore non presentano nessun problema perché sono scritti in un font di serie media. Il comando in questione è `\@dottedtocline`, che va modificato inserendo la dichiarazione per l'uso del maiuscoletto, come si vede nella terzultima riga del codice seguente:

```
\def\@dottedtocline#1#2#3#4#5{%
  \ifnum #1>\c@tocdepth \else
    \vskip \z@ \@plus.2\p@
    {\leftskip #2\relax \rightskip \@tocrmarg
     \parfillskip -\rightskip
     \parindent #2\relax\@afterindenttrue
     \interlinepenalty\@M
     \leavevmode
     \@tempdima #3\relax
     \advance\leftskip \@tempdima \null\nobreak\hskip -\leftskip
     {#4}\nobreak
     \leaders\hbox{$\m@th
        \mkern \@dotsep mu\hbox{.}\mkern \@dotsep
        mu$}\hfill
     \nobreak
     \hb@xt@\@pnumwidth{\hfil\normalfont\scshape \normalcolor #5}%
     \par}%
  \fi}
```

Questa operazione, tra l'altro semplicissima, potrebbe essere evitata usando il pacchetto *tocloft*, che serve per personalizzare il modo di comporre gli indici, purché i font da usare abbiano la serie e la forma desiderate.

Per la riga che serve per comporre nell'indice i riferimenti ai capitoli, che è composta con un font della serie nera, bisogna agire con un comando di simulazione; infatti il font Latin Modern usato in questo testo, non dispone del maiuscoletto nero. A nulla varrebbe ricorrere a pacchetti come *tocloft*, se poi il font da usare non contiene la serie o la forma giusta. Bisogna pertanto

mettere le mani nei comandi interni. La stessa cosa che qui farò per comporre la voce relativa ai capitoli dovrebbe venire ripetuta per comporre le altre voci che appaiono in neretto, come per esempio le voci che riguardano le parti; siccome questo testo non è diviso in parti (è diviso in tomi, però), mi accontento di modificare solo le voci dei capitoli.

La riga che definisce il modo di comporre la voce dei capitoli si chiama `\l@chapter`; visto che il font usato in questo testo non contiene il maiuscoletto nero, bisogna usare la simulazione mediante il comando `\simulatedSC`, ma con apposite varianti rispetto a quanto esposto precedentemente:

```
\renewcommand*\l@chapter[2]{%
  \ifnum \c@tocdepth >\m@ne
    \addpenalty{-\@highpenalty}%
    \vskip 1.0em \@plus\p@
    \setlength\@tempdima{1.5em}%
    \begingroup
      \parindent \z@ \rightskip \@pnumwidth
      \parfillskip -\@pnumwidth
      \leavevmode \bfseries
      \advance\leftskip\@tempdima
      \hskip -\leftskip
      #1\nobreak\hfil \nobreak
      \hb@xt@\@pnumwidth{\hss\simulatedSC{#2}}\par
      \penalty\@highpenalty
    \endgroup
  \fi}
```

Ecco, ora il comando `\simulatedSC` deve svolgere il suo compito correttamente, e per questo deve agire solo sulle lettere minuscole, non sulle cifre o sulle lettere maiuscole; la definizione data precedentemente non fa queste distinzioni, quindi ci vogliono definizioni nuove. Innanzi tutto ridefiniamo il `\@roman` in modo che la ridefinizione venga usata solo nella parte principale e non nella parte iniziale del testo:

```
\renewcommand\mainmatter{%
  \cleardoublepage
  \@mainmattertrue
  \pagenumbering{arabic}%
  \def\@roman##1{\expandafter\simulatedSC
                 \expandafter{\romannumeral##1}}%
  }
```

e notiamo che invece di ricorrere a `\@Roman`, come fatto precedentemente, usiamo il comando primitivo `\romannumeral` il cui argomento, sia esso un numero o un contatore, viene espresso mediante la stringa di lettere minuscole che ne rappresentano la scrittura in numeri romani ma, attenzione, queste lettere non

sono di categoria 11, come lo sono abitualmente le lettere lette dal programma di composizione dai file sorgente, ma sono di categoria 12 (segni analfabetici); quindi non ne possiamo controllare la categoria, visto che in questo modo lettere e cifre hanno la stessa categoria. Bisogna aggirare questo problema con mezzi che potrebbero ricorrere ai comandi di ε-TEX, oppure ai nuovi comandi avanzati di LaTeX 3, ma qui si mostra un modo per superare questo scoglio che ricorre a semplici applicazioni del linguaggio primitivo di TEX. Definiamo una nuova variabile booleana `lettere` per sapere se la stringa da convertire in maiuscoletto simulato sia formata da lettere, e impostiamo il valore di questa variabile mediante un comando che prende la stringa da convertire, ne controlla la prima (e forse unica) lettera, e se è una lettera minuscola imposta il valore 'vero' per la variabile booleana:

```
\newif\iflettere\letterefalse

\def\testlettere#1#2!{\count256=‘#1
    \ifnum\count256<‘a \letterefalse
    \else
        \ifnum\count256>‘z \letterefalse
        \else\letteretrue
        \fi
    \fi}
```

La definizione di `\testlettere` viene eseguita con un comando primitivo, perché usa argomenti delimitati; in questo caso la lista degli argomenti è delimitata a destra da un punto esclamativo; `\testlettere` riceve la stringa da convertire, ne estrae la prima lettera nell'argomento `#1`, e le altre (eventualmente nessuna) fino al delimitatore costituito dal punto esclamativo, le assegna all'argomento `#2`; si presuppone che la stringa da convertire sia formata solo da lettere; per fare le cose meglio bisognerebbe usare ricorsivamente questo comando su tutta la stringa e lo si potrebbe fare passandole di nuovo la stringa contenuta in `#2` ma, con questa riserva mentale, ci accontentiamo di supporre che la stringa da controllare sia formata solo da lettere e ne controlliamo solo la prima. Poi il valore numerico dell'indirizzo interno della lettera contenuta in `#1` viene assegnata al contatore 256 (valore consentito da quando pdflatex contiene le estensioni di ε-TEX); questo si ottiene usando il 'backtick' (o virgoletta semplice aperta o accento grave, purtroppo assente dalla tastiera italiana, con la quale bisogna arrangiarsi immettendo il codice 96 con il tastierino numerico e il tasto Alt); il backtick è un comando primitivo di TEX che consente di estrarre il codice del carattere a cui esso è premesso; infatti poco dopo si usano `‘a` e `‘z` per avere i codici interni della lettera 'a' e della lettera 'z'. Ecco quindi che con un doppio test numerico si controlla se la lettera contenuta in `#1` è esterna all'intervallo delle lettere minuscole latine `a...z`, nel qual caso la variabile booleana `lettere` viene posta falsa, altrimenti è vera, cioè la lettera controllata è una lettera minuscola.

A questo punto possiamo definire `\simulatedSC` in modo che agisca solo sulle lettere minuscole:

```
\DeclareRobustCommand*{\simulatedSC}[1]{%
  \testlettere#1!\iflettere\textormath{%
  \hbox{$\relax$%
  \fontsize{\sf@size}{\f@baselineskip}\selectfont\uppercase{#1}}%
  }{%
  \scriptstyle\mathrm{#1}}%
  \else#1\fi}%
```

Il meccanismo è sempre lo stesso; si determina il corpo dei pedici di primo ordine in relazione al font in uso e con quello si compone in maiuscoletto simulato se si è in modo testo; si noterà però che a causa del comando `\testlettere` questa operazione è fatta solo se stiamo trattando una stringa di lettere minuscole (almeno se la prima lettera della stringa è una lettera minuscola) altrimenti la stringa viene lasciata inalterata.

Questa serie di definizioni, inserite nel file `MacroGuida.sty`, mi ha permesso di scrivere correttamente BibTeX, anche se la famiglia di font senza grazie non dispone del maiuscoletto; mi ha permesso (nelle edizioni precedenti di questa guida) di comporre correttamente l'indice generale, le testatine e i piedini della pagine iniziali di questo testo.

Tanto fumo per così poco? Ebbene sì, ma la cosa dipende principalmente dal fatto che i font Latin Modern sono privi di maiuscoletto nero. Se si fossero usati dei font non standard, per esempio attraverso il pacchetto *fourier*, contenenti il maiuscoletto nero, tutto ciò si sarebbe potuto evitare e si sarebbe potuto usare il pacchetto *tocloft* che con semplici comandi di impostazione avrebbe permesso di ottenere gli stessi risultati.

19.4.2 La virgola intelligente

Un altro esempio usato in questo testo è costituito dalla ridefinizione della virgola. È noto che in tutte le lingue, tranne l'inglese, le norme ISO prescrivono come separatore decimale la virgola. In questo modo in tutte le altre lingue la virgola svolge due ruoli, ma solo in matematica. Quando si è in modo matematico il segno della virgola può essere dichiarato attivo e gli si può associare una definizione. Qui si è fatto esattamente così:[9]

[9] Attenzione con le virgolette `"`: se i comandi che seguono vengono emessi quando è già in vigore l'opzione *italian* di *babel*, l'interprete quando legge `"8000` segnala un errore; per convenienza e semplicità di scrittura sarebbe meglio sostituire in quella definizione la notazione esadecimale con quella ottale: `'100000`; più semplicemente ancora si può scrivere `\string"8000`. Si faccia anche attenzione al 'backtick' `` ` ``, che non è un accento ma il carattere ASCII 96; la scrittura di questo carattere può essere difficile con la tastiera italiana e con sistema operativo Windows; in un laptop che non abbia il tastierino numerico separato (quasi tutti) bisogna affidarsi alle prestazioni dello *shell editor*; in un PC con la tastiera munita di tastierino numerico separato dal resto, il backtick può venire introdotto premendo sul tastierino il numero 96 mentre si tiene premuto il tasto Alt. Purtroppo la tastiera italiana non è di grande aiuto; per fortuna non capita spesso di dover usare il backtick.

19.4.2.1 La virgola intelligente che riconosce lo spazio

Definiamo una semplice soluzione al problema della virgola che capisca da sola se in matematica deve comportarsi da segno di punteggiatura o da separatore decimale; Basta definire il glifo della virgola come attivo solo in matematica e dargli una definizione per riconoscere se nel testo sorgente è o non è seguito da uno spazio.

```
\DeclareMathSymbol{\virgola}{\mathpunct}{letters}{"3B}
\DeclareMathSymbol{\virgoladecimale}{\mathord}{letters}{"3B}
\AtBeginDocument{\mathcode`\,="8000}
{\catcode `,=\active \gdef,{\futurelet\let@token\m@thcomma}}

\def\m@thcomma{%
   \ifx\let@token\@sptoken \virgola
   \else \virgoladecimale\fi
}
```

Precisamente si sono definite due entità matematiche: `\virgola` che rappresenta un segno matematico di punteggiatura e `\virgoladecimale` che rappresenta la virgola decimale; la differenza risiede nel codice interno che il carattere `"3B` (la virgola) riceve attraverso i comandi `\DeclareMathSymbol`, visto che nel primo caso è specificato appartenente alla categoria `\mathpunct` (punteggiatura matematica), mentre nel secondo caso esso è definito come `\mathord`, cioè come un normale simbolo o una variabile matematica. La specifica `letters` in entrambi i casi dice che il segno deve essere tratto dalla stessa polizza di caratteri dai quali si traggono i simboli letterali (il font corsivo matematico).

Le definizioni del codice attivo e della sua definizione diventano attive solo all'inizio del documento mediante `\AtBeginDocument`; vi si dice che il codice matematico deve essere posto a `"8000`, lo speciale codice che definisce attivo un carattere in modo matematico. Gli si assegna poi attraverso il comando primitivo `\gdef` una definizione globale nella quale esso deve assegnare a `\let@token` non il significato del token successivo, che sarebbe `\m@thcomma`, ma quello del token che incontrerà ancora dopo; allora e solo allora potrà eseguire `\m@thcomma`. Questo a sua volta è definito in modo da esaminare il tipo di token il cui significato è stato attribuito a `\let@token`; se questo è uno spazio (`\@sptoken`) usa la virgola di punteggiatura, altrimenti usa la virgola come simbolo. I comandi `\ifx`, `\else` e `\fi` servono appunto per eseguire questi comandi condizionali. Sta ora al compositore lasciare uno spazio dopo la virgola (in matematica) solo quando questa rappresenta un segno di punteggiatura, mentre non deve lasciare nessuno spazio se deve comportarsi da separatore decimale.

Il risultato di questo comando è visibile in ogni numero decimale fratto scritto in questo testo[10], ma qui è opportuno confrontare bene che cosa si ottiene se in matematica si scrive `f(x,y)` oppure `f(x,␣y)`: $f(x,y)$ nel primo caso, ma $f(x, y)$ nel secondo. La differenza è piccola, ma si vede benissimo.

[10] Sebbene in effetti in questo testo si sia usata una delle definizioni descritte qui di seguito.

19.4.2.2 La virgola intelligente che riconosce se è seguita da una cifra

Tuttavia ci si può domandare se per caso non sia possibile fare a meno dell'inserimento di uno spazio dopo la virgola, quando questa ha il significato di segno di interpunzione; sì, è possibile, ma bisogna definire la virgola attiva in modo un poco più complesso:

```
\def\m@thcomma{\let\@tempB\virgola
\@tfor\@tempA:=0123456789\do{%
\expandafter\ifx\@tempA\let@token\let\@tempB\virgoladecimale
\@break@tfor\fi}\@tempB}
```

che si basa su una macro interna del nucleo di LaTeX, `\@tfor`, la quale ripete un certo insieme di comandi tante volte quanti sono i token che seguono l'indicazione `:=`. Nel nostro caso questi token sono costituiti dalle dieci cifre decimali; l'argomento del comando `\do`[11] viene ripetuto assegnando successivamente uno dei token della lista alla macro `\@tempA`, quindi viene ripetuto al massimo dieci volte. Se il token seguente la virgola nel testo sorgente, memorizzato nella sequenza `\let@token`, è uguale al token via via memorizzato nella macro `\@tempA`, cioè è un cifra decimale, allora il valore di default di `\@tempB`, inizializzato con la `\virgola` interpuntiva, viene cambiato nel valore attributo alla virgola decimale, `\virgoladecimale`, e al termine della macro ripetitiva viene eseguito il comando `\@tempB`. Gli altri comandi presenti nel corpo della macro ripetitiva, come `\expandafter` o `\@break@tfor`, servono per scopi di programmazione particolari: `\expandafter` è strumentale per eseguire il confronto fra il contenuto di una macro e un token implicito e, per poterlo fare, bisogna prima sviluppare la macro; `\@break@tfor` serve per interrompere le iterazioni del comando ripetitivo non appena si è scoperto che il token implicito è una cifra.

Usando questa definizione per `\m@thcomma`, invece di quella precedentemente indicata, il "tastierista" non deve ricordarsi di inserire spazi dopo la virgola, tranne in un unico caso, quando dovesse scrivere una lista di numeri, come in:

```
...␣per␣ogni␣$i=1,␣2,␣3,\dots,n$
```

che produce "... per ogni $i = 1, 2, 3, \ldots, n$". Si vede chiaramente che prima di n c'è lo stesso spazio che precede le cifre, anche se non c'è lo spazio nel codice sorgente. Se si fosse omesso lo spazio fra la virgola e la cifra successiva si sarebbe ottenuto invece: "... per ogni $i = 1,2,3, \ldots, n$", che è palesemente errato.

19.4.2.3 La virgola intelligente che riconosce il codice di categoria del carattere successivo

Ci sarebbe qualche altro modo per evitare di dover ricorrere a comandi interni del nucleo di LaTeX o a magie del linguaggio primitivo di TeX? Sì, si potrebbe

[11] La sequenza di controllo `\do` sembra un comando perché è seguita da una coppia di graffe che contengono ciò che si deve eseguire; in realtà `\do` è solo un segno per delimitare a destra il primo argomento di `\@tfor`, cioè le dieci cifre, delimitate a sinistra da `:=`.

fare, anche se è necessario comunque ricorrerete ai comandi primitivi `\ifx`[12], `\else` e `\fi`, abbastanza chiari nel loro semplice inglese anche a chi non conosce la programmazione condizionale. Bisogna definire un comando ausiliario:

```
\def\vic@code{%
\ifx\let@token0\number1\else
\ifx\let@token1\number1\else
\ifx\let@token2\number1\else
\ifx\let@token3\number1\else
\ifx\let@token4\number1\else
\ifx\let@token5\number1\else
\ifx\let@token6\number1\else
\ifx\let@token7\number1\else
\ifx\let@token8\number1\else
\ifx\let@token9\number1\else
\number0\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi}
```

e poi definire:

```
\def\m@thcomma{\ifnum\vic@code>0\relax
  \virgoladecimale\else\virgola\fi}
```

Da un punto di vista pratico il risultato è lo stesso di quello ottenuto con la seconda definizione; ora però il codice è più facilmente comprensibile ed è anche più efficiente. I dieci confronti fra `\let@token`, (che, ricordiamo, è il carattere implicito equivalente al segno che segue immediatamente la virgola nel codice sorgente) con le dieci cifre decimali sembra più lungo di quello eseguito con il ciclo `\@tfor`, ma è molto efficiente. Il motivo è estremamente tecnico e lo si tralascia; tuttavia non è difficile capire che per gestire il ciclo `\@tfor` il motore di composizione deve sviluppare ripetutamente diverse macro che non appaiono nella versione scritta dall'utente, ma che sono nascoste nel nucleo di LaTeX. Questa terza soluzione, invece, ricorre solo a comandi primitivi e non deve sviluppare niente; inoltre uno dei pregi dei comandi condizionali è che sono costruiti in modo tale da eliminare i rami "falsi" dalla lista di token da scandire in fase di input, per cui alla fine il motore di composizione si ritrova solamente a inserire nella lista dei token che `\ifnum` deve esaminare, solo uno 0 o un 1, avendo eliminato tutti i rami risultati "falsi" di quella lunga serie di `\ifx`.

19.4.2.4 La virgola intelligente che riconosce se il carattere successivo ricade nella lista dei caratteri ASCII dele cifre

Sembra strano, ma non c'è nessun comando primitivo né una macro che permetta di stabilire direttamente se un token sia una cifra o un altro segno. Tuttavia si

[12] Il significato di `\ifx` è diverso da quello di `\if`; entrambi esistono, ma il secondo sviluppa gli oggetti da confrontare prima di confrontarli, mentre il primo confronta i loro significati senza svilupparli.

potrebbe ricorrere a questa quarta soluzione che ricorre sempre a comandi primitivi e a *sporchi trucchi* del linguaggio primitivo TeX; richiede, però, conoscenze più approfondite:

```
\def\m@thcomma#1{%
\unless\ifcat\noexpand\let@token*%
  \virgola
  \expandafter\expandafter\expandafter
      \ifcsname\expandafter\@gobble\string#1\endcsname
    \virgola
  \else
    \ifnum`#1<`0\relax
      \virgola
    \else
      \ifnum`#1>`9\relax
        \virgola
      \else
        \virgoladecimale
      \fi
    \fi
  \fi
\fi#1}
```

Questa soluzione richiede solo quattro comandi condizionali; il primo `\ifcat` serve per verificare il codice di categoria di due token[13]; perciò `\ifcat` controlla se il token memorizzato in `\let@token` abbia lo stesso codice di categoria dell'asterisco; questo è un segno 'qualunque', dell'*alfabeto* ASCII, che ha lo stesso codice di categoria delle cifre; `\let@token` invece potrebbe contenere una cifra (codice di categoria 12), uno spazio (codice di categoria 10) un carattere attivo o un comando (codice di categoria 13), una lettera (codice di categoria 11) un segno analfabetico come l'asterisco, le parentesi, i segni 'più' o 'meno', eccetera (codice di categoria 12 come le cifre), un altro carattere implicito[14].

Allora se il codice di categoria *non* è 12 (quello dell'asterisco) la virgola non è certo seguita da una cifra e deve essere resa con la virgola di punteggiatura `\virgola`; altrimenti bisogna riconoscere se si tratti di un altro carattere implicito e lo si fa con quella lunga serie di comandi intercalati con sequenze di `\expandafter` che servono per 'invertire' l'ordine di esecuzione dei vari comandi: prima `\string`, poi `\@gobble` poi `\ifcsname`; in questo modo si prende l'argomento `#1`, lo si converte in una stringa di caratteri, gli si toglie il primo carattere, e si verifica se quello che resta può essere il nome di una sequenza di controllo; le cifre non sono assegnate con caratteri impliciti, ma potrebbe essere un carattere (come, per esempio, `\infty`) che potrebbe essere stato reso equivalente mediante

[13] Siccome non sappiamo se `\let@token` contenga un carattere normale o un comando sviluppabile, lo facciamo precedere da `\noexpand` così da evitarne lo sviluppo.

[14] Cioè potrebbe essere un'altra sequenza di controllo a cui è stato assegnato il significato di un carattere mediante `\let`.

`\let` o un altro comando primitivo legato a un carattere matematico; in questo caso non si può procedere al riconoscimento di una cifra con la "magia" del segno ‘ che si può applicare ai caratteri espliciti; bisogna poi riconoscere se si tratti di una cifra. Si noti che il primo token diverso da uno spazio che segue la virgola è stato assorbito dal comando `\m@thcomma` come suo argomento; il primo test controlla se il codice ASCII di questo token (`‘#1`) precede il codice ASCII dello zero (`‘0`): se il test è vero non si tratta di una cifra e ci vuole la virgola di punteggiatura. Bisogna però fare un altro test dello stesso genere per verificare se il codice ASCII del token segue il codice ASCII della cifra nove (`‘9`), nel qual caso ci vuole la virgola interpuntiva, altrimenti siamo in presenza di una cifra e si usa la `\virgoladecimale`. Tenuto conto che i comandi condizionali primitivi eliminano dal flusso di ingresso i rami falsi, questo algoritmo è quello che consente la massima efficienza fra i quattro esposti.

19.4.3 Comandi da eseguire solo per determinate lingue

Prima di continuare il commento su queste macro, vale la pena di sottolineare che raramente con il linguaggio primitivo, o con il mark-up di LaTeX, un problema di programmazione ha una sola soluzione; questi problemi hanno praticamente sempre una moltitudine di soluzioni e tutto sta nel trovare quella più robusta, o quella più efficiente, o quella che, insomma, di volta in volta sembra la migliore sotto punti di vista diversi.

Componendo in inglese non si deve eseguire nessuna definizione strana della virgola; quindi se si vogliono inserire questi comandi in un file di macro personali, è meglio racchiuderli dentro gli argomenti del comando di *babel* `\iflanguage` che ha la seguente sintassi:

`\iflanguage{`⟨*lingua*⟩`}{`⟨*vero*⟩`}{`⟨*falso*⟩`}`

dove ⟨*lingua*⟩ è il nome dell'opzione passata a *babel* come lingua di default, mentre ⟨*vero*⟩ corrisponde alle azioni da eseguire se la lingua di default è quella specificata, altrimenti si eseguono le azioni indicate da ⟨*falso*⟩.

Tuttavia in realtà non c'è bisogno di fare niente, perché in inglese la virgola verrebbe usata come punteggiatura, quindi non sarebbe mai seguita da una cifra (se non nelle liste di valori numerici) e come separatore decimale verrebbe usato il punto il cui codice matematico non è stato cambiato; quindi scrivendo in inglese basterebbe avere le stesse attenzioni che si hanno in italiano, salvo usare il punto come separatore decimale.

19.4.4 Approfondimenti

Questi esempi ricorrono a comandi primitivi e ai codici di categoria matematici; non si tratta di concetti adatti ad una introduzione, ma non sono nemmeno concetti inaccessibili; il lettore che voglia approfondire deve esaminare e studiare il TeXbook, dove questi argomenti sono trattati in diversi capitoli; si veda qui

anche il capitolo 21. Conviene anche documentarsi sul nucleo di LaTeX, sebbene la documentazione sia piuttosto essenziale; la si legge dando da terminale il comando `texdoc source2e`.

19.4.5 Lo spazio per il numero nelle liste delle tabelle e delle figure

Un altro esempio, tratto dalle piccole modifiche eseguite per la composizione di questo testo, è un comando per dare un formato alle righe degli elenchi delle figure e delle tabelle. Il comando di formato di queste righe è specificato nel file della classe, nel nostro caso di `book.cls`; vi si legge

```
\newcommand*\l@figure{\@dottedtocline{1}{1.5em}{2.3em}}
...
\let\l@table\l@figure
```

Si osserva innanzi tutto che il comando per la formattazione delle righe della lista delle tabelle, `\l@table` viene reso equivalente alla definizione appena data al comando per la formattazione delle figure, `\l@figure`. Se quindi cambiamo il secondo, dobbiamo ricordarci di eseguire nuovamente l'equivalenza dei comandi.

Per quel che riguarda la formattazione di queste righe vediamo che il comando `\l@figure` si affida all'esecuzione del comando `\@dottedtocline`, il cui nome ricorda il fatto che queste righe degli indici sono di solito fornite della fila di puntini che guidano l'occhio dal titolo al numero della pagina. I parametri che questo comando riceve sono in realtà cinque, ma gli altri gli vengono passati dagli altri comandi che servono per recuperare e stampare le informazioni da inserire nell'indice, cioè il titolo (nel nostro caso la didascalia (breve) della figura o della tabella) e il numero della pagina. Ma nel listato di sopra i tre parametri hanno i significati seguenti: (*a*) il primo dice che la riga deve essere composta come quella dei paragrafi (livello 1); (*b*) il secondo rappresenta il rientro della riga; si nota infatti che la riga dei paragrafi nell'indice di questo testo è rientrata, quella dei sotto paragrafi è rientrata di più, eccetera; (*c*) il terzo parametro specifica lo spazio da destinare al numero specifico della figura o della tabella oltre allo spazio che divide questo numero dall'inizio del titolo riportato nell'indice. Ecco, questo terzo valore va benissimo se il numero del capitolo non è superiore a 9 o se il numero della figura o della tabella non è superiore anch'esso a 9, cioè se almeno uno dei due sia di una sola cifra. In questo testo alcuni dei capitoli con un numero di due cifre contengono più di nove figure, e non resta nessuno spazio fra il numero e il titolo; il che sta evidentemente male. Lo spazio di `2.3em` è insufficiente, quindi bisogna aumentarlo di quanto basta per metterci un'altra cifra; `1em` è largo praticamente come una lettera 'M', e un cifra è larga circa la metà (la cifra 1 graficamente sembra più stretta, ma in realtà insieme agli spazi che la contornano occupa lo stesso spazio delle altre; è voluto, perché così le tabelle di numeri risultano incolonnate anche se contengono delle cifre 1). Basta allora ridefinire il comando di sistema così:

```
\renewcommand*\l@figure{\@dottedtocline{1}{1.5em}{2.8em}}
\let\l@table\l@figure
```

ricordando di inserire queste ridefinizioni in un file di macro personali (a causa del segno `@`) e di non dimenticare l'equivalenza fra `\l@table` e `\l@figure`.

19.5 Esiste già o non esiste ancora il comando?

Talvolta si vuole essere sicuri che un comando sia disponibile ma non si sa se esso sia già stato definito in qualche pacchetto di estensione. Allora si può usare il comando `\providecommand` già descritto all'inizio di questo capitolo.

Si ricorre spesso a questo tipo di definizione mediante `\providecommand` nel proprio file di macro personali, cosicché il comando definito attraverso questa istruzione viene sempre definito se il comando non è già stato definito da altri, altrimenti questa definizione viene tranquillamente ignorata e viene usata la definizione già esistente. Logicamente perché questo funzioni come desiderato, è necessario che il proprio file di macro personali sia sempre invocato per ultimo.

Nel preambolo di questo testo, per esempio, si è provveduto il comando `\ohm` nel modo seguente

```
\providecommand*{\ohm}{\textormath{\textohm}{\mathrm{\Omega}}}
```

dove `\textormath` è un comando fornito dal pacchetto *babel* per specificare due azioni diverse da eseguire in modo testo o in modo matematico; in modo matematico si è specificato di usare il font tondo diritto, per ovviare al caso in cui si sia deciso di usare le lettere greche maiuscole inclinate nelle normali espressioni matematiche.

19.6 Definizione di comandi robusti

Un comando si dice robusto se esegue quel che deve eseguire in qualunque circostanza; ciò non è sempre vero, specialmente se esso contiene nella sua definizione dei comandi condizionali e se costituisce o può costituire l'argomento di un altro comando.

È prudente, allora, definire comandi robusti confezionati con cautele particolari, i cui dettagli sono troppo tecnici da spiegare qui. Il tutto viene eseguito mediante il comando `\DeclareRobustCommand` che può venire usato solo nel preambolo o, bene inteso, anche nei file di classe o di estensione, che comunque vengono letti solo nel preambolo. La sua sintassi è identica a quella di `\newcommand`; bisogna stare attenti che esso non verifica se il comando da definire esista già o sia davvero un comando nuovo, quindi questa 'dichiarazione' deve venire usata con molta cautela da programmatori esperti; tuttavia se si fosse definito un comando con `\newcommand` e questo avesse passato i controlli di definibilità, ma nell'uso risultasse fragile, allora non sarebbe un problema

cambiare ilcomando `\newcommand` con `\DeclareRobustCommand` e tutto procederebbe come desiderato con la certezza di non avere pasticciato con comandi già esistenti. In queste circostanze, se si è caricato nel preambolo o nei file di estensione, prima della definizione della macro eseguita con `\newcommand` e la si vuole trasformare in modo che sia robusta, diventa comodo premettere al comando l'operatore `\robustify` che rende robusto il comando ricorrendo ai comandi estesi presenti nel motore di composizione in uso già da almeno una decina d'anni. Nell'esempio del paragrafo 19.3.2 invece di sostituire `\newcommand` con `\DeclareRobustCommand`, si sarebbe potuto premettere al `\newcommand` il comando`\robustify` (sempre che nel preambolo o nel file di macro si fosse caricato il pacchetto *etoolbox*.

Nel capitolo 29 viene descritto il comando `\protect`; esso serve per rendere robusti i comandi fragili; il comando `\DeclareRobustCommand` agisce attraverso `\protect` ma passa anche per l'intermediario di un nuovo comando, apparentemente con lo stesso nome di quello che costituisce il suo argomento, ma prolungato con uno spazio che fa parte integrante del suo nome: se si vuole rendere robusto il comando `\pippo`, `\DeclareRobustCommand` protegge mediante `\protect` il comando `\pippo␣` e a questo assegna la definizione specificata. Questo aiuta a rendere robusti i comandi quando essi vengono scritti nei file ausiliari; quando questi file vengono riletti in una esecuzione successiva, tutto quello che viene letto ripassa attraverso il processo di 'tokenizzazione', e lo spazio letto dal file ridiventa un normale spazio privo di significato quando segue il nome di un comando. Ingegnoso, ma talvolta i neofiti non riescono a rendersi conto di questo passaggio. A questo proposito merita leggere nel capitolo **??** il comando `\ShowCommand` che è in grado di mostrare nella console e nel file `.log` il significato anche di macro protette.

19.7 Definizione di un nuovo ambiente

Per definire un nuovo ambiente si ricorre al comando `\newenvironment` che, prima di agire, verifica che l'ambiente da definire non sia già definito; se lo fosse emetterebbe un messaggio d'errore e la compilazione verrebbe interrotta. Il comando ha la seguente sintassi:

```
\newenvironment{⟨ambiente⟩}[⟨num_arg⟩][⟨default⟩]%
                {⟨comandi di apertura⟩}{⟨comandi di chiusura⟩}
```

dove ⟨*ambiente*⟩ è il nome dell'ambiente che si vuole definire.

L'ambiente che viene definito può ricevere *solo in apertura* un certo numero ⟨*num_arg*⟩ di argomenti il primo dei quali è facoltativo quando viene specificato anche⟨*default*⟩; esso va trattato come il primo argomento facoltativo definito con `\newcommand`. Con la stessa sintassi esiste il comando `\renewenvironment` che serve per ridefinire un ambiente già definito.

Seguono poi i ⟨*comandi di apertura*⟩ da eseguire all'inizio dell'ambiente, il quali fanno uso esplicito degli argomenti eventualmente presenti, obbligatori o

facoltativi; infine vengono definiti i ⟨*comandi di chiusura*⟩ da eseguire appena prima di chiudere l'ambiente.

In realtà definire un ⟨*ambiente*⟩ equivale a definire due comandi uno per l'apertura e l'altro per la chiusura, come se si fossero definiti i due comandi seguenti:

```
\newcommand{\⟨ambiente⟩}[⟨num_arg⟩][⟨default⟩]{⟨comandi di apertura⟩}
\newcommand{\end⟨ambiente⟩}{⟨comandi di chiusura⟩}
```

I comandi `\begin` e `\end` con i quali si aprono e si chiudono gli ambienti eseguono delle funzioni importanti fra le quali la costituzione di un gruppo, cosicché tutto quello che si fa dentro l'ambiente rimane locale e circoscritto solo e soltanto a quell'ambiente; il comando `\end`, oltre a chiudere il gruppo, provvede a verificare che si stia chiudendo l'ultimo ambiente che si era aperto; in caso contrario viene emesso un messaggio d'errore.

19.7.1 Gli ambienti *medaglione* e *sintassi*

Il fatto dei due comandi distinti per l'apertura e la chiusura di un ambiente può essere sfruttato nella definizione di nuovi ambienti; per esempio, in questo documento il file `MacroGuida.sty`, che contiene le definizioni utili alla composizione di questo testo, si definisce l'ambiente *sintassi* che provvede a stampare la sintassi dei vari comandi racchiudendo il suo contenuto in un rettangolo che fa da cornice. In questo modo si è pensato di definire l'ambiente *medaglione*[15] che incornicia il suo contenuto e l'ambiente *sintassi* che espone il medaglione incorniciato ma separato dal testo che lo precede e che lo segue mediante uno spazio verticale uguale a quello che si ha nei testi in display; in particolare si richiede che il testo sia giustificato solo a sinistra; le due definizioni sono le seguenti:

```
\newenvironment{medaglione}[1][\linewidth]{\setbox0\vbox\bgroup
       \hsize=\dimexpr#1-2\fboxsep-2\fboxrule\relax
\noindent}{\par\egroup\setbox0\vbox{\unvbox0}%
       \framebox{\box0}}
%
\newenvironment{sintassi}{\flushleft\medaglione}
                    {\endmedaglione\endflushleft}
```

L'interpretazione delle definizioni per l'ambiente *sintassi* è abbastanza semplice: in apertura esegue le istruzioni per comporre un testo in display giustificato solo a sinistra e poi esegue le operazioni per l'apertura di un medaglione; in chiusura esegue le operazioni per terminare il medaglione e quelle per terminare il testo in display giustificato a sinistra.

[15] Nella pagina 769 c'è un altro esempio simile, l'ambiente *riquadro*; merita osservare le differenze. Qui si descrive una definizione che era stata preparata nelle prime versioni di questo testo e che ricorre a comandi primitivi del linguaggio TEX; è una occasione per mostrare l'uso di tali comandi.

I comandi per la definizione del *medaglione* sono più complicati e richiedono la conoscenza approfondita dei comandi primitivi. Si osservi innanzi tutto che l'ambiente accetta un argomento facoltativo il cui valore in questo esempio vale `0.8\linewidth`. Questo può essere utile per incorniciare e mettere in evidenza anche testi che non abbiano riferimento con la sintassi dei comandi, come si è fatto con questo capoverso.

A parte l'argomento facoltativo, in sostanza i comandi di apertura dicono che con `\setbox` si deve riempire un registro di tipo `box`, quello numerato 'zero' con una scatola verticale `\vbox` la cui apertura è indicata con il comando `\bgroup`; in questa scatola verticale si compone del testo usando una giustezza `\hsize` impostato al valore corrente del primo argomento `#1` esplicitamente specificato o il suo valore di default; ma si diminuisce questo valore di quanto basta per tenere conto dello spazio fra la cornice e il suo contenuto, valore che è conservato dentro il registro di tipo 'lunghezza' chiamato `\fboxsep`, e dello spessore del filetto conservato dentro il registro `\fboxrule`; inoltre si ordina di comporre senza il rientro dei capoversi mediante `\noindent`.

Nei comandi di chiusura bisogna accertarsi che il capoverso costituito dal contenuto del medaglione sia effettivamente terminato ordinandone la terminazione con il comando `\par` e si termina il gruppo che chiude la scatola verticale con `\egroup`; si riassegna poi il contenuto della scatola 'zero' ancora alla scatola 'zero' estraendo le singole righe, ovverosia disfacendo la scatola 'zero' con il comando `\unvbox`; questo serve per essere sicuri che le righe siano convenientemente spaziate; spesso questa operazione è inutile, ma non guasta; infine si usa questa scatola con `\box` come argomento di `\framebox` al fine di incorniciarla.

19.7.2 Definizione di un ambiente per due figure

Talvolta si devono esporre figure di dimensioni relative piccoline, meno larghe di mezza griglia di stampa, o, almeno, la cui somma finale sia inferire alla larghezza della griglia. Verrebbe voglia di metterle fianco a fianco, ciascuna con la sua didascalia. Non si possono usare le funzionalità del pacchetto *subfigure*, perché questo serve per inserire diverse piccole figure *correlate* dentro un unico ambiente flottante, ciascuna dotata di una sua "sotto didascalia" numerata con lettere, e subordinata alla numerazione della didascalia principale. Utilissimo pacchetto, ma serve per una funzionalità diversa. Si vorrebbe realizzare un ambiente che, appunto metta due immagini le le cui larghezze non devono superare la larghezza della gabbia, ciascuna con la sua didascalia composta con la stessa giustezza dell'immagine ridimensionate in modo che abbiano la stessa altezza.

Il codice che segue funziona abbastanza bene:

```
1 \newenvironment{FIGURA}[2]{% apertura
2 % #1 := file della figura di sinistra
3 % #2 := file della figura di destra
```

```
4   \def\TempUno{#1}\def\TempDue{#2}%
5   \dimen6=0.4755\textwidth
6   \setbox0=\hbox{\includegraphics[width=\dimen6]{#1}}
7   \setbox2=\hbox{\includegraphics[width=\dimen6]{#2}}
8   \dimen4=0.5\dimexpr\ht0+\ht2\relax
9   \edef\factuno{\strip@pt\dimexpr\dimen4*\p@/\ht0}
10  \edef\factdue{\strip@pt\dimexpr\dimen4*\p@/\ht2}
11  \dimen0=\factuno\dimen6
12  \dimen2=\factdue\dimen6
13  \begin{figure}
14  }{% chiusura
15  \end{figure}
16 }
```

Esso definisce l'ambiente *FIGURA* il cui comando di apertura richiede due argomenti, come indicato nelle righe di commento: il primo corrisponde al nome del file che contiene la figura da inserire nella parte di sinistra, e il secondo quello della figura di destra. Nella riga 4 i due nomi vengono salvati in due macro temporanee, valide solo dentro l'ambiente stesso. Nelle righe da 5 a 8 si usano quattro registri di dimensione, numerati $0, 2, 4,$ e 6; secondo una convenzione che risale a Knuth stesso, i primi dieci registri possono essere usati come registri temporanei, se hanno numeri pari; non è necessario usare solo i pari, perché tanto all'uscita dell'ambiente essi vengono ristabiliti con lo stesso valore che avevano all'apertura dell'ambiente. Nei registri scatola 0 e 2 si inseriscono con `\includegraphics` le due figure, scalate in modo che abbiano la stesa larghezza pari al valore conservato nel registro 6, in modo da poterne rilevare le altezze al fine di determinare i fattori di scala necessari per portarle alla stessa altezza. Queste altezze si estraggono dalle due scatole mediante il comando nativo di TEX `\ht` (*height*, altezza); nel registro dimensionale 6 si era messo un valore un poco più piccolo della giustezza della gabbia di stampa; invece nel registro dimensionale 4 si mette la somma delle due altezze misurate e la dividiamo a metà.

Ora viene il momento dei calcoli nelle righe 9 e 10. Siccome il sistema TEX usa i valori fratti solo come moltiplicatori delle unità di misura, va bene che si lavori con le grandezze dimensionali, ma dobbiamo ricordare che i valori numerici che si possono estrarre da quei registri sono in punti tipografici, non in millimetri o altre unità. Ecco allora che i due fattori di scala `\factuno` e `\factdue` (numeri fratti adimensionati) si determinano ciascuno prendendo l'altezza media dal registro 4 dividendola per l'altezza naturale , in modo che quando si scalano le due figure ciascuna per il suo fattore di scala esse risultano della stessa altezza, pari alla media delle altezze iniziali.

Ma questi fattori di scala servono per scalare le due figure sia in altezza sia in larghezza; ecco allora che possiamo riutilizzare i registri dimensionali 0 e 2 per conservare la larghezza scalata con cui dimensionare le due figure da affiancare; ciò viene fatto nelle righe 11 e 12.

Ora si hanno tutti gli elementi per usare l'ambiente che con la sintassi seguente:

```
\begin{FIGURA}{⟨Foto1⟩}{⟨Foto2⟩}
  \begin{minipage}[]t]{\dimen0}
    \resizebox{\hsize}{!}{\box0}
    \caption{⟨didascalia1⟩}\label{⟨label1⟩}
  \end{minipage}
  \hfill
  \begin{minipage}Oarg[t]{\dimen2}
    \resizebox{\hsize}{!}{\box2}

cscaption{⟨didascalia2⟩}\label{⟨label2⟩}
  \end{minipage}
\end{FIGURA}
```

Volendo si potrebbe definire a parte un comando che esegua ciascuna delle due linee che contengono `\resizebox`, in modo da rendere più facile ancora l'uso di questo ambiente.

Il lettore attento scopre da solo, prima ancora di cominciare ad usare questo ambiente e questa sintassi, che il calcolo dei fattori di scala produce effetti accettabili se le figure inizialmente non hanno coefficienti di forma *molto* diversi: se una è più larga che alta e l'altra è più alta che larga, i fattori di scala possono portare a larghezze complessive tali che le due figure non possono essere separate da nessuno spazio e formano un blocco che esce dai margini della griglia di stampa. Ma nella maggior parte dei casi il risultato è accettabile. Si vedano le figure 19.1 e 19.2 per apprezzare il risultato.

19.7.3 Commenti sugli ambienti

La persona che si avvicina per la prima volta al contenuto di questo capitolo può restare sconcertata anche da questo linguaggio strano, costituito dai comandi primitivi di TEX che, tra l'altro, non sono mai citati o descritti in questo testo. Be', questo testo è introduttivo, non esaustivo; il lettore si accontenti di questo primo assaggio e poi si documenti convenientemente nel TEXbook se vuole andare oltre questi primi passi.

Nello stesso tempo si capisce bene a che cosa possano servire le macro; per eseguire nell'ordine giusto tutta una serie di operazioni; se si dovessero specificare ogni volta che si deve fare una certa operazione costringerebbero a scrivere e a riscrivere sempre le stesse cose con il pericolo di aggiungere errori ogni volta che si ripetono queste scritture.

19.8 La ridefinizione di ambienti esistenti

La ridefinizione di ambienti già esistenti si esegue con `\renewenvironment` che usa la stessa sintassi di `\newenvironment`. Viene scambiato solo il tipo di test:

Figura 19.1: La nuova cucina

Figura 19.2: Mansarda con un letto a due piazze

se l'ambiente da ridefinire non esiste, viene emesso un messaggio di errore e la compilazione viene interrotta.

Per gli ambienti non esiste un comando simile a `\providecommand` ma, in base a ciò che è stato detto a proposito dei nuovi ambienti, non dovrebbe essere difficile usare due volte il comando `\providecommand` per definire l'apertura e la chiusura di un ambiente; purché tutto ciò sia fatto nel proprio file di macro personali. Se si usa il pacchetto ***xparse***, invece, sono disponibili anche i comandi per definire, ridefinire e provvedere gli ambienti che occorrono; vale la pena di notare che gli ambienti definiti mediante i comandi di ***xparse*** accettano argomenti obbligatori e facoltativi che possono venire usati anche nei domandi di chiusura; questa è un caratteristica importante, perché con i comandi normali di LaTeXsi può usare un solo argomento facoltativo e tutti gli argomenti possono essere usati solo nei comandi di apertura.

Non è nemmeno difficile copiare dai file di classe o di estensione le definizioni di ambienti già esistenti travasando le definizioni nel proprio file di macro personali, per poi separare le due definizioni di apertura e di chiusura in due distinti comandi `\renewcommand` dentro alle cui definizioni apportare tutte le modifiche che si credono opportune.

19.8.1 Ridefinizione dell'ambiente *theindex*

Qui si porterà un esempio abbastanza articolato relativo alla modifica di un ambiente di sistema, la modifica dell'ambiente *theindex* per la composizione

dell'indice analitico.

Si vorrebbe far sì che l'indice sia sempre composto a due colonne, ma facendolo precedere da una spiegazione composta a piena pagina. Vorremmo anche che l'indice analitico figurasse anche nell'indice generale. La prima cosa da fare è quella di esplorare l'archivio CTAN per vedere se il problema è già stato risolto. Qui supporremo di non aver trovato nulla che vada bene per il nostro caso[16] e quindi, rimboccandoci le maniche, ma esercitando la nostra creatività, ci accingiamo a modificare la definizione esistente.

Bisogna quindi copiare la definizione dell'ambiente *theindex* dal file di classe che stiamo usando per comporre il nostro documento nel nostro file di macro personali `mymacros.sty`. Per la classe *book* copieremo pertanto il codice seguente:

```
\newenvironment{theindex}
               {\if@twocolumn
                  \@restonecolfalse
                \else
                  \@restonecoltrue
                \fi
                \twocolumn[\@makeschapterhead{\indexname}]%
                \@mkboth{\MakeUppercase\indexname}%
                        {\MakeUppercase\indexname}%
                \thispagestyle{plain}\parindent\z@
                \parskip\z@ \@plus .3\p@\relax
                \columnseprule \z@
                \columnsep 35\p@
                \let\item\@idxitem}
               {\if@restonecol\onecolumn\else\clearpage\fi}
%
\newcommand\@idxitem{\par\hangindent 40\p@}
\newcommand\subitem{\@idxitem \hspace*{20\p@}}
\newcommand\subsubitem{\@idxitem \hspace*{30\p@}}
\newcommand\indexspace{\par
                 \vskip 10\p@ \@plus5\p@ \@minus3\p@\relax}
```

Le ultime definizioni servono per comporre le singole voci, in particolare per inserire i debiti rientri al fine di mettere in evidenza i tre livelli di voci disponibili. L'ultima definizione, quella di `\indexspace` serve per inserire uno spazio verticale nell'elenco composto ogni volta che si inizia una serie di voci con una nuova lettera dell'alfabeto.

Per poter scrivere su due colonne dopo aver cominciato a scrivere a piena pagina è conveniente aver caricato il pacchetto *multicol*; potremmo anche inserire un test per verificare se il pacchetto è stato davvero già caricato, ma per semplicità ometteremo questo controllo.

[16] In realtà esiste il pacchetto *imakeidx* che fa molto di più di quello che si descrive qui a titolo di esempio.

Allora modifichiamo la definizione dell'ambiente, in particolare la definizione dei comandi di apertura: (*a*) in modo da inserire l'ordine di scrivere nell'indice generale il nome dell'indice analitico e in modo da non usare il comando `\twocolumn` ma l'ambiente *multicols* fornito dal pacchetto *multicol*. Lasciamo stare gli ultimi quattro comandi come sono, per cui qui non li ripetiamo. Ovviamente cambiamo `\newenvironment` in `\renewenvironment` visto che stiamo ridefinendo un ambiente esistente. Inoltre il nuovo ambiente accetta un argomento facoltativo come primo ed unico argomento (quindi da racchiudere fra parentesi quadre) che contiene il testo esplicativo per l'uso dell'indice analitico.

```
\renewenvironment{theindex}[1][]
     {\if@twocolumn
         \@restonecolfalse
       \else
          \@restonecoltrue
        \fi
        \@makeschapterhead{\indexname}%
        \@mkboth{\MakeUppercase\indexname}%
                {\MakeUppercase\indexname}%
        \addcontentsline{toc}{chapter}{\indexname}%
        \thispagestyle{plain}%
        \parindent\z@
        \parskip\z@ \@plus .3\p@\relax
        \columnseprule \z@
        \columnsep 35\p@
        \let\item\@idxitem
        %               Testo di spiegazione
        #1
        \par
        \vspacs{3\baselineskip}%
        \multicols{2}}%                Fine comandi di apertura
       {%                              Inizio comandi di chiusura
        \endmulticols
        \if@restonecol\onecolumn\else\clearpage\fi}
```

Qualche parola di spiegazione non guasta. Il test iniziale eseguito mediante il comando `\if@twocolumn` appare all'inizio di ogni definizione di comandi o di ambienti che comincino un nuovo capitolo numerato o non numerato. Servono per stabilire se dopo la chiusura dell'ambiente sia o non sia necessario ripristinare la composizione a una colonna; l'ultimo test, nei comandi di chiusura eseguito con il comando `\if@restonecol`, serve appunto per ripristinare la composizione ad una colonna nel caso che sia necessario.

Il comando `\@makeschaperhead` è il comando con il quale viene composto il titolo di un capitolo non numerato; in questo caso il titolo del capitolo è contenuto nella variabile `\indexname` che *babel* carica con il nome giusto a seconda della lingua in uso: 'Indice analitico', 'Index', ...

Il comando `\mkboth`, seguito dai suoi due argomenti serve per impostare il contenuto delle due testatine, prima viene indicato il contenuto della testatina di sinistra, poi di quella di destra. Il contenuto è sempre il nome in lingua corrente dell'indice analitico, ma reso in lettere maiuscole attraverso il comando `\MakeUppercase`.

Il comando `\addcontentsline`, come dice il nome, serve per aggiungere una riga alla 'table of contents', cioè all'indice generale. I suoi tre argomenti indicano rispettivamente la sigla della 'table of contents', `toc`, il tipo di voce da aggiungere a questa lista; in questo caso si tratta di una voce corrispondente al titolo di un capitolo; segue poi il titolo del capitolo che in questo caso continua ad essere il nome dell'indice analitico nella lingua corrente.

Il comando `\thispagestyle` specifica lo stile compositivo di questa prima pagina di questo nuovo capitolo; in questo caso si tratta dello stile `plain` che non contiene la testatina, ma contiene il numero della pagina nel piedino. Gli altri stili definiti di default sono `empty`, senza testatina né piedino, `headings` con piedino vuoto e testatina il cui contenuto viene specificato automaticamente dai comandi `\chapter` oppure `\section`; in particolare nella testatina di sinistra compare il titolo del capitolo e in quella di destra il titolo del paragrafo corrente all'inizio della pagina. Infine `myheadings` è simile a `headings` solo che il contenuto delle due testatine deve essere fissato manualmente dal compositore specificando di volta in volta entrambe le testatine oppure solo quella di destra attraverso i comandi `\markboth` o `\markright`.

La scrittura `\parindent \z@` serve per specificare che il rientro del capoverso deve essere di zero punti; questo valore è memorizzato nella variabile di sistema `\z@` che equivale alla scrittura 0 pt.

La successiva espressione `\parskip\z@ \@plus .3\p@\relax` serve per specificare un pochino di gomma elastica da inserire fra un capoverso e l'altro; in pratica fra una voce e l'altra; questo minimo di allungamento consente di giustificare verticalmente le colonne della composizione a due colonne.

La specificazione successiva `\columnseprule \z@` specifica che lo spessore del filetto verticale che separa le colonne è di zero punti (quindi il filetto non c'è). Invece `\columnsep 35\p@` dice che la separazione fra le due colonne vale 35 pt, quindi una decina di millimetri.

Al comando `\item`, che abitualmente viene usato nelle liste, qui viene assegnata un'altra definizione, quella che è stata attribuita al comando `\@idxitem` e che non è stata modificata nell'eseguire questa nuova definizione dell'ambiente *theindex*.

Segue poi il testo di spiegazione, che qui è stato indicato col numero dell'unico argomento facoltativo passato al comando di apertura dell'ambiente; volendo si potrebbe fare un test per verificare se l'argomento `#1` (che di default non contiene nulla) contenga un po' di testo; se si carica il pacchetto *etoolbox* questo test si potrebbe fare semplicemente sostituendo le righe

```
#1
\par
```

con la riga:

```
\ifcsvoid{#1}{}{#1\par}
```

Finita la spiegazione il comando `\par` assicura che il capoverso contenente le spiegazioni sia veramente terminato; il successivo comando di spaziatura verticale dato con `\vspace` inserisce una spaziatura di tre righe di testo; infatti `\baselineskip` indica appunto lo scartamento delle righe del testo corrente.

I comandi di apertura finiscono con l'apertura dell'ambiente *multicols* al quale viene specificato di comporre su due colonne; non è stata usata la sintassi normale per aprire l'ambiente, perché siamo sicuri di chiuderlo nei comandi di chiusura, come in effetti facciamo eseguendo il comando `\endmulticols`.

Questo esempio è istruttivo sotto molti aspetti; dà una buona idea di ciò che viene eseguito dietro le quinte quando si dà un comando qualunque; dice quanto sia comodo che le macro possano fare tutto quello che fanno, così che con un semplice comando di mark-up si possano sistematicamente e uniformemente eseguire le stesse operazioni compositive; quanto lavoro si risparmi nell'usare macro ogni volta che sia possibile, e quindi quanto sia opportuno che, in base al testo che si sta componendo, il file `mymacros.sty` contenga proprio le definizioni delle macro personali che servono per comporre le strutture lessicali che più frequentemente compaiono nel testo.

19.8.2 Ridefinizione dell'ambiente *thebibliography*

L'ambiente *thebibliography* serve per comporre una bibliografia a mano, come descritto nel capitolo 11. Questo ambiente non inserisce il "capitolo" della bibliografia nell'indice generale. Usando i pacchetti che estendono la gestione della bibliografia all'uso dei database bibliografici, viene generato un comando del tipo `\printbibliography`, che accetta alcune opzioni fra le quali si trova anche l'opzione *intoc* (o simile, a seconda del pacchetto usato) che permette di aggiungere all'indice anche la voce per la bibliografia.

Ma qui ci si riferisce all'ambiente di default della bibliografia, in particolare quello della classe *book*. Come si è fatto per l'indice analitico, si copia dal file *book.cls* la definizione dell'ambiente *thebibliography* e la si incolla nel file di macro personali. Il codice originale è il seguente:

```
\newenvironment{thebibliography}[1]
     {\chapter*{\bibname}%
      \@mkboth{\MakeUppercase\bibname}{\MakeUppercase\bibname}%
      \list{\@biblabel{\@arabic\c@enumiv}}%
           {\settowidth\labelwidth{\@biblabel{#1}}%
            \leftmargin\labelwidth
            \advance\leftmargin\labelsep
            \@openbib@code
            \usecounter{enumiv}%
            \let\p@enumiv\@empty
```

```
          \renewcommand\theenumiv{\@arabic\c@enumiv}}%
      \sloppy
      \clubpenalty4000
      \@clubpenalty \clubpenalty
      \widowpenalty4000%
      \sfcode`\.\@m}
     {\def\@noitemerr
       {\@latex@warning{Empty `thebibliography' environment}}%
      \endlist}
\newcommand\newblock{\hskip .11em\@plus.33em\@minus.07em}
\let\@openbib@code\@empty
```

Il lettore ha qui un ottimo esempio per studiare i comandi interni dell'ambiente. Noti che le ultime due righe (che non verranno modificate) sono comandi utili per dare attuazione all'opzione *openbib* della classe *book*.

Per ottenere l'inserimento della voce in lingua contenuta nella macro `\bibname` impostata da *babel*, si deve ridefinire l'ambiente aggiungendo `\addcontentsline`; lavorando, quindi, sul codice copiato nel file delle macro personali, si deve cambiare `\newenvironment` con `\renewenvironment` e si deve aggiungere la nuova istruzione completa dei suoi tre argomenti obbligatori. Si ottiene, perciò, il codice seguente:

```
\renewenvironment{thebibliography}[1]
     {\chapter*{\bibname}%
      \@mkboth{\MakeUppercase\bibname}{\MakeUppercase\bibname}%
      \addcontentsline{toc}{chapter}{\bibname}% <--- aggiunta
      \list{\@biblabel{\@arabic\c@enumiv}}%
           {\settowidth\labelwidth{\@biblabel{#1}}%
            \leftmargin\labelwidth
            \advance\leftmargin\labelsep
            \@openbib@code
            \usecounter{enumiv}%
            \let\p@enumiv\@empty
            \renewcommand\theenumiv{\@arabic\c@enumiv}}%
      \sloppy
      \clubpenalty4000
      \@clubpenalty \clubpenalty
      \widowpenalty4000%
      \sfcode`\.\@m}
     {\def\@noitemerr
       {\@latex@warning{Empty `thebibliography' environment}}%
      \endlist}
```

Vale la pena osservare la presenza alla fine dei comandi di apertura dell'istruzione `\sfcode`\.\@m`; il comando `\sfcode` serve per impostare il valore dello *space factor* del punto fermo indicato con la forma che si riferisce al suo codice

interno: "`\."; gli viene assegnato il valore 1000, contenuto nella sequenza `\@m`. Il valore 1000 esprime in millesimi il valore del fattore che si usa per allargare lo spazio interparola dopo il carattere indicato dal primo argomento di `\sfcode`. È uno dei valori impostati dal comando `\frenchspacing`, di cui si parla nel paragrafo 29.2.1, nel quale si è osservato che nella bibliografia, dove le abbreviazioni sono numerose lo spazio dopo in segni di interpunzione di fine periodo sono resi uguali. Precisamente l'ambiente per la bibliografia cambia *solo* lo spazio dopo il punto fermo, ma non lo cambia dopo i punti esclamativo e interrogativo, che verosimilmente non compaiono mai nelle bibliografie.

19.9 Situazioni particolari

19.9.1 Le linee guida

Le *linee guida* sono quelle linee continue oppure linee di puntini o di altri simboli che guidano l'occhio da una parte all'altra della pagina come in questa riga:

Inizio.....Fine

In inglese le linee guida si chiamano *leaders*.

LaTeX contiene già le definizioni di alcune linee guida: `\hrulefill` e `\dotfill`; il primo comando produce una linea continua appoggiata alla base della riga di stampa_____come in questa riga formata da una linea spessa 0,4 pt; agendo a livello di comandi primitivi è possibile ridefinire il comando o definirne un altro con una riga più spessa e/o non adagiata sulla linea di base; l'altro comando produce una fila di puntini..... come in questa riga.

È possibile definire o ridefinire i comandi che generano le linee guida, ma ovviamente è sconsigliabile ridefinire i comandi del nucleo di LaTeX indicati sopra, perché LaTeX li usa dentro altri comandi e la loro modifica ne potrebbe alterare in modo irreparabile il loro funzionamento; perciò qui faremmo esempi che usano `\newcommand` per essere sicuri di non ridefinire nulla di preesistente.

Ogni linea guida è creata con una serie di comandi secondo la sintassi seguente:

```
\leaders⟨scatola⟩\hskip⟨gomma⟩
\cleaders⟨scatola⟩\hskip⟨gomma⟩
\xleaders⟨scatola⟩\hskip⟨gomma⟩
```

La comprensione dei termini usati in queste definizioni è facilitata se si rivedono questi concetti nel capitolo 29.

Brevemente, in questo contesto per ⟨*scatola*⟩ si intende sia una scatola vera e propria da riempire con quello che si vuole, sia un rettangolo nero (una scatola riempita di nero); il comando `\hskip` serve per produrre uno spostamento orizzontale; perché la linea guida sia arbitrariamente estensibile lo spostamento deve essere *elastico*, ed è per questo che viene chiamato ⟨*gomma*⟩; questa ⟨*gomma*⟩ è una speciale lunghezza che è formata da tre termini:

1. la lunghezza naturale;
2. l'allungamento il cui ammontare è preceduto dalla parola chiave `plus`, sostituita nel nucleo di LaTeX, dalla macro `\@plus`;
3. l'accorciamento il cui ammontare è preceduto dalla parola chiave `minus`, sostituita nel nucleo di LaTeX, dalla macro `\@minus`;

Tutte le lunghezze definite con i comandi di LaTeX sono in effetti delle *gomme*. Per esempio la lunghezza definita come segue:

```
\newlength{\lamialunghezza}
\setlength{\lamialunghezza}{10pt plus 5pt minus 3pt}
```

definisce una lunghezza elastica (gomma) con il nome `\lamialunghezza`, la cui lunghezza naturale è di 10 pt, il cui allungamento vale 5 pt e il cui accorciamento vale 3 pt: insomma, una lunghezza che si può estendere da 7 pt a 15 pt a seconda delle necessità.

Tanto per focalizzare meglio la natura della gomma, lo spazio interparola è una lunghezza elastica che ha una sua larghezza naturale pari a quella della lettera 'e' nel font corrente e un allungamento e un accorciamento entrambi più piccoli della larghezza naturale; TeX tiene conto di quanto lo spazio interparola presente fra le parole di una riga si debba allargare o restringere per la sua giustificazione al fine di determinare la "bellezza" della riga, o meglio la sua "bruttezza"; in base ad elaborati calcoli eseguiti sui valori di allungamento e accorciamento effettivamente usati, TeX determina un livello di bruttezza da confrontare con il valore della `\tolerance` o della `\pretolerance` al fine di accettare o non accettare quella particolare riga, oppure per decidere se eseguire la cesura in fin di riga, oppure se emettere un messaggio che avvisa l'operatore che in quel capoverso c'è una riga troppo vuota (underfull hbox) o sporgente fuori dal margine (overfull hbox). Si tratta di concetti molto tecnici, ma è meglio averne una conoscenza almeno superficiale.

Tornando alle nostre linee guida, la ⟨*gomma*⟩ da usare deve avere una componente di allungamento infinita se vogliamo che i nostri leader si possano allungare a piacere. Infatti le definizioni delle due linee guida predefinite sono:

```
\def\hrulefill{\leavevmode\leaders\hrule\hfill\kern\z@}

\def\dotfill{\leavevmode
             \cleaders \hb@xt@ .44em{\hss.\hss}\hfill\kern\z@}
```

Il comando `\leavevmode` assicura che il comando venga usato in modo orizzontale. Il comando `\kern\z@` assicura che dopo la linea guida si prosegua senza ulteriori spazi e senza andare a capo; il comando `\hfill` indica di inserire un spazio dato da una gomma con una lunghezza naturale nulla e un allungamento infinito.

Nella definizione di `\hrulefill` compare la scatola nera, il rettangolo nero, `\hrule`, con una larghezza naturale che non influisce sulla lunghezza della linea

guida, con altezza pari a 0,4 pt e profondità nulla; il suo effetto è quindi di produrre una linea guida lunga quanto occorre, ma spessa solo 0,4 pt appoggiata alla linea di base.

Nella definizione di `\dotfill` compare il codice `\hb@xt@ .44em` il cui argomento vale `{\hss.\hss}`. Il primo comando è scritto in modo sintetico ed equivale al comando LaTeX: `\makebox`. L'argomento di questa scatola è un punto fermo e da due gomme `\hss` di spazio orizzontale di lunghezza naturale nulla ma di allungamento e accorciamento infiniti. Il tutto è equivalente al comando LaTeX `\makebox[0.44em]{.}` e il punto è centrato dentro la scatola di 0,44 pt.

Tuttavia sopra si sono indicati tre tipi di linee guida, uno introdotto da `\leaders`, il secondo da `\cleaders`, il terzo da `\xleaders`; che differenza c'è fra questi tre tipi di linee guida?

1. `\leaders` introduce i suoi puntini definiti mediante una scatola vera, non con un rettangolo nero, come se fossero su una griglia fissa, per cui da una riga all'altra, come succede per esempio in un indice generale, i puntini sono sempre incolonnati, anche se questo comporta un eventuale spazio bianco all'inizio e alla fine, ma che l'incolonnamento dei puntini impedisce di riempire.

2. `\cleaders` introduce i suoi puntini centrando la linea di puntini senza preoccuparsi che siano incolonnati; va bene in una linea guida isolata di testo ma non in un indice generale dove le linee guida, se ci sono, è meglio che abbiano i puntini incolonnati.

3. analogamente `\xleaders` introduce i suoi puntini allargando un poco lo spazio fra di loro in modo da estendere la linea guida così che non solo il primo e l'ultimo spazio siano uguali, ma anche lo spazio restante sia distribuito fra i puntini presenti.

Si considerino le seguenti tre definizioni di tre diverse linee guida:

```
\newcommand*\puntini{\leaders\hbox to 10pt{\hss.\hss}\hfill\kern0pt}
\newcommand*\cpuntini{\cleaders\hbox to 10pt{\hss.\hss}\hfill\kern0pt}
\newcommand*\xpuntini{\xleaders\hbox to 10pt{\hss.\hss}\hfill\kern0pt}
```

e si inseriscano le linee guida così definite in una scatola larga 99 pt inserita fra due parole. Siccome ogni scatola-puntino delle linee guida è larga 10 pt, nello spazio di 99 pt ce ne stanno 9; i restanti 9 pt vengono distribuiti diversamente fra l'inizio e la fine a seconda del tipo di linea guida usata:

`\leaders`: Prima . . . . . Dopo
`\cleaders`: Prima . . . . . Dopo
`\xleaders`: Prima . . . . . Dopo

Si noti anche che nelle definizioni del nucleo di LaTeX la grandezza della scatola-puntino è definita in termini di 'em', cioè di una unità di misura che dipende dal font usato; quindi anche se `\leaders` dovrebbe essere in grado di

assicurare l'incolonnamento dei puntini, questo non può succedere se i capitoli sono composti in neretto e i paragrafi in serie media. Questo è il motivo per il quale nell'indice generale di questo testo (e nelle classi standard) le righe che riguardano i capitoli sono senza linee guida. Se le avessero, succederebbe quello che appare nel seguente esempio:

1 Primo capitolo 1
1.1 Primo paragrafo 1
1.2 Secondo paragrafo 2

Si vede chiaramente la mancanza dell'incolonnamento fra i puntini della riga del capitolo e quella dei paragrafi; si nota anche la presenza di spazi diversi fra le linee guida e il contenuto che loro "uniscono" visivamente. Queste disuniformità, comunque, diminuiscono via via che le scatole-puntino vengono accorciate e i puntini effettivamente inseriti nelle linee guida sono più densi.

Va anche da sé che al posto dei puntini si potrebbero indicare degli altri simboli testuali, matematici o decorativi.

19.9.2 Controllo della posizione di grandi oggetti

In questo testo lo schema del layout delle pagine è mostrato sia a partire dal verso dell'occhiello, sia nelle pagine 166 e seguente, sia nelle pagine 528 e seguente.

Nello stesso modo si potrebbe spezzare una lunga tabella di due pagine affinché cominci su una pagina pari o una larga tabella da mettere su due pagine affacciate con metà delle colonne nella pagina pari e l'altra metà nella pagina dispari. Gli esempi potrebbero andare avanti a lungo.

Si potrebbe pensare di usare il comando `\afterpage` fornito dal pacchetto *`afterpage`*. Ma questo comando esegue quanto indicato nel suo argomento non appena è finita la pagina corrente, indipendentemente dal numero pari o dispari della pagina appena terminata. Bisognerebbe procedere a tentoni e correggerne la posizione ad ogni modifica del testo del documento.

Abbiamo chiesto all'autore del pacchetto *`afterpage`* se non avesse voluto rimettere le mani sul suo codice per aggiungere altri comandi che verificassero in automatico la parità del numero della pagina. Noi non volevamo mettere le mani sul suo codice, tanto più che il suo codice agiva sulla routine di uscita, cosa che solo un *TEXwizard* può essere capace di fare. David Carlisle ci ha risposto gentilmente che non se la sentiva di rimettere le mani nel suo codice scritto quindici anni prima, ma ci incoraggiava ad eseguire un comando ricorsivo di cui ci forniva il modello, dicendoci che non l'aveva controllato e amichevolmente invitandoci a tentare la sorte. Ma i veri *TEXwizard* come lui la sanno molto lunga; lo ringraziamo moltissimo per averci permesso di realizzare delle macro ricorsive a partire dalle sue indicazioni (che funzionavano benissimo, naturalmente).

Nel file `MacroGuida.sty`, che contiene le definizioni delle macro usate per comporre questo testo, appaiono le righe di codice che seguono, dove vengono

definiti i due comandi `\Supaginapari` e `\Supaginadispari`, il secondo dei quali meno utile del primo, ma certamente non inutile; qui si illustra solo il primo.

```
\@ifpackageloaded{afterpage}{\relax}{\RequirePackage{afterpage}}

\makeatletter % necessario per inserire il codice nel preambolo
\newcommand\Supaginapari[1]{%
  \afterpage{%
    \ifodd\value{page}\expandafter\@firstoftwo
    \else\expandafter\@secondoftwo
    \fi{\Supaginapari{#1}}{#1}
  }
}
```

Poiché i due comandi si affidano all'esistenza della macro `\afterpage` definita dal pacchetto *afterpage*, per prima cosa si verifica che il pacchetto sia già stato caricato; se non lo fosse si provvede in merito. Se la chiamata fosse in seguito ripetuta, non sarebbe un problema, perché i comandi `\RequirePackage` e `\usepackage` verificano sempre da soli se il o i pacchetti indicati nel loro argomento siano o non siano già stati caricati, affinché non vengano caricati nuovamente con il pericolo di annullare eventuali ridefinizioni di comandi eseguite dal programmatore, o da altri pacchetti caricati fra la prima e la seconda chiamata. Il comando `\makeatletter` è necessario solo se si inserisse questo codice nel preambolo del proprio documento, mentre non è necessario se lo si inserisce in file di macro personali da richiamare nel preambolo con `\usepackage`. Nel nostro caso non è necessario perché il codice si trova dentro al file `MacroGuida.sty`.

La struttura dei due comandi è simile: per il primo comando, quando l'argomento della macro`\afterpage`, viene eseguito, è appena stata emessa una pagina e il numero di pagina è stato appena incrementato; se era stata appena emessa una pagina pari, il nuovo numero di pagina è dispari o viceversa. Quindi il test eseguito con il comando primitivo `\ifodd` verifica la disparità del contenuto del contatore `page` attraverso l'uso della macro `\value`; se questo contenuto ha la parità giusta, emette direttamente l'argomento ricevuto, altrimenti rimette in coda l'intero comando col suo argomento passandolo di nuovo al comando `\afterpage`, che lo dilaziona alla fine della pagina successiva.

Ecco quindi che i layout delle pagine affacciate di cui si parlava all'inizio di questo paragrafo è stato specificato con

```
\Supaginapari{\layout*}
```

e il disegno delle due pagine affacciate appare sempre a cominciare da una pagina pari, la prima disponibile dopo aver dato quel comando.

19.9.3 Immagini, tabelle e scatole

Il titolo di questo paragrafo è un po' enigmatico; in realtà il problema è semplice: inserire immagini nelle celle di una tabella può risultare difficile, a meno che non

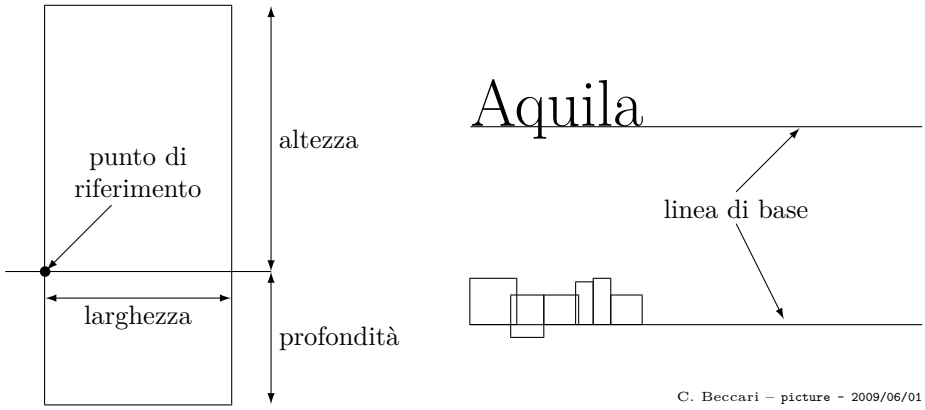


Figura 19.3: Alcune proprietà delle scatole. Nella figura a destra si vedono le scatole corrispondenti ai caratteri che formano la parola 'Aquila' allineate sulla linea di base. Le crenature fra i caratteri sono visibili dove le scatole si sovrappongono.

si sia disposti a manipolare le scatole.

Nel resto di questo testo non si è detto molto sulle scatole; l'arte di eseguire la composizione tipografica con LaTeX dovrebbe astenersi dal manipolare le scatole interne del programma di composizione, almeno in prima battuta.

Ciò non toglie che qualche nozione possa essere utile, anche per capire meglio alcuni strani costrutti eseguiti con le scatole nella definizione di alcuni comandi.

I programmi di composizione del sistema TeX maneggiano quasi esclusivamente scatole; sono scatole i singoli caratteri, le righe del testo, le pagine composte, tanto per fare esempi molto semplici. Ogni scatola è definita dal suo tipo e da tre dimensioni particolari: la larghezza, l'altezza e la profondità. Il rettangolo che può simboleggiare una scatola ha un punto di riferimento all'intersezione del bordo sinistro con la linea che separa la sua parte alta dalla sua parte profonda; il segmento intercettato su questa linea dal bordo destro e dal bordo sinistro è lungo quanto la larghezza della scatola; l'altezza è la lunghezza del segmento che parte dal punto di riferimento e raggiunge il vertice in alto a sinistra del rettangolo, mentre la profondità è la lunghezza del segmento che parte dal punto di riferimento e raggiunge il vertice in basso a sinistra; figura 19.3.

Le scatole dei caratteri vengono allineate a formare una riga semplicemente allineando i loro punti di riferimento sulla linea di base della riga composta, come si vede nella figura 19.3; il programma prende le dimensioni delle scatole che racchiudono i caratteri dal file metrico del font che sta usando; questo contiene non solo tutte le informazioni dimensionali di ciascun carattere, ma anche le dimensioni delle crenature[17] e le possibili legature fra caratteri. La riga

[17]La crenatura (*kerning* in inglese) è l'ammontare dell'accostamento di due caratteri adiacenti, in modo da non fare apparire troppo spazio fra caratteri consecutivi dalle forme complementari;

così formata rappresenta una scatola orizzontale, la cui larghezza è la somma delle larghezze dei caratteri componenti e degli spazi vuoti che separano le parole; l'altezza della riga è la maggiore delle altezze delle scatole allineate orizzontalmente, mentre la profondità della riga è la maggiore delle profondità delle scatole suddette; il punto di riferimento è perciò l'intersezione della linea di base con il lato sinistro del rettangolo che racchiude la riga composta.

Le scatole formate dalle righe composte vengono incolonnate in verticale allineando lungo una retta verticale i punti di riferimento di tutte le scatole orizzontali usate e intercalando alle scatole orizzontali gli spazi bianchi di interlinea e i contrografismi verticali prima e dopo ogni scatola che contenga del testo in evidenza. La pagina composta è quindi una scatola verticale; ma il programma può gestire anche altre scatole, differenti dall'intera pagina, scatole che a loro volta possono contenere altre scatole, e così via come una serie di matriosche.

LaTeX ha due tipi di scatole *orizzontali* che l'utente può usare con comandi accessibili direttamente dal compositore: `\mbox` e `\makebox`; alcune varianti sono le scatole con cornice, come `\fbox` e `\framebox`; il compositore può definire per nome dei registri-scatola nei quali può salvare scatole orizzontali mediante i comandi `\newsavebox`, `\sbox` e `\savebox`; il loro contenuto viene richiamato e usato con `\usebox`.

LaTeX ha due tipi di scatole *verticali* a cui il compositore può accedere liberamente; una è la scatola verticale prodotta con il comando `\parbox` e l'altra è la 'minipagina' ottenibile con l'ambiente *minipage*. Le scatole verticali di LaTeX possono ricevere il loro contenuto come sequenze di caratteri e di altri oggetti tipicamente di tipo orizzontale, oppure come scatole orizzontali già composte; nel primo caso LaTeX provvede a costruire le scatole orizzontali da impilare in verticale. La larghezza di simili scatole verticali coincide con quella della scatola orizzontale più larga, l'altezza e la profondità dipendono dalle opzioni che vengono specificate al comando o all'ambiente, dipendendo da come queste opzioni specificano la posizione del punto di riferimento. La scatola di tre righe Pippo Pluto Paperino può avere il suo punto di riferimento collocato in modo che la prima

Pippo
Pluto
Paperino

riga sia allineata con il testo circostante; oppure

Pippo
Pluto
Paperino

può avere l'ultima riga allineata con il testo circostante; oppure

Pippo
Pluto
Paperino

può avere il suo asse matematico allineato con l'asse matematico del testo circostante. Come si vede da questo capoverso le scatole verticali non sono molto utili da usare all'interno del testo; esse trovano applicazioni particolari in altri tipi di composizione.

I comandi primitivi di TeX sono meno elaborati; i registri-scatola possono essere chiamati per nome, come in LaTeX, oppure per numero, purché questo

per esempio 'VA' sono accostate, mentre 'VA' non lo sono; la differenza è evidente e un font nel quale manchi l'accostamento sarebbe di qualità assai modesta.

sia strettamente compreso nell'intervallo da 0 a 254[18]; il registro-scatola 255 è riservato e l'utente non lo deve mai toccare; le scatole orizzontali sono contenute dentro `\hbox`; le scatole verticali si chiamano `\vbox`, `\vtop` e `\vcenter`; i comandi per utilizzare il contenuto delle varie scatole registrate nei registri-scatola sono invece più elaborati di quelli di LaTeX. Il lettore è invitato ad approfondire l'argomento nel TeXbook, che dedica a questi argomenti diversi capitoli.

Dopo queste premesse, possiamo tornare alle nostre immagini e alle celle delle tabelle; abbiamo visto che uno dei descrittori delle colonne è `p{`⟨*larghezza*⟩`}`; questo in sostanza definisce una scatola verticale di larghezza pari a ⟨*larghezza*⟩ dentro la quale viene composto un breve capoverso; ma questa scatola ha la *prima* riga allineata con la prima o unica riga delle celle adiacenti. Mediante l'uso del pacchetto *array* abbiamo visto che si possono usare altri due descrittori, `b{`⟨*larghezza*⟩`}` e `m{`⟨*larghezza*⟩`}` che si comportano rispettivamente come scatole verticali con l'ultima riga allineata ovvero con l'asse matematico allineato.

Le immagini introdotte nel documento ricorrendo al pacchetto *graphicx* mediante il suo comando `\includegraphics`, sono delle scatole orizzontali col loro punto di riferimento coincidente con il loro vertice inferiore sinistro. Si capisce allora che per collocare una immagine bisogna tenere conto di altezza e profondità delle scatole usate e regolarsi in modo da ottenere un buon risultato. La tabella 19.1 che si ottiene senza pensare a queste cose è orrenda. La tabella 19.2, composta ricordando le proprietà delle scatole appena descritte, invece, è composta abbastanza correttamente, anche se, come vedremo, si può fare di meglio.

La tabella 19.2 può ancora essere ritoccata aggiungendo un piccolo spazio sopra l'immagine, in modo che il filetto orizzontale non la sfiori: tabella 19.3. Questa operazione viene eseguita automaticamente se si usano il filetti orizzontali del pacchetto *bookktabs*

Infatti la tabella 19.3 è stata composta con i comandi seguenti:

```
\begin{tabular}{m{120pt}m{35mm}}
   \toprule
   Immagine & Descrizione\\
   \midrule
   \includegraphics[width=120pt]{Formaggio} &
   Il formaggio è un prodotto derivato dal latte, attraverso un
   processo che coagula la caseina, una proteina del latte.\\
   \bottomrule
\end{tabular}
```

Premesso che il descrittore di colonna `m{`⟨*larghezza*⟩`}` del pacchetto *array* risolve il problema, che cosa bisogna fare se per qualche motivo non si potesse

[18] In realtà i programmi di composizione pdftex, xetex e luatex incorporano le estensioni del programma etex, come si è già spiegato nei capitoli iniziali, e quindi non è limitato nel numero dei suoi registri al numero di 255, ma ne può avere 2^{15}, numerati da 0 a $2^{15} - 1$; insomma ne può avere in pratica un numero illimitato, visto che è difficile immaginare la composizione di un documento qualsiasi che impegni simultaneamente un numero così elevato di registri-scatola.

Immagine	Descrizione
	Il formaggio è un prodotto derivato dal latte, attraverso un processo che coagula la caseina, una proteina del latte.

Tabella 19.1: Tabella in cui l'immagine e il testo adiacente hanno i punti di riferimento allineati in modo scorretto

Immagine	Descrizione
	Il formaggio è un prodotto derivato dal latte, attraverso un processo che coagula la caseina, una proteina del latte.

Tabella 19.2: Tabella in cui l'immagine e il testo adiacente hanno i punti di riferimento allineati correttamente

Immagine	Descrizione
	Il formaggio è un prodotto derivato dal latte, attraverso un processo che coagula la caseina, una proteina del latte.

Tabella 19.3: Tabella in cui l'immagine e il testo adiacente hanno i punti di riferimento allineati correttamente e dove uno spazio verticale pari a 0,8 ex separa l'immagine dal filetto sovrastante

conservare quello stesso tipo di descrizione per tutte le celle della stessa colonna? Bisogna ricorrere a comandi di basso livello, cioè ai comandi che manipolano direttamente le scatole in modo da risolvere il problema compositivo solo per la cella specifica che contiene l'immagine. Conviene anzi definirsi un comando che esegua direttamente l'inclusione dell'immagine, ma con la posizione del punto di riferimento specificata mediante una opzione. Si potrebbe definire un nuovo comando `\tabularimage` con la sintassi:

`\tabularimage[`⟨*posizione*⟩`]{`⟨*larghezza*⟩`}{`⟨*file immagine*⟩`}`

nel modo seguente:

```
\newcommand*\ttoprule{\hrule \@height0pt \@width0pt \@depth-1ex}
\newcommand*\ctoprule{\hrule \@height0pt \@width0pt \@depth.8ex}
\let\btoprule\ctoprule \let\bbotrule\ctoprule
\let\cbotrule\bbotrule \let\tbotrule\bbotrule
\newcommand*{\tabularimage}[3][b]{\parbox[#1]{#2}{%
   \csname#1toprule\endcsname %
   \includegraphics[width=#2]{#3}\par
   \csname#1botrule\endcsname}}
```

dove la ⟨*posizione*⟩ di default è la stessa che si avrebbe con il semplice comando `\includegraphics`. Con questo nuovo comando si potrebbe comporre la tabella 19.4 con i comandi:

```
\begin{tabular}{lp{35mm}}
   \hline
   Immagine & Descrizione\\
   \hline
   \tabularimage[t]{120pt}{Formaggio} &\kern-6pt
   Il formaggio è un prodotto derivato dal latte, attraverso un
   processo che coagula la caseina, una proteina del latte.\\
   \hline
\end{tabular}
```

La definizione di `\tabularimage` è un po' particolare, perché i filetti (invisibili) inseriti prima e dopo l'immagine all'interno della scatola verticale `\parbox` non possono essere i comuni filetti di LaTeX specificati con `\rule`; infatti qui abbiamo bisogno di incolonnare solo scatole verticali o altri oggetti che si possono mettere in una lista verticale; il filetto `\hrule` nonostante l'iniziale `h`, è un comando primitivo che specifica un filetto le cui altezza, larghezza e profondità sono specificate con i parametri che seguono quel comando, ma è un oggetto che può stare solo nelle liste di oggetti incolonnati in verticale. I punti di riferimento di questi filetti diventano quelli della scatola verticale quando le si specifichi l'opzione di ⟨*posizione*⟩ `t` oppure `b`; nel caso dell'opzione `c`, non conta tanto il punto di riferimento, quanto l'asse matematico, che TeX determina come linea mediana della scatola. Si capisce, allora, che a seconda che il collocamento della

Immagine	Descrizione
	Il formaggio è un prodotto derivato dal latte, attraverso un processo che coagula la caseina, una proteina del latte.

Tabella 19.4: Tabella con l'immagine allineata con la parte superiore delle minuscole della prima riga della seconda colonna

scatola corrisponda ad una delle tre specificabili, i filetti (invisibili) superiore e inferiore debbano avere caratteristiche diverse; si sono definiti sei filetti invisibili, che in realtà sono solo due distinti ma hanno sei nomi che si equivalgono; questi nomi sono ottenuti agglutinando la lettera di opzione per la collocazione con i nomi `toprule` e `botrule`; questa agglutinazione viene eseguita dalla definizione di `\tabularimage` solo al momento in cui il parametro `#1`, che corrisponde a questa opzione, è noto e l'operazione viene eseguita dalla coppia di comandi `\csname` e `\endcsname` che creano un nome di comando valido per il programma a partire da qualsiasi stringa essi racchiudano.

Questo genere di incapsulamenti di scatole con diversi punti di riferimento può essere fatto anche con le minipagine create con l'ambiente *minipage* o con le scatole orizzontali con diversi allineamenti del loro contenuto e diverse larghezze nominali.

Non si insiste oltre essenzialmente per non intimorire l'utente alle prime armi. È certo che la definizione di questi comandi, specialmente ricorrendo ai trucchi esposti, rientra nella programmazione col linguaggio LaTeX, certamente non nell'arte di comporre tipograficamente con LaTeX, anche se ne costituisce il supporto. Ma l'appetito vien mangiando; prima o poi tutti sentiamo il bisogno di definirci i nostri comandi, i nostri ambienti, le nostre scorciatoie per comporre meglio e più correttamente.

Conclusioni

Gli ultimi esempi sono piuttosto ricchi di spunti, ma non esauriscono certamente la questione della creazione delle proprie macro. Questa è un'arte difficile, ma non impossibile; bisogna avere la pazienza e l'umiltà di imparare senza fretta, sapendo però che la ricompensa sarà forte sotto l'aspetto della facilità del comporre.

Bisogna avere la pazienza di imparare dai libri e dai manuali esistenti, alcuni dei quali elencati nel capitolo 21. Bisogna avere anche la pazienza di leggere il codice dei file scritti da altri, in primis quelli che definiscono il formato e le classi e i pacchetti standard. Alcune parti sono veramente difficili da capire senza l'aiuto

di una documentazione, ma questa è generalmente disponibile nei file `.dtx`; questi sono file TeX documentati. Se si lancia pdfLaTeX su uno qualunque di questi file, si ottiene un documento nel quale il codice è inframmezzato a spiegazioni; se queste siano utili dipende dalla competenza informatico/programmatoria del lettore, ma è certo che, superato un primo scoglio per familiarizzarsi con il linguaggio, la cosa diventa molto fruttuosa.

Capitolo 20

LaTeX: la geometria delle pagine

20.1 Introduzione

Uno degli argomenti più trattati nei forum degli utenti di TeX, non escluso il forum di GuIt, è quello dei margini, della giustezza del testo, delle testatine e dei piedini, di come fare per rendere la pagina del documento finale gradevole ed efficace.

Qualche volta si tratta di rendere un documento il più simile possibile ad una serie di altri documenti composti con altri sistemi; qualche altra volta si tratta di ricomporre un'opera antica, o comunque fuori commercio, per la quale si vuole mantenere il più possibile l'aspetto originale, e questo coinvolge anche la scelta dei font, non solo la geometria della pagina o il corpo dei caratteri.

Va allora detto prima di tutto che questo genere di attività può rendere un documento bellissimo, se il disegno della pagina e dei suoi elementi è fatto da una persona di gusto, colta, esperta di grafica editoriale o, almeno, competente a livello di 'intenditore', se non proprio di grafico professionista. Il risultato può essere pessimo se viene eseguito da una persona incompetente, ancorché animata dalle migliori intenzioni; peggio ancora se questa persona tenta di imitare lo stile di noti word processor, specialmente se i documenti scritti con quegli strumenti sono stati composti da persone che si sono divertite a scegliere corpo e stile dei caratteri senza nessun criterio in mente, spaziature, margini, inserimento di figure, creazione di tabelle, eccetera, fatte a seconda dell'ispirazione del momento e del comando più facile da raggiungere con un click di mouse.

Dietro il disegno di una pagina stampata ci sono in realtà quattro o cinque secoli di storia; considerando anche i codici manoscritti, una quindicina di secoli di storia. Questa lunghissima tradizione non può essere ignorata perché quello che la tradizione ha prodotto è quello che è sopravvissuto alla selezione naturale,

che ha tenuto conto dei mezzi a disposizione, ma anche della qualità del risultato confermata dall'apprezzamento dei lettori.

Non bisogna quindi cedere a qualunque tentazione compositiva, ma documentarsi al meglio e cercare di affinare il gusto per saper discernere i bei prodotti editoriali, dai prodotti accattivanti per colori, per font fantasiosi, per disposizione del testo che convoglia il messaggio, ma che non potrebbero essere letti da nessuno per più di due o tre pagine consecutive. Non si può confondere un opuscolo pubblicitario con un testo di poesie; l'uno e l'altro hanno i loro impieghi, ma non si possono comporre le poesie come una serie di réclame, e le réclame come dei poemi.

In parte si è trattato di questi argomenti nel capitolo 6; qui però si desidera andare più a fondo per mostrare non solo come fare, ma anche che cosa ci sta dietro; sapendo il perché, spesso si evitano errori stilistici che dovrebbero sempre essere evitati quando si cerca di comporre bene, secondo l'arte tipografica.

È chiaro a tutti che la parte più importante di un documento è il suo contenuto. Questo aspetto non verrà trattato in questa guida; esistono ottime grammatiche e testi di 'retorica' per scrivere in modo efficace e che sappia convogliare il messaggio giusto al lettore.

Uno degli aspetti più importanti in questa azione è proprio quello di eseguire una buona analisi del lettore; se si riesce a fare un buon profilo del lettore, allora è possibile confezionare il messaggio in modo che sia adatto a quel tipo di lettore. È facile capire che un testo destinato a una moltitudine di lettori dai profili differenti può essere difficilissimo da scrivere; se si scrive per i lettori più informati, quelli meno informati non capiscono; se si scrive per i lettori meno informati, gli altri si annoiano. Questo stesso testo di introduzione all'arte tipografica con LaTeX, che si rivolge ad una moltitudine di lettori, cerca di essere un compromesso fra i lettori esperti e quelli neofiti; sicuramente non è completamente adatto né ai primi né ai secondi, tuttavia si affida alla loro comprensione nella speranza che i primi possano trovare qualche spunto che non conoscevano e che i secondi siano invogliati ad andare più a fondo.

Certamente LaTeX consente di dedicarsi con tutta l'attenzione possibile al confezionamento del messaggio; alla sua corretta scrittura, alla strutturazione, all'evidenza da dare ai concetti base, eccetera. Tuttavia questi aspetti vengono sottolineati meglio da una particolare disposizione del testo sulla pagina, dai font usati, dalle decorazioni, dai colori, eccetera. Questi aspetti fanno parte della grafica editoriale; il grafico che si accinge a definire un layout particolare deve stare attento a non esagerare perché il layout non deve attrarre o distrarre l'attenzione del lettore, specialmente in un documento contenente un testo molto strutturato come può essere un libro. Il layout deve essere sobrio, aiutare il lettore nella sua lettura, evidenziare ciò che deve essere evidenziato, senza richiamare su di sé l'attenzione del lettore. Aprendo un libro qualsiasi ben disegnato si ha una immediata percezione dell'equilibrio grafico anche senza leggerne il testo; si percepisce una sensazione confortevole che invoglia a leggere. Un libro mal disegnato attrae l'occhio sui fronzoli, le decorazioni, i colori, ma distrae dalla lettura.

Vale la pena di riportare il commento di Peter Wilson, l'autore della classe *memoir*, a questo proposito[1]:

> L'essenza della buona tipografia è che essa non viene notata alla prima occhiata, forse nemmeno alla seconda o alle successive, se non da chi abbia un occhio esercitato. Se la vostra reazione iniziale quando guardate un libro è quella di esclamare qualcosa a proposito del suo layout, allora probabilmente quel libro è mal progettato, ammesso che sia mai stato progettato. La buona tipografia è fatta di sottigliezze, non di esagerazioni.

Qui di seguito si indicheranno alcuni aspetti della grafica editoriale che derivano dalla tradizione e si cercherà di spiegare da dove derivano alcune di queste 'norme' (piuttosto elastiche, per fortuna); si indicheranno anche alcuni pacchetti di estensione che consentono di personalizzare diverse parti del layout.

20.2 La geometria della pagina

Il lettore non si preoccupi se finora si è usata la parola 'layout' senza definirla. In inglese la parola deriva dalla locuzione verbale *to lay out* che vuol dire adagiare, distendere; ma in tipografia essa non implica solo la disposizione geometrica delle varie parti che compongono una pagina, ma anche i font e le loro dimensioni, gli spazi (i contrografismi) fra le varie parti del testo, e tanti altri dettagli più o meno evidenti, che non hanno a che fare solo con la geometria della pagina, ma hanno a che fare con la pagina nel suo complesso.

Bisogna innanzi tutto stabilire di che tipo di testo si vuole confezionare il disegno: è un testo di consultazione, un testo letterario, un romanzo, un saggio prevalentemente testuale, un testo tecnico-scientifico, un testo artistico con molte immagini; è un testo di lusso o si tratta di dispense; richiede una lettura continua, oppure può o deve essere letto saltando da una parte all'altra; deve essere stampato in grande formato o in formato tascabile.

Come si vede, già queste poche cose richiedono diverse scelte grafiche, alcune delle quali, talvolta, sono più di tipo economico che di tipo estetico.

20.2.1 Il formato delle pagine

In Italia un formato corrente per libri di ogni genere rilegati in brossura è quello la cui pagina rifilata ha le dimensioni di 170 mm per 240 mm; misurando alcuni libri presi a caso, chi scrive ha trovato dimensioni leggermente diverse, di solito non più del 10% dai valori indicati. Libri più importanti, rilegati con copertina rigida (incassati) possono avere dimensioni maggiori, anche molto maggiori. I

[1] The essence of good typography is that it is not noticeable at first, or even second or later, glances to any without a trained eye. If your initial reaction when glancing through a book is to exclaim about its layout then it is most probably badly designed, if it was designed at all. Good typography is subtle, not strident.

Pieghe	Nome	Formato del foglio da piegare					Pagine	Facciate
		A0	A1	A2	A3	A4		
1	folio	A1	A2	A3	A4	A5	2	4
2	quarto	A2	A3	A4	A5		4	8
3	ottavo	A3	A4	A5			8	16
4	sedicesimo	A4	A5				16	32
5	trentaduesimo	A5					32	64

Tabella 20.1: Formati delle pagine a seconda delle dimensioni del foglio di partenza e del numero di pieghe

libri di consultazione, anche se incassati, possono avere dimensioni minori, con la pagina in formato A5, cioè di 148 mm per 210 mm. Dimensioni minori, per esempio A6 di 105 mm per 148 mm, si incontrano in opuscoletti decisamente tascabili e di piccolo spessore.

La scelta dei formati dipende dalla carta disponibile e dal numero di pieghe che si possono fare per creare ogni segnatura. Oggi si tende a stampare a nastro continuo, mentre in passato si stampava su grandi fogli di dimensioni standardizzate. Il nastro continuo permette di tagliare delle sezioni più o meno lunghe che svolgono la funzione dei grandi fogli isolati del passato, ed è per questo che con il nastro continuo è facile avere più libertà nel formato della pagina. Tuttavia, una volta piegato il foglio un certo numero di volte, si ottiene una 'segnatura', un fascicoletto contenente un numero di facciate pari ad una potenza di due, se le pieghe sono sempre state fatte 'a metà'; talvolta si fanno delle pieghe in tre, per cui il numero di facciate di una segnatura può anche essere un multiplo di sei.

Un grande foglio piegato solamente in due produce una segnatura con due pagine e quattro facciate di solito di grandi dimensioni; in passato era comune, per esempio, per i messali; nelle nostre università, prima di essere sostituiti dalle registrazioni informatiche, i registri relativi agli studenti erano spesso dei libroni 'in folio' con le pagine di circa 400 mm per 600 mm.

Con un'altra piega a metà si hanno segnature con quattro pagine e otto facciate; sono segnature in 'quarto'. Con un'altra piega a metà si hanno segnature con otto pagine e sedici facciate; sono segnature in 'ottavo'. Via via che si piega a metà l'ordinale che indica la segnatura, sedicesimo, trentaduesimo, ricorda sempre una potenza di due; si capisce però che più si piega, più diventa spessa la piega dove bisogna cucire e meno precisa diventa la posizione della parte stampata su ogni facciata. La tabella 20.1 si riferisce alle piegature dei fogli della serie A ISO-UNI; il nome della piegatura, folio (o quartino), quarto, eccetera, dà luogo a formati sempre della serie A ma più piccoli. Evidentemente al crescere del numero di pieghe, le pagine e le facciate diventano sempre più piccole e sempre più numerose; la tabella in questione si riferisce comunque sempre a piegature a metà, non anche in terzi, per cui le successive piegature danno sempre luogo a

formati della serie A.

Eseguendo delle pieghe in tre si possono avere delle segnature in 'sesto', in 'dodicesimo o in 'ventiquattresimo, ma sono segnature meno frequenti e che richiedono fogli iniziali di dimensioni insolite.

A prescindere dal numero di pagine che formano ogni segnatura, questa rappresenta il risultato della piegatura dei grandi fogli e anche la 'materia prima' per la cucitura a macchina o a mano per confezionare il libro; quando questo è cucito, prima o dopo venire brossurato, sicuramente prima di venire incassato, il blocco delle segnature cucite viene rifilato su tre lati per avere le pagine tutte delle stesse dimensioni e per eliminare quei piccoli difetti di allineamento che potrebbero essersi formati durante la cucitura. Il libro brossurato può avere la copertina un poco più ampia delle pagine che racchiude; il libro incassato ha sempre la copertina che sporge oltre le pagine; il libro incassato spesso ha una sovracoperta, destinata a proteggere la copertina rigida che talvolta è realizzata con coperture di materiali pregiati come stoffa o pelle.

20.2.2 Le segnature e le imposizioni

Questa rapida scorsa sulla fabbricazione del libro permette di capire che il 'desk top publishing', a cui spesso ci si riferisce quando si usano sistemi di composizione 'fai da te', non può prescindere dalla conoscenza di queste regole elementari, anche quando il testo composto su fogli A4, eventualmente 'in folio' e perciò con pagine in formato A5, sono incollati a caldo e brossurati con metodi economici; anche in questi casi è necessario un passaggio alla taglierina per rendere i tagli superiore, inferiore ed esterno ben pareggiati e facili da far scorrere fra le dita. LaTeX può essere utile in questi casi; sicuramente è molto utile per il lavoro tipografico a foglio continuo, anche se il suo file di uscita contiene una sequenza di pagine non necessariamente nella sequenza giusta e ruotate quanto basta per comporre i due lati dei grandi fogli che formeranno ciascuna segnatura; questa operazione verrà eseguita in tipografia con software adeguato e con macchine adatte per marcare le grandi lastre di fotolitografia con le quali verrà eseguita la stampa con foglio continuo.

A casa, per produrre un fascicoletto o un libro in formato A5, il sistema TeX offre un certo numero di possibilità; dal punto di vista delle macchine bisogna immaginare di disporre di una stampante che possa stampare fronte e retro; in mancanza bisogna stare attenti a stampare prima tutti i fronti; poi ricaricare a rovescio il pacco dei fogli stampati da un lato per stamparli dall'altro; è ovvio che questa operazione richiede la massima attenzione per essere sicuri che ogni foglio risulti stampato correttamente sui due lati con quello che deve contenere. Meglio ancora sarebbe disporre di una stampante che accetti in entrata fogli in formato B4 e che possa stampare fronte e retro, con le attenzioni esposte sopra nel caso essa possa stampare solo fronte. Il vantaggio sarebbe che piegando a metà segnature su fogli B4, si ottengono pagine in formato B5, cioè di 176 mm per 250 mm, che ammettono abbastanza margine, ma senza eccessivo sfrido, per

stampare pagine in formato 170 mm per 240 mm, il formato della maggior parte dei libri in Italia.

Ma per stampare questi fogli è necessario riordinare le pagine; bisogna decidere innanzi tutto se si vogliono fare segnature di quattro fogli (otto pagine e sedici facciate) o segnature con più o meno fogli. Bisogna comporre il file in formato PDF, magari correggendo i margini affinché le pagine appaiano alla giusta distanza dalla piega centrale di ogni foglio A4[2]. Poi bisogna eseguire l'*imposizione*, cioè bisogna riordinare e ruotare le pagine affinché appaiano nell'ordine e nel verso giusto quando si compone la segnatura; per fare questo si invoca il pacchetto *pdfpages*: per esempio, con il file PDF che si chiami `libretto.pdf`, che contenga pagine in formato A5, e disponendo di una stampante A4, bisogna predisporre un piccolo file `.tex` così composto:

```
% File segnature.tex
\documentclass{article}
\usepackage[final]{pdfpages}
\begin{document}
\includepdf[pages=-,nup=1x2,landscape,signature=16]{libretto.pdf}
\end{document}
```

Compilando questo file tramite pdflatex si ottiene un file con le pagine A4 in posizione 'panoramica' (*landscape*), su ogni facciata della quale si trovano due pagine A5 estratte dal file iniziale, e ordinate correttamente per formare segnature di sedici facciate; quindi la prima segnatura avrà un primo foglio che sul fronte avrà le pagine 16 e 1 e sul retro le pagine 2 e 15; un secondo foglio con le pagine 14 e 3 sul fronte e 4 e 13 sul retro; un terzo foglio con le pagine 12 e 5 sul fronte e 6 e 11 sul retro; un quarto foglio con le pagine 10 e 7 sul fronte e 8 e 9 sul retro. Secondo questa stessa sequenza vengono imposte le ulteriori pagine sui fogli che formeranno le segnature successive.

Stampando il file `segnature.pdf` sulla stampante A4 in modalità fronte e retro, i vari fogli vengono stampati in ordine; ogni quaterna di fogli contiene già una segnatura che basta piegare a metà per poi impilarla con le successive segnature in bell'ordine; non resta che incollare o cucire e poi brossurare o incassare. Per maggiori dettagli ci si può riferire alla documentazione del pacchetto *pdfpages*.

Esiste anche il pacchetto *booklet* per ottenere più o meno lo stesso risultato; le differenze sostanziali fra le strategie di imposizione dei due pacchetti sono essenzialmente queste:

- *pdfpages* presuppone che il documento di cui si devono creare le imposizioni sia già composto in formato PDF; *booklet* invece compone il documento a partire dal solito file sorgente producendo in uscita un file PDF contenente le imposizioni. Questo implica, specialmente con le segnature con più

[2]Si suppone che a casa si disponga dei una normale stampate per fogli A4; se si disponesse di una stampante per fogli A3, quanto viene detto qui vale per un fascicolo in formato A4, oppure per *segnature* in sedicesimo (otto fogli, sedici pagine, 32 facciate) in formato A5.

pagine, che si metta a dura prova l'intera memoria del programma di composizione; questa possibilità è menzionata anche nella documentazione di `booklet`.

- Il testo da comporre potrebbe non avere un numero di pagine complessivo pari ad un multiplo intero del numero di pagine delle segnature specificate; entrambi i pacchetti aggiungono il numero necessario di pagine bianche all'ultima segnatura. Ma `pdfpages` aggiunge un numero di pagine tale che l'ultima segnatura abbia lo stesso numero di pagine delle altre, mentre `booklet` consente che l'ultima segnatura abbia un numero di pagine multiplo di 4, anche se questa segnatura dovesse avere meno pagine delle altre.

Il testo di Gustavo Cevolani [15] spiega molto dettagliatamente il procedimento delle imposizioni con molti esempi ed esegue un confronto accurato dei due pacchetti.

Naturalmente per un lavoro non casalingo basta predisporre il file PDF, magari con i *crocini* (vedi più avanti) e in tipografia verranno eseguite le imposizioni corrette per la produzione delle segnature sui grandi fogli da piegare con diverse pieghe, per poi passare le segnature alla sezione di cucitura o incollatura e poi alla sezione di confezionamento finale con la coperta in brossura o incassata, prodotte da una linea di produzione diversa.

Con riferimento alla figura 20.1 vale la pena di osservare come sono eseguite le imposizioni per generare le segnature su un foglio piegato a metà, piegato due volte a metà e piegato tre volte a metà, oppure, per la segnatura in sesto, piegato in tre e poi a metà. Le pieghe si suppongono sempre eseguite piegando a metà (o in tre parti) il lato lungo del foglio o del fascicoletto parziale; le pieghe disegnate in rosso rappresentano quella che diventerà il dorso della segnatura, la piega sulla quale viene eseguita la cucitura o la spillatura; le pieghe disegnate in nero saranno eliminate con la rifilatura dei tre bordi esterni del fascicolo o del libro.

20.2.3 I crocini

Per eseguire il lavoro in tipografia è indispensabile poter disporre di una estensione a LaTeX che permetta di segnare i 'crocini' in italiano, i 'crop marks' in inglese; si tratta di segni spesso a forma di croce che permettono di registrare con precisione le varie pagine durante il lavoro fatto in tipografia, in modo che le stampe delle due facciate di una stessa pagina siano esattamente allineate: guardando in controluce, il margine sinistro del testo della facciata sul verso deve corrispondere esattamente con il margine destro del testo stampato sul recto della stessa pagina e lo stesso deve succedere per tutti gli altri margini corrispondenti.

Ogni compositore può crearsi i suoi crocini ricorrendo all'ambiente *picture* come nella figura 20.2, che mostra un esempio predisposto per una tipografia italiana; in essa appaiono sia i mezzi crocini che indicano i limiti della pagina rifinita e i crocini interi che servono per registrare con precisione le imposizioni. Non è difficile fare una cosa come quella mostrata nella figura 20.2, ma è delicata;

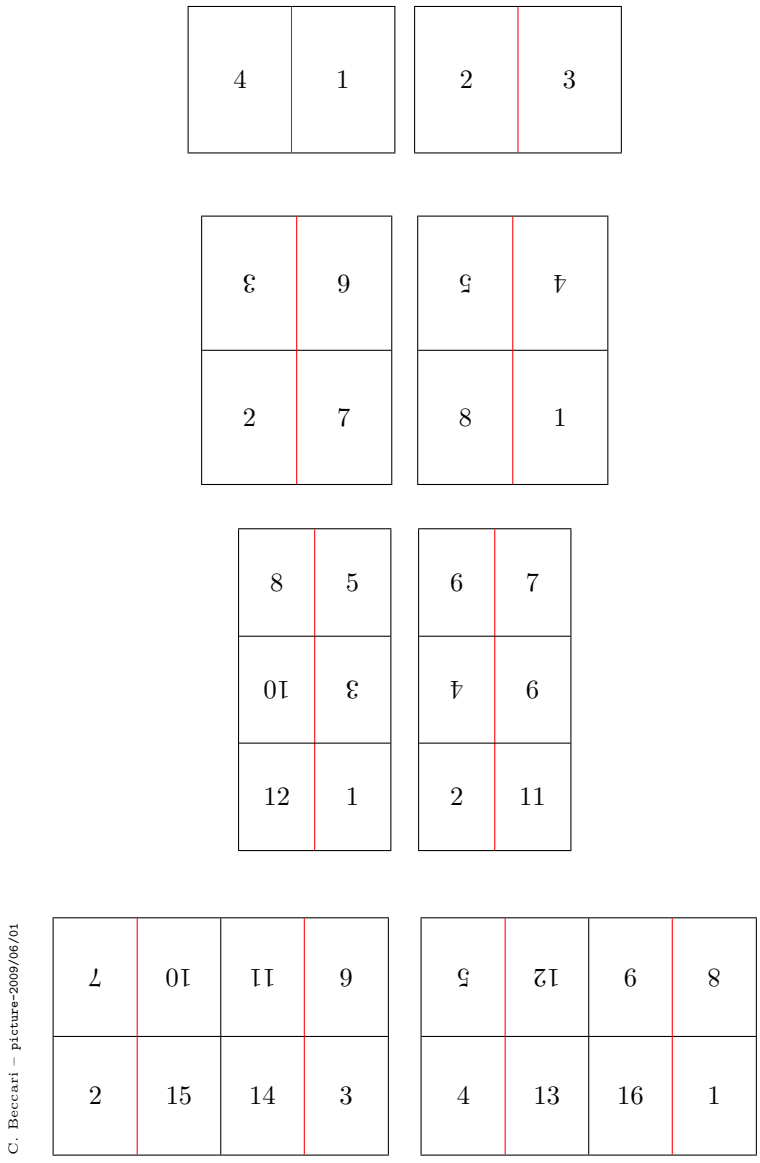
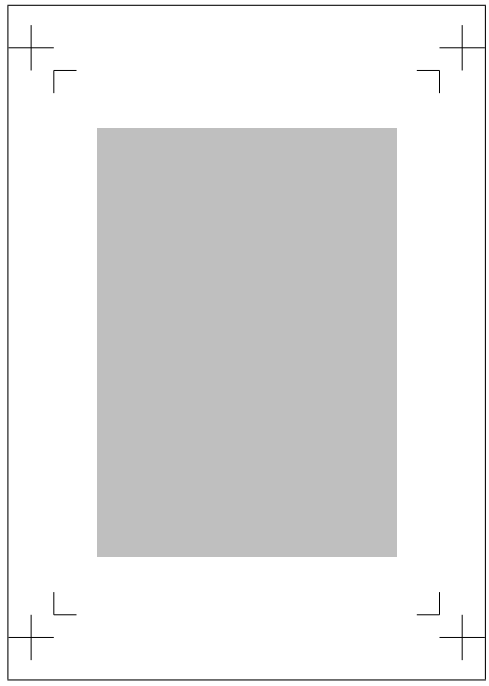


Figura 20.1: Le imposizioni per generare una segnatura in folio, in quarto, in sesto e in ottavo



C. Beccari – picture-2009/06/01

Figura 20.2: Esempio di crocini predisposti per una tipografia italiana

in alternativa si può ricorrere al pacchetto *crop* che provvede da solo a disporre i crocini (preconfezionati secondo la tradizione USA) nelle posizioni specificate in relazione alle dimensioni della pagina rifinita. La documentazione si trova in `.../doc/latex/crop/crop.pdf`.

20.2.4 Dimensioni della gabbia del testo

Su ogni facciata va collocata *la gabbia del testo* stampato, chiamata anche *specchio di stampa* o *gabbia interna*; questa è la parte più significativa della pagina e va dimensionata con cura. Non è solo una questione di margini: il rettangolo che contiene il testo deve essere adeguato al testo che esso contiene e ai font con cui è scritto.

Studi di carattere psicosociologico hanno messo in evidenza che il lettore medio si trova a suo agio nel leggere se le righe non contengono più di 65–70 caratteri, spazi e segni di interpunzione compresi (regola di Bringhurst). La scelta del font e la scelta del corpo normale, quello con cui è scritto il testo corrente, diventano perciò fondamentali per definire la giustezza, cioè la base del rettangolo che circoscrive il testo.

Esistono tabelle per determinare la giustezza ottimale per ciascun font e per ogni corpo dello stesso font; tuttavia non è facile procurarsi queste tabelle per

qualunque font. LaTeX però offre un metodo semplice e pratico per determinare la giustezza migliore con un dato font. Basta predisporre un piccolo file di prova nel quale si sceglie il font e con quello si misura la lunghezza dell'alfabeto minuscolo.

Per esempio, con il carattere Latin Modern, il semplice file seguente permette di sapere che la larghezza dell'alfabeto minuscolo di corpo 10 pt vale 137,58371 pt:

```
\documenctlass{minimal}
\newlength{\alfabeto}
\begin{document}
\settowidth{\alfabeto}{abcdefghijklmnopqrstuvwxyz}
L'alfabeto minuscolo LM a 10\,pt vale \the\alfabeto.
\end{document}
```

Sapendo che ci sono 72,27 punti in un pollice e che un pollice vale 25,4 mm, si può determinare una volta per tutte la dimensione in millimetri di un punto; precisamente la costante di conversione vale 0,351 459 804 mm/pt. Con questa costante di conversione possiamo quindi determinare che l'alfabeto minuscolo del font Latin Modern di 26 lettere è lungo 48,36 mm, quindi si ha mediamente 1,859 813 mm/lettera. Questo semplice conto svolto una volta per tutte permette di calcolare che con questo font la giustezza ottimale di 65–70 caratteri per ogni riga vuol dire una giustezza da 120 mm a 130 mm.

Esercizio 20.1 Questo testo è stato composto con il carattere Latin Modern a 10 punti con la classe *book* senza modifiche sostanziali. Il file `bk10.clo`, che specifica le misure relative al font di 10 pt, dice che la giustezza `\textwidth`, quando si compone ad una colonna, è impostata a 345 pt; la giustezza è nell'intervallo indicato sopra?

Il conto appena fatto potrebbe sembrare troppo semplificato; non si è tenuto conto degli spazi e dei segni di interpunzione; siccome gli spazi interparola sono dello stesso ordine di grandezza di una 'e' (anche se possono risultare un poco allargati o ristretti per la giustificazione) e i segni di interpunzione più frequenti (il punto e la virgola) contano poco, anche se contano gli spazi da cui sono seguiti, il conto appena fatto, benché approssimato, fornisce una eccellente stima della lunghezza della riga ideale.

A titolo di esempio nella tabella 20.2 sono riportate le giustezze in millimetri per righe composte con 60, 65, 70 e 75 caratteri con diversi font. I dati relativi ai font CM e LM sono praticamente identici, mentre si osservano diversità sensibili nelle giustezze ottimali con i font TX e PX. Ma una differenza che non emerge a colpo d'occhio è la seguente: rispetto alle giustezze relative al corpo di 10 pt, le giustezze con corpi maggiori o minori per i font TX e PX sono strettamente proporzionali al corpo, perché con questi font LaTeX lavora solamente con il corpo di 10 pt ingrandito o rimpicciolito a seconda della necessità. Con i font CM ed LM cade la proporzionalità, perché LaTeX con questi font usa caratteri disegnati apposta per ciascun corpo[3]. Il corpo di 10 pt è il corpo del testo corrente con

[3] Veramente il corpo indicato con 11 pt in realtà è di 10,95 pt e viene ottenuto per ingrandimento del font CM a 10 pt, e per rimpicciolimento del font LM a 12 pt.

Giustezze in millimetri				
	Numero di caratteri			
Corpo	60	65	70	75
8	87	95	102	109
9	95	103	111	119
10	103	112	120	129
11	113	122	132	141
12	121	131	141	151

(*a*) Font CM

Giustezze in millimetri				
	Numero di caratteri			
Corpo	60	65	70	75
8	87	95	102	109
9	95	103	111	119
10	103	111	120	129
11	111	120	129	138
12	121	131	141	151

(*b*) Font LM

Giustezze in millimetri				
	Numero di caratteri			
Corpo	60	65	70	75
8	77	83	90	96
9	87	94	101	108
10	96	104	113	121
11	106	115	124	133
12	116	125	135	145

(*c*) Font TX

Giustezze in millimetri				
	Numero di caratteri			
Corpo	60	65	70	75
8	86	93	100	108
9	97	105	113	121
10	108	117	126	135
11	118	128	138	148
12	129	140	151	162

(*d*) Font PX

Tabella 20.2: Giustezze determinate con il metodo della lunghezza dell'alfabeto. Le giustezze sono espresse in millimetri e i valori sono arrotondati al valore intero. (*a*) font CM (Computer Modern); (*b*) font LM (Latin Modern); (*c*) font TX (Times eXtended); (*d*) font PX (Palatino eXtended).

l'opzione *10pt*, il corpo 11 pt è relativo all'opzione *11pt* e, corrispondentemente, il corpo di 12 pt si riferisce all'opzione *12pt*; per il corpo di 10 pt il corpo di 9 pt si riferisce al comando `\small` e quello di 8 pt al comando `\footnotesize`. Sotto questo aspetto i font nativi del sistema TEX sono migliori di quelli commerciali distribuiti nel solo corpo a 10 pt. Fra i font commerciali esistono anche font 'esperti' che hanno forme diverse a seconda del corpo, ma sono meno diffusi.

Il font usato determina pertanto un intervallo ottimale entro cui scegliere il valore della giustezza. Come si determina l'altezza? La risposta dipende molto dalla natura della testatina e del piedino.

20.2.4.1 Testatine e piedini; il pacchetto *fancyhdr*

Bisogna innanzi tutto decidere che cosa contengano le testatine e i piedini delle pagine dispari e di quelle pari. Se il testo è un romanzo, le testatine e i piedini servono a poco e possono essere anche vuoti o uno dei due può contenere solo il numero della pagina; quest'ultimo, se è l'unica indicazione oltre al testo presente sulla pagina potrebbe essere collocato nel piedino, centrato o a filo del margine

esterno del blocco di testo, oppure potrebbe essere collocato in testa o al piede del margine esterno della pagina (come una nota marginale). In entrambi i casi esso non impegna visivamente la pagina, quindi l'altezza del blocco di testo contiene effettivamente solo il testo.

Se invece il testo è un documento di consultazione, le testatine destra e sinistra, e i piedini destro e sinistro giocano un ruolo essenziale nella 'navigazione' del testo alla ricerca delle informazioni che interessano. Ecco allora che il pacchetto *fancyhdr* diventa utilissimo per costruire testatine e piedini fortemente strutturati che aiutino davvero nella navigazione del testo; questo pacchetto consente di gestire l'informazione variabile di pagina in pagina da collocare nelle testatine di sinistra e di destra; lo stesso vale per i piedini. Il pacchetto consente di immaginare ogni testatina o piedino diviso in tre parti, quella di sinistra, quella di centro e quella di destra e di scrivere in ciascuna parte una informazione diversa. La documentazione del pacchetto *fancyhdr* si trova in `.../doc/latex/fancyhdr/fancyhdr.pdf`.

Si capisce subito che questo genere di testatine e di piedini fortemente strutturati possono essere molto utili nei rapporti tecnico-scientifici; adeguatamente predisposti possono indicare in un vocabolario il primo e l'ultimo lemma che compaiono in una pagina. Ma quello che è più importante è che questi elementi di riferimento sono otticamente 'ingombranti' e quindi bisogna ritagliarli dal rettangolo del testo, tenendo conto anche di un adeguato spazio di separazione, eventualmente anche di un filetto di delimitazione.

20.2.4.2 Le proporzioni della gabbia di testo

Stabilito se le testatine e/o i piedini facciano o non facciano parte della gabbia del testo, bisogna decidere quali proporzioni abbia la gabbia del testo, il rettangolo che racchiude il testo, in relazione al rettangolo che rappresenta la pagina rifinita. Si può decidere che abbia le stesse proporzioni (lo stesso rapporto altezza/base) come si può optare per proporzioni diverse. Ma non si può prendere questa decisione se non si prendono in considerazione anche i margini e ciò che eventualmente questi devono contenere.

Una decisione preliminare molto importante è se si stampa solo sul recto o sul recto e il verso (in bianca e volta, come si dice in gergo tipografico). In realtà se si stampa solo in bianca, lo stile grafico perde abbastanza la sua importanza, perché si sta parlando di un fascicolo o incollato a caldo o semplicemente inserito in una legatura a spirale, o a punti o ad anelli. Ciò non toglie che non si debba prendere qualche precauzione per evitare che una parte di testo risulti illeggibile.

Normalmente se si stampa solo in bianca le proporzioni della gabbia del testo rispecchiano le proporzioni della pagina fisica, eventualmente depurata dell'ulteriore margine a sinistra per tener conto della legatura artigianale o dei fori per gli anelli; su un foglio A4 si può pensare di assegnare circa 7-8 mm per la rigidezza della incollatura a caldo o della legatura a spirale, un poco di più per i fori per i raccoglitori ad anelli. La larghezza utile della pagina diventa perciò da

202-203 mm a 195 mm e la proporzione altezza su larghezza va da circa 1,47 a circa 1,52, invece del valore di $1.41 \approx \sqrt{2}$ tipico della serie UNI.

Per esempio, per un raccoglitore ad anelli il rapporto utile vale 1,52. Se la giustezza è di 125 mm, allora l'altezza della gabbia del testo deve essere di 190 mm. Centrando la pagina nello spazio utile, i margini superiore e inferiore diventano di $(297\,\text{mm} - 190\,\text{mm})/2 = 53{,}5\,\text{mm}$. I margini destro e sinistro diventano invece diversi se si deve tenere conto dei 15 mm assegnati allo spazio per i fori degli anelli; il margine destro diventa $(210\,\text{mm} - 15\,\text{mm} - 125\,\text{mm})/2 = 35\,\text{mm}$ e il margine sinistro è 15 mm più grande e risulta 50 mm; come controprova: $50\,\text{mm} + 125\,\text{mm} + 35\,\text{mm} = 210\,\text{mm}$.

Per la stampa in bianca e volta c'è maggiore libertà, sia pure tenendo conto che le pagine destre e sinistre sono diverse, se non altro perché i margini interni devono essere diversi dai margini esterni; anche in questo caso si può tenere conto di qualche millimetro per contrastare l'effetto di curvatura in corrispondenza del margine interno, il margine di cucitura. Tuttavia non si è tenuti a rispettare la stessa proporzione della pagina fisica. Anzi, i grandi book designer giocano con la forma della pagina e la forma della gabbia del testo per ottenere effetti speciali. Bringhurst, per esempio, gioca con i rapporti geometrici come se fossero i rapporti di note musicali sia della scala diatonica sia di quella cromatica e ricerca i rapporti che formino accordi gradevoli; altri invocano il rapporto aureo; altri invocano i rapporti che esistono negli elementi delle figure geometriche, in particolare dei poligoni regolari. Un esempio è dato dal disegno riportato nel capitolo 6 nella figura 6.2. Definita la diagonale di un poligono regolare come la corda più lunga, allora la diagonale di un triangolo equilatero sta al lato in rapporto unitario e dà quindi luogo ad una gabbia quadrata. Con il quadrato il rapporto è $\sqrt{2}$; con il pentagono il rapporto è quello aureo 1,618...; con l'esagono il rapporto vale 2, con l'ettagono il rapporto vale 2,247; con l'ottagono il rapporto vale 2,613. Raramente si usano rapporti così grandi (superiori a 2) ma non sono esclusi per particolari documenti; in un catalogo di una casa editrice ho trovato una gabbia con il rapporto pari a 2,25, praticamente il rapporto dell'ettagono.

20.2.4.3 I margini

Tradizionalmente i margini inferiore e superiore stanno fra di loro come il margine esterno sta al margine interno. Inoltre in una coppia di pagine affacciate (in inglese uno *spread*) spesso si desidera che la coppia di margini interni siano complessivamente equivalenti a ciascuno dei margini esterni. Esaminando il layout di diversi testi antichi e moderni, Wilson ha trovato che in realtà questa tradizione non era rispettata nemmeno secoli fa, sebbene moltissimi libri pubblicati in passato la rispettassero. Wilson ha raccolto diversi disegni di layout antichi e moderni nella breve documentazione storica che si può leggere con `texdoc memdesign`.

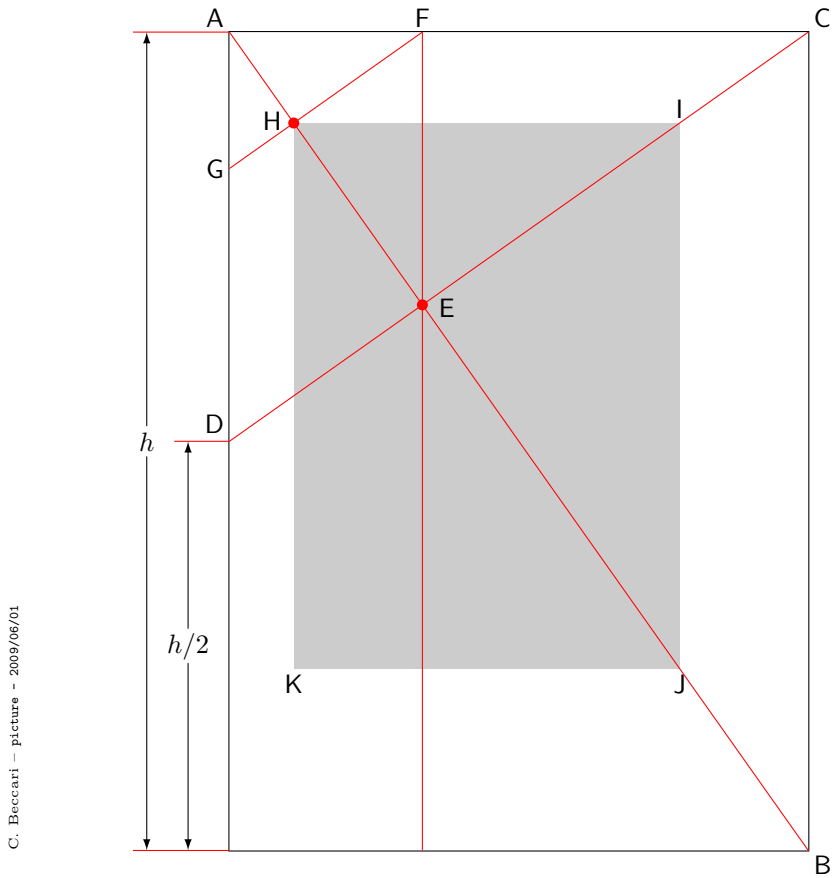


Figura 20.3: Costruzione geometrica usata da Gutenberg per determinare le proporzioni e la posizione della gabbia delle sue famose Bibbie

Per seguire questa regola gli antichi usavano squadra e compasso ed eseguivano delle ingegnose costruzioni geometriche per determinare le proporzioni della gabbia e dei margini.

Gutenberg avrebbe disegnato la gabbia delle sue famose Bibbie e di diversi altri stampati seguendo la costruzione della figura 20.3. La costruzione è più lunga da commentare che non da valutare graficamente. Comunque si basa sul metodo delle due diagonali. La diagonale AB indicata nella figura è la diagonale della pagina di destra, mentre la diagonale CD è la diagonale delle due pagine affacciate, ed è per questo che interseca il margine sinistro della pagina di destra a metà altezza. Le due diagonali si intersecano nel punto E per il quale si traccia la verticale EF; di qui si traccia la parallela FG alla diagonale delle due pagine affacciate; il punto di intersezione H di questa parallela con la diagonale della pagina individua il primo vertice della gabbia dal quale è semplice tirare le debite linee verticali ed orizzontali per completare il rettangolo HIJK della gabbia.

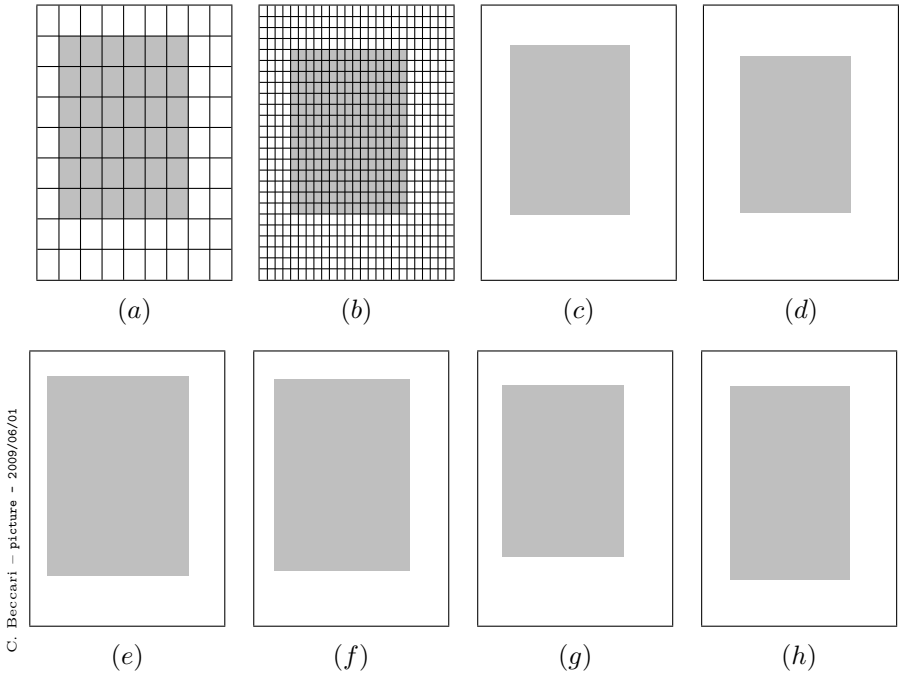


Figura 20.4: Determinazione della gabbia e dei margini con il metodo delle strisce. (*a*) esempio con 9 strisce e rapporto 2 : 1; (*b*) esempio con 25 strisce e rapporto 3 : 2; (*c*) esempio con il rapporto aureo – si noti che la gabbia non è un rettangolo aureo; (*d*) esempio con il rapporto 4 : 3. Nella seconda riga viene rappresentato il metodo delle strisce riferito solo al rapporto dei margini: (*e*) metodo dei margini con 11 strisce; (*f*) metodo delle 10 strisce usato dalla classe `octavo`; (*g*) metodo dei margini con 8 strisce; (*h*) gabbia aurea con margini aurei su foglio A4

Analizzando la costruzione geometrica si determina abbastanza facilmente che il risultato della costruzione di Gutenberg è lo stesso di quello che si ottiene con il metodo delle nove strisce descritto qui di seguito. Infatti la prima metà della costruzione determina il punto E tale che il segmento AE è un terzo del segmento AB; la seconda parte della costruzione determina il punto H in modo che nuovamente il segmento AH sia un terzo del segmento AE; perciò il margine interno e quello superiore sono rispettivamente un nono della larghezza e dell'altezza della pagina rifinita sulla base della quale si è fatta la costruzione geometrica. Per la similitudine dei vari triangoli che hanno generato il margine interno e quello superiore, i margini esterno e quello inferiore sono esattamente il doppio, quindi due noni della corrispondente dimensione della pagina rifinita.

La costruzione più semplice consiste nel dividere il rettangolo della pagina rifinita un un certo numero di strisce uguali sia in verticale sia in orizzontale; due strisce verticali esterne vengono riservate al margine esterno e due strisce orizzontali in basso vengono riservate al margine inferiore; una striscia orizzontale

in alto viene riservata al margine superiore e, infine, una striscia verticale viene riservata al margine interno. In questa maniera le proporzioni della gabbia del testo risultano uguali a quelle della pagina; il margine interno è metà del margine esterno e il margine superiore è la metà del margine inferiore. La figura 20.4 permette di seguire la costruzione su un foglio con le proporzioni della carta A4; nella costruzione della figura (a) viene usato il rapporto $2 : 1$ tra la larghezza del testo e la larghezza complessiva dei margini, così come tra la larghezza del margine esterno rispetto al margine interno, e lo stesso viene fatto per le altezze. Quindi sei strisce per il testo e tre complessivamente per i margini, delle quali due per il margine maggiore e una per il margine minore

Sempre usando le strisce, ma con un rapporto diverso, si può ottenere una costruzione differente, ma che conserva la stessa proporzione fra le dimensioni della pagina e quelle della gabbia. Per esempio se si sceglie il rapporto $3 : 2$, occorrono complessivamente 25 strisce: 4 per il margine piccolo, 6 per il margine grande, 10 in totale per i margini, 15 per la gabbia; questo è quello che si è fatto nella figura 20.4(b).

Sviluppando i calcoli in modo algebrico, senza fare riferimento alle strisce (che diventerebbero infinite) si potrebbe calcolare la divisione della pagina con il rapporto aureo; il margine piccolo sta a quello grande come $1 : \phi$; e la somma dei margini sta al testo ancora come $1 : \phi$. Quindi il margine piccolo sta a quello grande che sta al testo come la proporzione $1 : \phi : \phi(1 + \phi)$. Svolgendo i calcoli, si ottiene che se w è la dimensione orizzontale o verticale della pagina rifinita, allora le dimensioni orizzontali o verticali rispettivamente dei margini e della gabbia sono $0{,}145\,898w$ per il margine piccolo, $0{,}236\,068w$ per il margine grande e $0{,}618\,034w$ per la gabbia. Ciò è quanto si vede nella figura 20.4(c).

La situazione è diversa per il disegno (h). In questo disegno si impone che la gabbia sia un rettangolo aureo, quindi la base della gabbia sta alla sua altezza come 1 sta a ϕ. Il foglio è un rettangolo UNI (A4) con rapporto fra la base e l'altezza pari a $\sqrt{2}$; se si impone il rapporto aureo anche fra i margini piccoli e quelli grandi, questo può essere fatto indipendentemente dalle dimensioni della gabbia. Quindi è possibile imporre, per esempio che la base della gabbia abbia il rapporto aureo rispetto alla somma dei margini laterali, ma non si può imporre la stessa relazione fra l'altezza della gabbia e la somma dei margini superiore e inferiore. Infatti, eseguiti i calcoli[4], si ottiene il disegno (h) che è piuttosto diverso dal disegno (c).

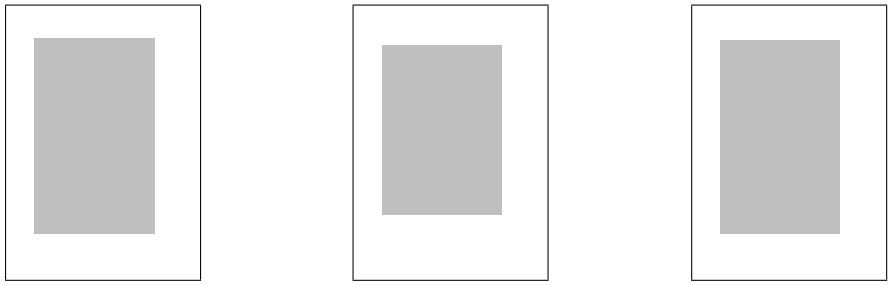
Nei primi tre schemi della figura 20.4 i rapporti numerici indicati sono sempre costituiti da due successivi numeri di Fibonacci[5]; è noto che la sequenza di Fibonacci ha il rapporto fra due successivi termini che tende al rapporto aureo; non è un caso che la figura 20.4(c) assomigli molto alla figura 20.4(b), benché in

[4] I calcoli danno:

base del foglio	210,000 mm	margine interno	30,639 mm
altezza del foglio	297,000 mm	margine esterno	49,574 mm
base della gabbia	129,787 mm	margine superiore	33,231 mm
altezza della gabbia	210,000 mm	margine inferiore	53,769 mm

[5] La sequenza di Fibonacci comincia con 1, 1 e prosegue con ogni elemento costituito dalla somma dei due precedenti, quindi: 1, 1, 2, 3, 5, 8, 13, ...

C. Beccari – picture - 2009/06/01



(*a*) (*b*) (*c*)

Figura 20.5: Confronto fra tre gabbie che sulla carta UNI A4 tengono conto del rapporto aureo. (*a*) gabbia ottenuta con il pacchetto `layaureo`; (*b*) gabbia con proporzioni auree fra le dimensioni della gabbia e i corrispondenti margini piccolo e grande; (*c*) gabbia aurea con margini aurei

questa il rapporto $3 : 2 = 1{,}5$ non sia tanto vicino al rapporto aureo 1,618 034...; ma la costruzione della figura 20.4 basata sulle strisce consente anche rapporti formati da numeri che non appartengono alla sequenza di Fibonacci. Se si sceglie il rapporto $4 : 3$ le strisce occorrenti sarebbero 49 e si omette di disegnarle; si ottiene la figura 20.4(*d*) e, come si vede, le differenze non sono enormi.

Visto che in diverse figure si sono presentati dei disegni della pagina con margini aurei, si è riportata ancora la figura 20.5 dove sono messi a confronto i tre schemi, ripetendone le caratteristiche:

(*a*) In questo schema prodotto dal pacchetto `layaureo` la gabbia ha la base pari alla lunghezza ottimale in base alla regola di Bringhurst, aumentata di 4 pc – secondo la tabella 20.2 la giustezza conterrebbe 75 caratteri del font Latin Modern; l'altezza in rapporto aureo con la base; il margine interno e il margine esterno in rapporto aureo fra di loro; il margine superiore $\sqrt{2}$ volte più piccolo del margine inferiore. La pagina è A4 e la larghezza della gabbia è di 31 pc, per il font da 10 pt, come specificato sopra. Secondo il pacchetto `layaureo` la gabbia risulterebbe linearmente più grande di circa il 6% per il corpo di 11 pt e di un altro 6% più grande per il corpo di 12 pt.

(*b*) In questo schema i margini, sia in orizzontale sia in verticale, sono in proporzione aurea e la loro somma è in proporzione aurea con la dimensione orizzontale o verticale della gabbia; il rettangolo della gabbia ha quindi le stesse proporzioni della pagina.

(*c*) In questo schema, invece, il margine interno è in proporzione aurea con il margine esterno e la loro somma in proporzione aurea con la base della gabbia; l'altezza della gabbia è in proporzione aurea con la base; il margine superiore e il margine inferiore sono in proporzione aurea fra di loro, ma non con l'altezza della gabbia.

In questi tre esempi la giustezza è sempre circa 4 pc (circa 17 mm) più grande di quella raccomandata dalla regola di Bringhurst, ma probabilmente solo di cinque o sei caratteri in più, quindi una misura assolutamente accettabile. Si vede ad occhio che, almeno per la composizione in corpo di 10 pt, le gabbie (a) e (c) sono praticamente identiche così come lo sono i margini. Con lo schema (a) usando un corpo maggiore per il font normale si avrebbe un riempimento della pagina significativamente maggiore.

Come si vede solo gli schemi (a) e (c) hanno la gabbia con la base e l'altezza in proporzione aurea, mentre nello schema (b) la gabbia ha la base e l'altezza in proporzione di $\sqrt{2}$. Nello schema (a) la base della gabbia è determinata in base alla regola di Bringhurst, mentre negli schemi (b) e (c) la base è determinata in base alla proporzione aurea rispetto alla somma dei margini. Le tre filosofie si rifanno tutte e tre al numero aureo, ma ne fanno un uso completamente diverso. Sta al compositore scegliere quella delle tre soluzioni che considera più gradevole, come sta a lui decidere se rifarsi al numero aureo o al metodo delle strisce.

Alcuni pacchetti determinano la gabbia e i margini riferendosi alle strisce, usando sempre una striscia per il margine piccolo, e due strisce per il margine grande, ma senza imporre nessun rapporto particolare fra le dimensioni della gabbia e quelle dei margini, lasciando, quindi, libera la scelta del numero delle strisce; nei disegni, (e), (f), (g), della figura 20.4 sono rappresentate, sempre con riferimento ad un foglio A4, le gabbie che si ottengono rispettivamente con 11, 10 e 8 strisce (il disegno con nove strisce è il disegno (a)). Si vede dunque che il metodo delle strisce realizzato dal pacchetto `typearea` o per le 10 strisce dalla classe `octavo` è molto versatile; volendo, con numeri alti di strisce, si può stabilire anche un rapporto diverso fra le strisce da dedicare al margine piccolo rispetto al margine grande, ed è quello che i parametri opzionali al pacchetto `typearea` consentono di fare.

Si osservi infine che in tutti i disegni della figura 20.4, tranne il disegno (h), la gabbia ha sempre le stesse proporzioni del foglio; inoltre per costruzione i margini grandi stanno ai margini piccoli come l'altezza del foglio sta alla base. Per cui la gabbia del testo rimane sempre adagiata con la sua diagonale sulla corrispondente diagonale del foglio. L'unico disegno che si discosta da questa proprietà è per costruzione il disegno (h), e quindi la diagonale della gabbia non giace sulla diagonale del foglio. Tuttavia la differenza non è enorme, ma alla fin fine il disegno (h) presenta una maggiore copertura del foglio con margini non troppo piccoli e ben proporzionati.

Il metodo delle strisce permette di suddividere anche le strisce destinate ai due margini in modo diverso fra i margini verticali e quelli orizzontali; è quindi un metodo molto versatile. Tuttavia si richiama ancora il lavoro di Wilson dove in molti casi le suddivisioni fra i margini non rispettano affatto la regola delle strisce e non ci si avvicina affatto al rapporto aureo. Appaiono curiosi gli esempi in cui il margine interno è vistosamente maggiore del margine esterno. Un caso particolare, che ha la sua giustificazione, è il libro di Knuth, *Concrete Mathematics*, per il quale l'autore ha sviluppato anche la famiglia di font denominata Concrete Computer Modern, con la sigla CC. A parte l'interesse specifico per il contenuto, il

layout della pagina presenta un margine interno vistosamente maggiore di quello esterno, ma anche pieno di note marginali sistemate in questo margine, dove riporta le battute dei suoi studenti in corrispondenza degli argomenti trattati. Questa libertà stilistica e di contenuto è forse tipica dello humor anglosassone, ma certamente la lettura di quelle battute alleggerisce molto la difficoltà della materia. Ecco quindi che con uno scopo preciso in mente si possono 'violare' le regole dettate dalla tradizione.

20.3 Lo scartamento e i contrografismi verticali

Sarebbe desiderabile che l'altezza della gabbia, in particolare quella che contiene il testo corrente abbia uno scartamento costante e contenga un numero intero di righe. La cosa non è semplicissima da calcolare, perché la prima riga del testo, non dovendo essere distanziata da nessuna riga precedente, ha una altezza convenzionale indicata da `\topskip`, quasi sempre pari a 10 pt, mentre le distanze fra le righe di base di tutte le righe successive sono pari a `\baselineskip`. Per calcolare l'altezza del testo, `\textheight`, in modo che contenga un numero intero di righe, bisogna eseguire dei calcoli fra numeri interi, con una aritmetica che non è familiare a chi è abituato ai calcoli a mano, ma che viene benissimo quando si eseguono i calcoli a macchina; si tratta di eseguire le divisioni conservando la parte intera del quoziente e gettando via il resto.

Scelto l'avanzamento di riga ottimale `\normalbaselineskip` per il font di corpo normale prescelto (anche questa una operazione delicatissima, che occupa diversi capitoli in ogni libro di tipografia) si può eseguire il calcolo in questa maniera: dal calcolo della gabbia si sottrae lo spazio destinato alle testatine, e ai piedini e ai loro contrografismi se si è scelto che tutti questi elementi facciano parte della gabbia; si ottiene quindi una prima approssimazione di `\textheight` che chiameremo x_1, si toglie `\topskip`, che chiameremo y e si divide per `\normalbaselineskip`, che chiameremo z; il risultato di questa divisione in generale non è un numero intero, ma la divisione che i programmi del sistema TEX eseguono è una divisione intera, quindi il risultato è la parte intera del quoziente, che chiameremo N; l'altezza della scatola del testo corrente non sarà quindi x_1, come assunto in partenza, ma sarà il nuovo valore x_2 calcolato come segue:

$$x_2 = Nz + y \qquad \text{dove} \qquad N \text{ è la parte intera di } \left(\frac{x_1 - y}{z}\right)$$

Esercizio 20.2 Per un romanzo privo di testatina e con un piedino contenente solo il numero della pagina, e che quindi non si considera facente parte della gabbia, `\textheight` coincide con l'altezza della gabbia. Se si usa il metodo delle strisce e il rapporto 2 : 1 con una pagina rifinita di 170 mm per 240 mm, e si è scelto il font Computer Modern di corpo 10 pt e un avanzamento di riga normale di 12 pt, quanto deve valere `\textheight` perché contenga un numero intero di righe?

Si può procedere anche a rovescio: scelto lo scartamento che si vorrebbe usare, z_1, si calcola quante righe N_t *totali* stanno nell'altezza desiderata; poi si ricalcola il nuovo scartamento z_2 in modo che nell'altezza specificata stiano esattamente le N righe così definite. Perciò i calcoli da fare sono i seguenti[6]:

$$M = N_\mathrm{t} - 1 = \left\lfloor \frac{x-y}{z_1} \right\rfloor \qquad \text{e poi} \qquad z_2 = \left(\frac{x-y}{M} \right)$$

L'ultima divisione deve essere una normale divisione fratta; è più facile da calcolare con una ordinaria calcolatrice, che non farla determinare a pdftex; in realtà *oggi* questo sarebbe perfettamente in grado di eseguire la divisione fratta senza problemi; oggi, infatti, le versioni moderne dei programmi di composizione del sistema TeX consentono di determinare nativamente il valore di alcune espressioni aritmetiche grazie alle estensioni di etex incorporate già da qualche anno. Si tralascia qui di mostrare come fare per far fare i calcoli al programma stesso, ma si indirizza il lettore alla documentazione di etex contenuta in `.../doc/etex/base/etex-man.pdf`, al paragrafo 3.5 intitolato significativamente *Expressions*; ci si ricordi però, che queste espressioni consentono di trasformare un numero fratto in un numero intero per arrotondamento, non per troncamento, quindi è importante ricordarsene per evitare di stupirsi per risultati diversi da quelli eseguibili a mano.

Nel presente testo lo scartamento è di 12 pt per il corpo normale di 10 pt; il file `bk10.clo`, corrispondente all'opzione *10pt*, determina l'altezza del testo con una prima approssimazione pari all'altezza del foglio diminuita di 3,5 pollici, un valore forfettario che tiene conto dei margini, della testatina e del suo spazio di separazione dal testo, della prima riga di testo, ma non del piedino; poi divide il risultato per l'avanzamento di riga normale conservando la parte intera del risultato; infine rimoltiplica l'avanzamento normale per questo numero intero a alla fine aggiunge `\topskip`. Praticamente quel che si è indicato con il primo procedimento.

Sarebbe desiderabile che le righe di testo corrente che si vedono controluce fra il recto e il verso della pagina siano appoggiate sulle stesse linee di base; in questo modo si riduce l'effetto della semitrasparenza della carta, che provocherebbe un disturbo alla lettura se ciò non avvenisse.

Bisognerebbe quindi che ogni struttura del testo stampato che compare su ogni pagina, dalla semplice riga di testo, ai titolini dei paragrafi e delle loro suddivisioni, dalle formule in display alle inserzioni di figure, tabelle o altro, fossero sempre anche loro dei multipli interi dell'avanzamento di riga normale. Questo è molto difficile da ottenere, anche se si prendono tutte le misure e le precauzioni possibili e immaginabili per avere strutture fatte come si è detto. Nei testi letterari ci si riesce abbastanza bene, ma in testi tecnici è difficilissimo. Anche i compositori che lavorano con programmi di impaginazione professionali trovano serie difficoltà e nella migliore delle ipotesi devono aggiustare a mano

[6] In matematica il troncamento all'intero n di un numero reale x si indica con $n = \lfloor x \rfloor$.

l'altezza di ogni struttura che non sia costituita da semplice testo composto con il font di default nel corpo normale.

In pratica bisogna determinare gli spazi bianchi, i contrografismi, in modo che la loro altezza sommata a quella delle strutture in evidenza sia sempre un multiplo intero dell'avanzamento di riga normale. Questo non è difficilissimo da fare ma richiede una pazienza infinita. Sembra, però che il mark up ConTEXt che usa come interprete il programma luatex riesca a comporre in questo modo senza che l'utente debba intervenire a mano. Chi scrive non ha esperienza in merito, ma il problema costituisce l'argomento di diversi articoli nella rivista ArsTEXnica del GUIT.

Il punto è che bisogna fare i conti con l'algoritmo che i programmi di composizione del sistema TEX usano per suddividere il testo in pagine. Come si può bene capire, l'interprete non lascia il titolo di un paragrafo in fondo ad una facciata per cominciare il testo del paragrafo nella facciata successiva (molti word processor di tipo WYSIWYG lo fanno!); l'interprete cerca anche di evitare le righe orfane e le righe vedove. Una riga orfana è una riga isolata in fondo alla pagina quando il capoverso prosegue nella pagina successiva ("le righe orfane non hanno un passato ma hanno un futuro" è la frase per ricordare di che cosa si tratta); una riga vedova è una riga isolata all'inizio di una pagina il cui capoverso è cominciato nella pagina precedente ("le righe vedove hanno un passato ma non hanno un futuro").

Questo significa che un paragrafo con il suo titolo corrente può cominciare a 4 o 5 righe equivalenti di testo corrente dal fondo di una pagina, perché deve contenere il suo titolo corrente, il suo contrografismo e almeno due righe di testo, altrimenti la sua sola prima riga resterebbe orfana. Analogamente la fine di un capoverso non può avvenire mandando a capo in una nuova pagina la sua ultima riga, perché questa resterebbe vedova. Quando c'è della matematica in display si possono avere altre possibilità di righe orfane o vedove prima o dopo la formula. Quando dunque si devono evitare le righe orfane o vedove si hanno due possibilità: o si compone senza giustificare il piede delle pagine affacciate, oppure si giustificano questi piedi, ma si introduce dello spazio in più nei contrografismi o fra un capoverso e l'altro; in sostanza si disturba la griglia delle righe.

Una soluzione sarebbe quella di accorciare di una riga due pagine affacciate (uno *spread*), in modo che la riga vedova che apparirebbe nella pagina di destra senza l'accorciamento suddetto, verrebbe preceduta dall'ultima riga dalla pagina di sinistra riportata a destra grazie all'accorciamento; nello stesso tempo, scorciando anche la pagina di destra le due griglie dello *spread* restano allineate. Se invece la riga vedova gira nella pagina di sinistra, la prima riga di uno *spread*, accorciando di una riga le due pagine dello spread precedente, la vedova non resta più tale perché verrebbe preceduta da due righe provenienti dallo *spread* precedente. Queste correzioni, come è facile immaginare, vanno fatte all'ultimo, perché qualunque modifica del testo potrebbe distruggere tutta la paziente operazione appena descritta.

Certo, in fase di rifinitura finale, si può intervenire inserendo in punti strategici opportuni comandi di `\newpage`, che eliminano la giustificazione al piede delle pagine, oppure comandi `\enlargethispage` con o senza asterisco che ugualmente possono eliminare la giustificazione al piede delle pagine affacciate. Questo fatto non ha nulla a che vedere con il disegno grafico, ma bisogna sempre ricordarsene per capire perché un disegno accuratamente pensato possa deludere le aspettative.

Un'altra tecnica poco conosciuta per gestire la lunghezza dei capoversi per allungarli o per accorciarli, senza modificarne il testo, consiste nello specificare il comando primitivo `\looseness` alla fine del capoverso insieme ad un valore intero positivo o negativo: per esempio `\looseness1` serve per consentire l'allungamento del capoverso di una riga. Questa tecnica è efficace: (*a*) se il capoverso è sufficientemente lungo; (*b*) se il capoverso è da allungare e il suo ultimo rigo è sufficientemente lungo; (*c*) se il capoverso è da accorciare e il suo ultimo rigo è sufficientemente corto. Se queste condizioni non sono verificate il comando `\looseness` lascia le cose come stanno e quindi non risulta efficace, ma non disturba la composizione. In questa guida l'artificio indicato è stato usato una mezza dozzina di volte.

20.4 I capoversi

Una delle caratteristiche che più distinguono le varie composizioni tipografiche riguarda lo stile di composizione dei capoversi.

In generale sarebbe preferibile che ogni capoverso non fosse separato dal capoverso precedente da una interlinea. Con LaTeX generalmente questa separazione dei capoversi con una interlinea è attuata solamente all'interno delle liste semplici, o delle enumerazioni o delle descrizioni. In questi casi, visto che il testo ha già un margine sinistro rientrato, il tipico rientro del capoverso sarebbe probabilmente eccessivo.

Invece i capoversi del testo 'normale' non sono separati da una interlinea, ma hanno la prima riga rientrata. Questo rientro serve per marcare l'inizio di un capoverso ed è particolarmente utile quando il capoverso precedente ha il suo ultimo rigo che termina praticamente a filo del margine destro. Si capisce subito, allora, che il primo capoverso di un paragrafo non ha bisogno di nessun rientro, perché non è direttamente preceduto da nessun altro capoverso, ma solo dal titolo del paragrafo.

Nella tradizione compositiva francese, invece, anche il primo capoverso di un paragrafo è rientrato. Nell'Ottocento e fino alla prima metà del Novecento anche la tipografia italiana seguiva la tradizione francese, non solo per quel che riguarda il rientro del primo capoverso, ma anche per le spaziature davanti ai segni di interpunzione alti e anche fra i caporali e il testo fra loro contenuto. Se si desidera seguire la tradizione francese di rientrare anche il primo capoverso di un paragrafo basta caricare il pacchetto *indentfirst*.

Il rientro del capoverso dovrebbe essere costante, indipendentemente dal corpo usato, per esempio 10 pt. Se si specifica il rientro mediante una unità

di misura dipendente dal font in uso, per esempio 1 em, questo coincide con 10 pt con il corpo 10, ma aumenta o diminuisce con il corpo del font con cui il capoverso viene composto; questo potrebbe essere un inconveniente estetico perché se si usano spesso incisi composti in corpo minore, i rientri dei capoversi non appaiono più incolonnati. Si consiglia quindi, nell'usare i pacchetti che consentono di modificare il layout della pagina, di definire il rientro mediante una misura assoluta, per esempio lo stesso ammontare dell'avanzamento di riga per il testo normale, da 12 pt a 15 pt.

* * *

NON SEMPRE IL PRIMO CAPOVERSO è preceduto dal titolo del paragrafo. Dipende dal contesto, ma in diversi tipi di documenti a stampa o in certe loro sezioni, come per esempio una introduzione, non si divide il testo in paragrafi, benché vengano trattati diversi argomenti fra di loro non tanto correlati, tanto da dover essere sezionati con i rispettivi titoli. Spesso si ricorre a delle decorazioni.

Nei libri moderni le decorazioni sono spesso assenti; altrettanto spesso si riducono a una breve sequenza di simboli, per esempio tre asterischi centrati, come alla fine del paragrafo precedente (terminato con i tre asterischi) e prima del capoverso precedente (il primo del paragrafo corrente, iniziato con un capolettera), e vengono usati per separare verticalmente parti di testo che non richiedono un sezionamento vero e proprio, ma nello stesso tempo non sono propriamente contigue da un punto di vista concettuale.

Esistono font che contengono sequenze di motivi floreali o motivi geometrici con i quali si possono realizzare decorazioni più eleganti, ma che spesso stonano nei libri moderni.

Se invece si deve riprodurre un libro antico, o anche solo vecchiotto, si può ricorrere a questi font e si possono cercare su CTAN i pacchetti per gestire queste decorazioni nel modo più efficace; tra gli altri si segnala il pacchetto *fourier*.

Al contrario anche in alcuni libri moderni si usano i capilettera; queste sono lettere di corpo decisamente maggiore di quello del testo circostante e vengono usate rialzate o ribassate per marcare l'inizio di un capoverso, o, meglio ancora, il capoverso iniziale di un capitolo e/o di un paragrafo; anticamente si usavano anche capilettera decisamente ornati e di corpo diverse volte maggiore del corpo normale, per esempio in corpo 70 pt quando si compone in corpo 10 pt.

Per usare i capilettera (*versal* in inglese e *lettrine* in francese) si può agevolmente usare il pacchetto *lettrine* con il quale sono disponibili comandi e opzioni di diverso genere per collocare i capilettera nella posizione desiderata scegliendone il font, le dimensioni e gli altri parametri stilistici. Vale la pena di leggerne la documentazione in `.../doc/latex/lettrine/lettrine.pdf`, ma è molto interessante esaminare gli esempi riportati in `.../doc/latex/lettrine/demo.pdf`.

È importante considerare anche l'effetto che si ottiene quando si comincia un capoverso con un capolettera e si prosegue con il maiuscoletto per qualche parola, per poi tornare al font normale; questo è un espediente che viene usato spesso quando si usano i capilettera, per mitigare la transizione fra le grandi

dimensioni della lettera iniziale e il corpo normale; è esattamente quello che si è fatto all'inizio di questo paragrafo.

20.5 Testatine e piedini

Delle testatine e dei piedini si è già commentato abbastanza sul fatto che intervengano o non intervengano nella definizione della gabbia; si è anche detto del pacchetto *fancyhdr* per mettere nella forma dovuta testatine e piedini.

Non si ritorna su questi argomenti, ma ci si sofferma sulla sostanziale differenza che per il sistema TeX esiste fra una testatina e un piedino; non si tratta semplicemente di due blocchetti di testo, generalmente composti di una sola riga ciascuno, messi uno in testa e uno al piede del testo adeguatamente spaziati da questo e, talvolta, adeguatamente separati da semplici decorazioni, in generale formate da un sottile filetto orizzontale.

Il punto è che lo spazio di separazione fra la testatina (eventualmente comprensiva del suo filetto) e il testo sottostante è costituito da spazio vero e proprio di una dimensione ben specificata dal grafico editoriale. Invece lo spazio specificato per distanziare il piedino dal piede del testo rappresenta lo scartamento fra la linea di base dell'ultima linea del testo (o dal fondo del rettangolo ideale che lo contiene) e la linea di base della prima o unica riga del piedino.

In buona sostanza, se si specifica che la distanza `\headsep` fra l'`\header` (testatina) e il testo è di 10 mm, una volta composta una pagina con quella testatina si prende il doppio decimetro, si misura e si rilevano esattamente 10 mm.

Se si specifica che la spaziatura del piedino `\footskip` sia di 10 mm, e, attenzione, non c'è nulla da specificare per l'altezza del piedino, dopo aver stampato una pagina piena con quel piedino si misurano i 10 mm fra le linee di base dell'ultima riga del testo e della prima o unica riga del piedino, si trovano quei 10 mm; ma, considerando i discendenti dell'ultima riga e il fatto che la prima o unica riga del piedino giace sopra la sua linea di base, la distanza effettiva fra il piedino e il testo risulta decisamente minore di 10 mm e questa differenza è otticamente ben percepibile. D'altra parte si è già mostrato lo schema quotato delle due pagine affacciate sinistra e destra nella pagina 166 e nella successiva: si vede chiarissimamente che la freccia che indica la misura di `\footskip` va dalla base del testo alla base del piedino, mentre la freccia che indica la misura di `\headsep` va dalla base della testatina al colmo del testo.

Di questo fatto bisogna tenere ben conto nel progettare la geometria della pagina; certamente i pacchetti impiegati per determinare i parametri geometrici aiutano, ma non si può fare a meno di eseguire alcune prove di stampa per valutare gli effetti; questi ultimi saranno tanto più vistosi quanto più 'ingombrante' è il piedino; un semplice numero di pagina centrato nel piedino ma un po' troppo vicino al testo sovrastante disturba la vista, ma lo si può sopportare; se invece lo stesso piedino contiene a sinistra il nome dell'autore, al centro il numero della pagina e a destra il nome della collana di pubblicazioni a cui appartiene

il rapporto corrente, allora un simile piedino troppo vicino al testo disturba davvero e non lo si può accettare.

Per avere lo schema metrico delle due pagine affacciate lo si riporta di nuovo nella pagina 528 e nella successiva in modo da avere a disposizione più facilmente le varie misure della pagina descritte in questo capitolo. Ci si ricordi che le quote indicate sono relative alla classe *book* con un formato della pagina B5, ma senza avere modificato le dimensioni predefinite. Su fogli della serie ISO la disposizione del testo non è perfetta, ma è accettabile, specialmente se si fa un largo uso di note a margine. Si capisce quindi perché la maggior parte degli utenti ricorre ai pacchetti di personalizzazione, specialmente *geometry* e *typearea*, per disporre di una maggiore flessibilità nel disegno della pagina[7]. Questo è bene, ma non bisogna esagerare: quei pacchetti eseguono al posto nostro i vari calcoli necessari e seguono le nostre indicazioni stilistiche come meglio possono, ma sta al compositore valutare il risultato da un punto di vista estetico.

20.6 I pacchetti di personalizzazione

20.6.1 Il pacchetto *geometry*

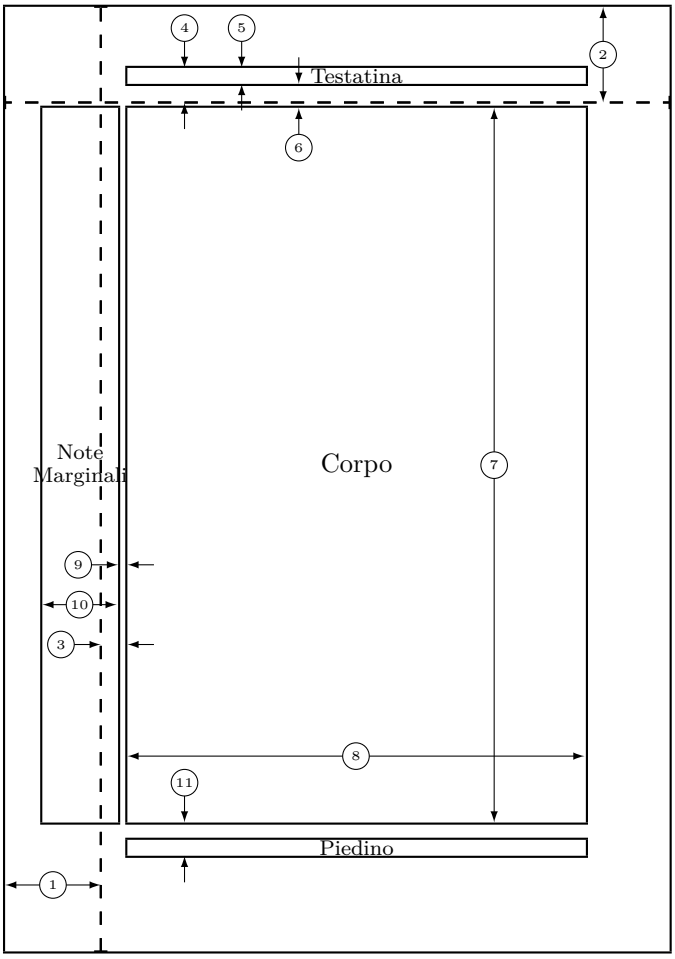
Il pacchetto *geometry* è un pacchetto che contiene tutte le necessarie opzioni e macro per descrivere il layout geometrico di una pagina 'normale', tenendo anche conto dell'ingombro ottico delle testatine e dei piedini; permette di specificare i margini, di scegliere un layout classico, di scegliere le proporzioni adeguate allo stile desiderato e al font usato. Esegue, insomma, tutte le operazioni descritte nei paragrafi precedenti.

Date le molteplici applicazioni, la sua documentazione è piuttosto importante; la si trova in `.../doc/latex/geometry/manual.pdf`.

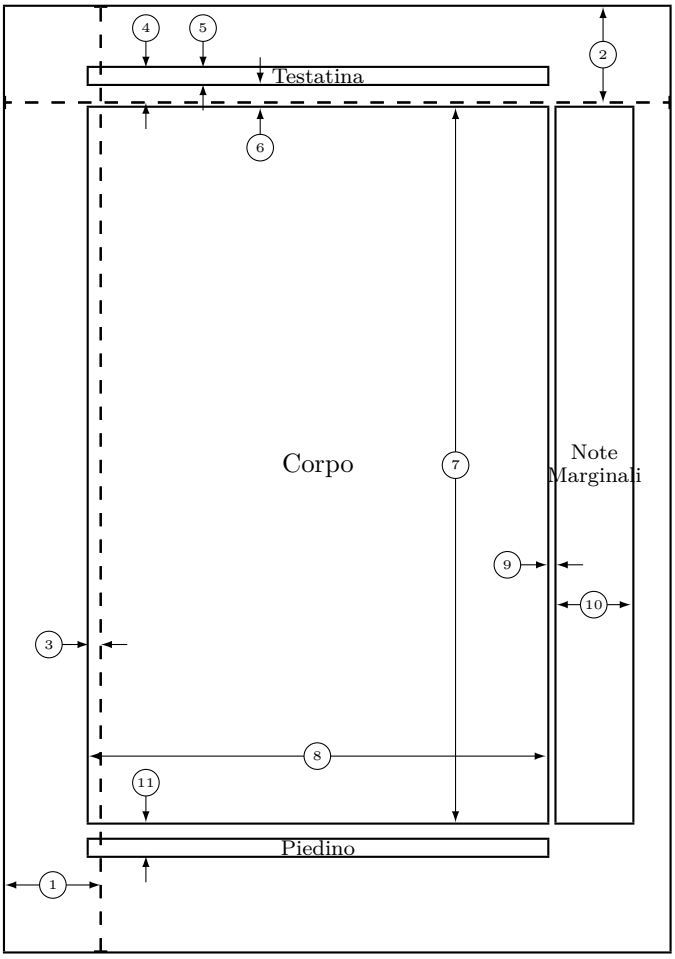
Una delle cose più utili del pacchetto è la sua capacità di determinare le informazioni geometriche mancanti a partire da un certo numero di parametri di default. Per esempio esso usa un rapporto predefinito (che ovviamente può essere modificato dal compositore) fra il margine interno (o quello superiore) e il margine esterno (o quello inferiore) delle pagine; se il documento è stampato solo in bianca, il rapporto di default viene usato solo per i margini superiore e inferiore, mentre per quello interno ed esterno viene usato il rapporto $1:1$, vale a dire che la gabbia viene centrata orizzontalmente fra margini uguali. Se invece il documento viene stampato in bianca e volta il rapporto predefinito o specificato dal compositore viene usato per tutti i margini. Conoscendo le dimensioni della pagina rifinita, il pacchetto è in grado di calcolare la gabbia e i margini a partire da una sola di queste informazioni.

Per esempio, il manuale presenta questo caso: supponiamo di avere una pagina finita in formato A4, quindi di 210 mm per 297 mm, e di specificare il

[7] La classe *memoir* contiene i suoi propri comandi per la creazione della geometria della pagina; essi sono molto flessibili e potenti; *memoir* mette anche a disposizione dei comandi per presentare graficamente la disposizione dei blocchi di testo, i titoli, e altre cose del genere in modo da aiutare il compositore a scegliere meglio i parametri geometrici delle varie pagine.



1	un pollice + \hoffset	2	un pollice + \voffset
3	\evensidemargin = 20pt	4	\topmargin = -26pt
5	\headheight = 12pt	6	\headsep = 18pt
7	\textheight = 538pt	8	\textwidth = 345pt
9	\marginparsep = 7pt	10	\marginparwidth = 57pt
11	\footskip = 25pt		\marginparpush = 5pt (non mostrato)
	\hoffset = 0pt		\voffset = 0pt
	\paperwidth = 500pt		\paperheight = 711pt



1	un pollice + \hoffset	2	un pollice + \voffset
3	\oddsidemargin = -9pt	4	\topmargin = -26pt
5	\headheight = 12pt	6	\headsep = 18pt
7	\textheight = 538pt	8	\textwidth = 345pt
9	\marginparsep = 7pt	10	\marginparwidth = 57pt
11	\footskip = 25pt		\marginparpush = 5pt (non mostrato)
	\hoffset = 0pt		\voffset = 0pt
	\paperwidth = 500pt		\paperheight = 711pt

margine esterno di 24 mm; il rapporto predefinito fra i margini vale 2 : 3 (come nella figura 20.4(b)) e il riempimento predefinito della pagina vale 0,7 (un po' maggiore rispetto al valore di 3/5 che appare nella figura 20.4(b)). Allora:

$$\begin{aligned}
\text{margine interno} &= (2/3) \times 24\,\text{mm} = 16\,\text{mm} \\
\text{giustezza orizzontale} &= 210\,\text{mm} - 24\,\text{mm} - 16\,\text{mm} = 170\,\text{mm} \\
\text{giustezza verticale} &= 0{,}7 \times 297\,\text{mm} = 207{,}9\,\text{mm} \\
\text{margine superiore} &= \frac{1}{1+3/2} \times (297\,\text{mm} - 207{,}9\,\text{mm}) = 35{,}64\,\text{mm} \\
\text{margine inferiore} &= (3/2) \times 35{,}64\,\text{mm} = 53{,}46\,\text{mm}
\end{aligned}$$

Allo stesso modo si sarebbe potuta specificare qualunque altra singola dimensione (oltre alle dimensioni della pagina finita) e si sarebbero potute calcolare le misure mancanti. *geometry* si incarica poi di tradurre i valori di giustezze e di margini in termini di lunghezze interne, quelle usate dall'interprete per eseguire la composizione tipografica.

Non si insiste oltre, perché la funzionalità del pacchetto è talmente vasta che continuando con gli esempi si finirebbe con riscrivere il manuale. Ciò non impedisce di ribadire che il pacchetto *geometry* è uno dei pacchetti preferiti da ogni compositore che voglia personalizzare un pochino i suoi documenti senza conformarsi agli stili predefiniti dalle classi standard della distribuzione del sistema TeX. Come è già stato notato, le funzionalità di *geometry* sono tutte incorporate nella classe *memoir* che le estende ulteriormente; se si usa questa classe, non è necessario caricare il pacchetto *geometry* a meno che non se ne vogliano sfruttare i comandi, invece di affidarsi ai comandi di *memoir* che hanno nomi diversi. Chi scrive ha usato sia l'una che l'altra strada e la sua opinione è che se si usa *memoir* è meglio non caricare né *geometry* né *typearea* (vedi sotto), e usare solo i comandi e le altre funzionalità della classe.

Anche la collezione KOMA-script dispone di un insieme di comandi per specificare la geometria della pagina; anzi, questi comandi sono contenuti nel pacchetto *typearea* che può essere caricato separatamente anche quando si usino classi non appartenenti alla collezione KOMA-script. Vale la pena di documentarsi nella guida `scrguien.pdf` (in inglese), in `.../doc/latex/koma-script`, per vedere che cosa offra il pacchetto *typearea* il quale, di default, usa il metodo delle strisce per la specificazione della geometria della pagina; naturalmente sono disponibili comandi accessori per eseguire delle variazioni sul layout basato sulle strisce; in ogni caso il procedimento di calcolo è molto flessibile.

20.6.2 I titoli e i titolini: i pacchetti *titlesec*, *fancyhdr*, *sectsty* e *tocloft*

Per definire lo stile compositivo dei titoli correnti (titolini) ci sono diversi pacchetti, sempre senza trascurare la classe *memoir* che include in sé la maggior parte della funzionalità dei vari pacchetti presenti nell'installazione completa del sistema TeX.

Uno di quelli maggiormente citati è *titlesec*, il cui manuale a sua volta si premura di avvisare i possibili utenti che se desiderano usare strumenti più semplici possono usare i pacchetti *fancyhdr*, *sectsty* e *tocloft*. Sono tutti pacchetti validissimi, in effetti, ma ognuno è destinato a modificare lo stile solo delle testatine, oppure dei titoli correnti, oppure degli indici[8]. Invece *titlesec* dovrebbe poter agire su tutti i fronti con una interfaccia unificata.

Per quel che riguarda l'aspetto da dare ai titolini, il pacchetto può semplicemente essere invocato con alcune opzioni; queste controllano la famiglia, la serie e la forma dei font; si può stabilire il corpo usato nei titolini come anche la loro composizione giustificata, centrata, o in bandiera. Si può facilmente scegliere come comporre l'etichetta identificativa della sezione sia per quel che contiene sia per il font con cui è scritta.

Se si vogliono fare personalizzazioni più estese il pacchetto offre una moltitudine di comandi per specificare ogni possibile dettaglio relativo a titoli e titolini. Naturalmente queste cose sono applicabili non solo a titoli correnti di ogni sezione ma anche alle testatine e ai piedini.

Caricando il pacchetto *xcolor* si possono usare anche i colori per tutte queste 'iscrizioni' particolari. Naturalmente sta al buon gusto del compositore o del grafico editoriale decidere se usare i colori e sceglierli in modo adeguato anche al tipo di documento che viene composto. Negli anni '90 era abbastanza frequente l'uso di un colore celeste sia per i titolini, sia per le formule, sia per le figure; era un tentativo per evitare le fotocopie, visto che le fotocopiatrici di quell'epoca avevano difficoltà a copiare alcuni colori, fra i quali il celeste. Oggi il celeste può ancora essere usato (magari non così chiaro come allora) ma certamente non per evitare le fotocopie perché oggi le fotocopiatrici copiano perfettamente qualsiasi colore.

I comandi estesi per affrontare personalizzazioni importanti nello stile della pagina servono anche per definire nuovi stili o per modificare quelli già definiti. In particolare è possibile definire variazioni di stile rispetto alla pagine normali per le pagine che contengono solo oggetti flottanti, nelle quali si possono usare altre iscrizioni nelle testatine e/o nei piedini, e si possono eventualmente eliminare i filetti normalmente presenti affinché non possano essere confusi con oggetti appartenenti agli oggetti flottanti.

Si possono gestire anche i 'mark'; i programmi di composizione del sistema TEX lavorano con tre speciali marcatori per definire che cosa deve essere scritto nelle testatine e/o nei piedini. Sono il `\topmark`, `\botmark` e `\firstmark`; essi sono caricati con il ⟨*titolo breve*⟩ dei comandi `\chapter`, `\section` (ed eventualmente `\subsection` con la classe *article*) e consentono di avere sulla pagina corrente o sulle pagine di destra l'informazione del titolo del paragrafo (o del sottoparagrafo) corrente per la pagina nella quale compare l'indicazione; se nel corso della pagina corrente viene iniziato un nuovo paragrafo, a seconda del tipo di documento può essere importante mostrare nella testatina, per esempio, il titolo del paragrafo

[8] È opportuno aggiungere anche il pacchetto *caption* che consente di personalizzare le didascalie degli oggetti flottanti.

che era in vigore all'inizio della pagina oppure quello in vigore alla fine della pagina; tutte queste informazioni vengono gestite diversamente dai programmi di composizione, ma in entrambi i casi esse sono limitate a quei tre marchi; LaTeX li gestisce attraverso entità che si chiamano `\leftmark` e `\rightmark` che vengono 'caricati' mediante l'uso implicito di `\markboth` e `\markright` attraverso i comandi di sezionamento; il processo è abbastanza schermato all'utente normale, che così non deve preoccuparsi di queste cose piuttosto delicate. Con il pacchetto *titlesec* si hanno molte più possibilità perché si dispone di più mark e di più flessibilità nei campi delle testatine e dei piedini. I programmi recenti, che contengono le estensioni di etex, permettono di disporre di molti più mark, con i quali si possono ottenere effetti speciali piuttosto interessanti, che però sono alla portata solo dei programmatori che scrivono le classi o in file di estensione per il linguaggio LaTeX.

Il pacchetto consente anche di comporre indici parziali; per esempio, all'inizio di ogni capitolo si potrebbero stampare l'indice interno del capitolo e anche gli elenchi delle tabelle e/o delle figure contenute nel capitolo; naturalmente questi indici parziali possono essere composti con uno stile diverso dall'indice generale; tuttavia, anche in questo caso, l'uso di questi indici parziali deve essere adeguatamente pensato in funzione del particolare documento che viene composto. Non vale il principio 'italico' che "siccome lo si può fare, allora lo si deve fare"!

Vale quanto viene detto nel paragrafo 8 della documentazione:

> Avendo letto questa documentazione dovrebbe essere chiaro che questo non è un pacchetto per l'utente occasionale a cui piace il layout standard e vuole fare qualche piccola modifica. Questo è invece uno strumento per il tipografo professionale che ha una idea precisa del layout che vuole realizzare ma non ha le conoscenze [informatiche] per farlo. Non si è fatto nessuno sforzo per migliorare il vostro gusto nella composizione dei titoli.[9]

20.6.3 Testatine

La descrizione dello stile delle pagine fatta nel paragrafo precedente è piuttosto sommaria ed è meglio approfondire per poter rendersi conto esattamente come sono composte le testatine negli stili delle pagine `headings` o `myheadings` o in qualunque altro stile che il compositore si voglia definire.

I comandi per definire il contenuto delle testatine di sinistra e di destra sono contenuti nelle macro `\@evenhead` e `\@oddhead`, comandi interni, protetti con il segno `@`. Se il documento è composto solo in bianca, viene usata solo la testatina composta con lo stile delle pagine dispari; se il documento è composto in bianca e

[9] Once you have read the documentation it should be clear that this is not a package for the casual user who likes the standard layout and wants to make simple changes. This is a tool for the serious typographer that has a clear idea of what layout wants and doesn't have the skill to get it. No attempt is made to improve your taste in section formatting.

volta, vengono usate entrambe le macro, in particolare `\@oddhead` per le testatine delle pagine dispari, e `\@evenhead` per le testatine delle pagine pari.

Per la classe *book* queste due macro contengono (semplificato un po') il codice seguente:

```
\def\@oddhead{\textsl{\MakeUppercase{\rightmark}}\hfill\thepage}
\def\@evenhead{\thepage\hfill\textsl{\MakeUppercase{\leftmark}}}
```

Come si vede, e come si constata guardando la pagina corrente, le testatine hanno i titoli correnti composti con un font tondo inclinato e completamente in lettere maiuscole, grazie ai comandi `\MakeUppercase`; il numero della pagina viene composto all'estremità esterna delle testatine in carattere normale. Ma che cosa viene scritto nelle testatine? Per la classe *book*, che compone di default in bianca e volta, nella testatina di sinistra, pagina pari, viene inserito il titolo del capitolo, mentre nella testatina di destra, pagina dispari, viene inserito il titolo del paragrafo. Queste due informazioni sono contenute nelle macro `\rightmark` e `\leftmark`. A loro volta questi marchi sono inseriti quando si usano esplicitamente o implicitamente il comandi `\markboth` o `\markright`. La grande macro che definisce lo stile dalla pagina, per esempio `headings`, non definisce solo il contenuto delle testatine stesse, ma anche che cosa i comandi `\chapter` e `\section` caricano attraverso i comandi `\markboth` e `\markright` rispettivamente, tenendo conto del fatto che capitoli e paragrafi siano numerati e compaiano nel corpo del testo (`\mainmatter`) invece che nelle parti preliminari (`\frontmatter`) o nelle parti finali (`\backmatter`).

Quando si usa un comando `\chapter` nel corpo del testo, il suo titolo (breve) viene implicitamente usato nel comando `\markboth`, a sua volta chiamato dal comando `\chaptermark` che a sua volta viene chiamato da `\chapter`. Quando si usa lo stile di pagina `myheadings` i comandi `\markboth` e `\markright` vengono usati esplicitamente dal compositore. È inoltre possibile definire altri stili di pagina attraverso i pacchetti citati prima oppure attraverso i comandi già presenti dentro alcuni file di classe, come *memoir*; infine possono venire usati i comandi primitivi simili a quelli che vengono usati nel nucleo di LaTeX per definire gli stili `headings` e `myheadings` a cui ci si può riferire per prendere spunto.

Bisogna capire bene come agiscono questi comandi di marcatura. È bene ricordare che LaTeX consiste di moltissime macro che durante la loro esecuzione vengono sviluppate fino ai comandi primitivi. Nella fattispecie i comandi primitivi relativi ai marchi sono `\firstmark`, `\topmark` e `\botmark`. Quando la routine di uscita di LaTeX compone la pagina completa di testatine, piedini, inserti flottanti, eccetera, fa riferimento ai marchi contenuti in `\firstmark` e `\botmark`; come suggerisce il loro nome, il primo rappresenta il marchio che si incontra all'inizio della pagina e il secondo è quello che si incontra alla fine della pagina. Invece `\topmark` è quello che ogni pagina eredita dall'ultimo marchio della pagina precedente, ed è quello che viene usato se la pagina non contiene un altro primo marchio. Se la pagina non contiene nessun nuovo marchio, tutti e tre sono uguali a `\topmark`, il cui valore viene passato per eredità alla pagina successiva e così via. Dopo il primo marchio che LaTeX incontra nel comporre un documento,

in tutte le pagine successive, che siano presenti o assenti nuovi marchi, i tre parametri sono sempre definiti.

Per rendere le cose un poco più flessibili, LaTeX usa i due comandi `\markboth` e `\markright` che inseriscono rispettivamente o due o una sola informazione in ogni marchio che viene messo nel codice da cui la routine di uscita preleverà le informazioni per le testatine. Precisamente il contenuto di ogni marchio è fatto di due informazioni raccolte fra parentesi graffe secondo lo schema seguente:

`{`⟨*marchio di sinistra*⟩`}{`⟨*marchio di destra*⟩`}`

dove le qualificazioni "di sinistra" o "di destra" si riferiscono al contenuto del gruppo di sinistra piuttosto che a quello di destra, ma non si riferiscono, è importante sottolinearlo, né alla pagina di sinistra, né alla pagina di destra. Di solito, dunque, il titolo di un capitolo forma il ⟨*marchio di sinistra*⟩ e il titolo del paragrafo forma il ⟨*marchio di destra*⟩.

Ecco la spiegazione del meccanismo: il comando `\rightmark` preleva dal *primo* marchio nella pagina, `\firstmark` oppure `\topmark`, la componente di destra di quel marchio; invece `\leftmark` preleva la componente di sinistra dell'ultimo marchio presente nella pagina, `\botmark`.

Questo spiega perché, se in una pagina non compare l'inizio di nessun paragrafo o un dato paragrafo inizia a metà pagina, il titolo corrente nella testatina è quello corrispondente al paragrafo che era ancora in vigore quando la pagina è iniziata (marchio ereditato dalle pagine precedenti), non all'eventuale paragrafo iniziato a metà pagina. Se si desiderasse l'indicazione del paragrafo che è in vigore alla fine della pagina, non bisognerebbe prenderlo dal `\firstmark` o dal `\topmark`, ma dal `\botmark`.

Ecco perché se si volesse indicare nella testatina di un dizionario il primo e l'ultimo lemma che compaiono nella pagina bisognerebbe innanzitutto definire il comando per inserire il lemma con, per esempio:

```
\newcommand*{\lemma}[1]{#1\markboth{#1}{#1}
```

e poi inserire il ⟨*lemma*⟩ con:

`\lemma{`⟨ *lemma*⟩`}`

Bisogna definire un nuovo stile di pagina in modo che le testatine di sinistra sia quelle di destra non contengano il numero della pagina, il quale va quindi messo nel piedino, ma contengano i riferimenti al primo lemma in vigore all'inizio della pagina e l'ultimo lemma inserito nella pagina; bisognerebbe cioè che le testatine e i piedini siano definiti nello stile della pagina pressappoco[10] così:

```
\def\@oddhead}{\null\hfill \rightmark\ -- \leftmark}
\def\@evenhead{\rightmark\ -- \leftmark \hfill\null}
\def\@oddfoot{\null\hfill\thepage\hfill\null}
\def\@evenfoot{\null\hfill\thepage\hfill\null}
```

[10] Il "pressappoco" si riferisce al fatto che sono possibili infinite varianti.

Si noti che `\null` rappresenta una scatola vuota contro cui spinge lo spazio elastico `\hfill`; per cui la sequenza costituita dal primo e dall'ultimo lemma compare allineata con il bordo esterno della testatina, mentre i numeri di pagina sono centrati nel piedino.

È anche vero che la classe *book* è una classe generica, non direttamente utilizzabile per comporre un dizionario; sarebbe meglio usare la classe *memoir* che incorpora già molte delle estensioni e dei comandi necessari per eseguire ogni genere di personalizzazione allo stile di composizione di qualunque parte del documento che si intende comporre.

Con la classe *memoir* la personalizzazione per lo stile delle pagine interne di un dizionario si ridurrebbe alle seguenti righe, ognuna autoesplicativa:

```
\makepagestyle{dizionario}
\makeheadrule{dizionario}{\textwidth}{\normalrulethickness}
\makeevenhead{dizionario}{\bfseries\rightmark}{}{\bfseries\leftmark}
\makeoddhead{dizionario}{\bfseries\rightmark}{}{\bfseries\leftmark}
\makeevenfoot{dizionario}{}{\thepage}{}
\makeoddfoot{dizionario}{}{\thepage}{}
```

Si noti il filetto che separa la testatina dal blocco del testo; si noti che testatine e piedini sono fatti di tre parti, quella di sinistra, quella di centro e quella di destra, e a queste parti si riferiscono gli ultimi tre argomenti, alcuni lasciati vuoti, di ciascuna testatina e di ciascun piedino. Infine, in questo stile il primo lemma della pagina è indicato a sinistra e l'ultimo lemma è indicato a destra, entrambi in neretto. Ecco quindi un altro modo di personalizzare lo stile delle pagine di un dizionario.

20.7 La pagina del titolo

Non è un caso che la pagina del frontespizio non sia componibile se non con pochissimi pacchetti disponibili su CTAN; le classi per la composizione di tesi generalmente ne dispongono di un loro ambiente o possono usare il pacchetto di estensione *frontespizio*. L'ambiente *titlepage* della classe *book* offre uno spazio per inserire il comando `\maketitle`, ma per la verità il risultato è molto modesto, come si può vedere dalla pagina 7 di questo documento. Nella prima pagina, senza usare *titlepage* si è cercato di ottenere un frontespizio accattivante da stampare sulla copertina (in brossura) di un esemplare stampato su carta. Anche questo risultato non è esaltante.

Il motivo è semplice ed è chiaramente espresso da Wilson nel manuale della sua classe *memoir*: il titolo/frontespizio va disegnato apposta per ciascun documento, tenendo conto del suo contenuto, del tipo di titolo, delle immagini evocative che possono esservi associate, eccetera. Ovviamente quanto verrà eventualmente stampato sulla prima pagina della copertina è una cosa, mentre è una cosa completamente diversa quello che verrà stampato nella pagina del titolo

(solitamente la terza delle facciate composte e cucite; la prima può contenere l'occhiello, detto *half title* in inglese[11]).

Probabilmente, secondo Wilson, esistono programmi grafici interattivi più adatti per comporre la prima e la quarta di copertina nonché per comporre il frontespizio (e l'occhiello); non è nemmeno da escludere che queste pagine, e quelle che devono contenere le informazioni di carattere legale, debbano o possano essere prodotte separatamente dalle segnature che compongono il documento.

Tuttavia chi scrive ha trovato piuttosto comodo ridefinire un ambiente *titlepage* in modo che dentro ai delimitatori d'ambiente il compositore possa scrivere tutte le informazioni che ritiene necessarie; egli le introduce mediante macro come `\titolo`, `\autore`, eccetera, non dissimili da quelle che si possono usare con i nomi inglesi nel preambolo: `\title`, `\author`, `\date`; sono solo in numero molto maggiore che non quelle previste di default da LaTeX.

Il comando di apertura dell'ambiente praticamente non fa nulla, salvo verificare che si stia lavorando in una nuova pagina; il comando di chiusura fa tutto, nel senso che prende ogni informazione definita con i comandi dati dal compositore, e ogni altra informazione definita di default per collocarle in una griglia di frontespizio abbastanza elastica, ma definita una volta per tutte; esso mette anche nel verso del frontespizio (talvolta impropriamente chiamato 'colophon') tutte le informazioni di carattere legale necessarie; può, se il compositore ha specificato il numero corrente dell'opera, generare e stampare anche il *barcode* calcolando persino la cifra di controllo per il *barcode* a 13 cifre; ovviamente questo è stato fatto per una collana di volumi di una data casa editrice, dove, proprio perché si ha a che fare con una collana, i frontespizi non possono essere molto diversi l'uno dall'altro; anche gli argomenti trattati nei volumi sono sempre di carattere tecnico. Tuttavia la casa editrice, pur mantenendo lo stesso layout delle parti scritte, affida ad un grafico il compito di disegnare la prima e la quarta di copertina sovrapponendo (talvolta ricomponendo) il testo del frontespizio sopra una immagine di fondo, in modo che anche l'immagine sia di aiuto al lettore per capire (o di farsi un'idea) di che cosa venga trattato all'interno del volume.

20.8 Le inserzioni flottanti e non flottanti

20.8.1 Gli oggetti flottanti

Il pacchetto *float*, la cui documentazione si trova in `.../doc/latex/float/float.pdf`, serve per definire nuovi oggetti flottanti oltre alle tabelle e alla figure, e serve anche per dare loro uno stile, un formato adeguato al loro contenuto secondo il gusto del grafico editoriale. Naturalmente serve anche per modificare le definizioni degli oggetti flottanti già definiti.

[11] Comunque la pagina dell'occhiello non deve essere confusa con il risguardo iniziale, che è generalmente una doppia pagina di carta più pesante incollata fra la seconda di copertina e il blocco delle segnature; il risguardo all'inizio del libro e quello alla fine sono sostanzialmente due elementi strutturali usati per collegare la copertina con il blocco delle segnature; questa struttura è particolarmente importante ed è sempre presente nei libri incassati.

Per esempio può servire per specificare la collocazione della didascalia; abbiamo notato più volte che con la classe standard `book` (ma in effetti con tutte le classi standard) la didascalia è sempre collocata sotto l'oggetto flottante, mentre per le tabelle (obbligatoriamente per quelle lunghe che si sviluppano su diverse pagine) in Europa è uso mettere la didascalia prima della tabella. La stessa cosa potrebbe essere necessaria per altri stili di oggetti flottanti.

Questi potrebbero essere dei medaglioni, degli algoritmi, dei programmi, delle raccolte di formule composte in modo particolare, eccetera. È evidente allora che ogni oggetto flottante deve avere una sua numerazione, una sua coda di gestione, eventualmente un suo elenco (come l'elenco delle figure o l'elenco delle tabelle); potrebbe essere riquadrato come un medaglione, potrebbe essere delimitato da filetti; potrebbe avere lo sfondo colorato.

Il pacchetto in questione serve appunto per fare tutte queste cose, ed altre ancora. Va però notato che mentre è relativamente facile, anzi, con questo pacchetto è facilissimo definire nuovi oggetti flottanti, la parte difficile consiste nella loro gestione.

Uno dei problemi apparsi nel forum di GUIT riguardava un tipo di medaglione flottante il cui contenuto potesse svolgersi su più pagine, in modo da non interrompere per tutte le sue pagine la composizione del testo corrente; tipicamente il medaglione avrebbe dovuto avere uno sfondo colorato e ogni suo moncone avrebbe dovuto essere delimitato da filetti e da adeguate intestazioni in testa e in calce ad ogni moncone; nello stesso tempo i vari monconi, avendo deciso che ognuno non avrebbe superato la metà dell'altezza della pagina ordinaria, avrebbero dovuto uscire in pagine assolutamente consecutive, senza essere disturbati dagli oggetti flottanti di altro genere eventualmente ancora in coda.

Oggi esiste una soluzione di Agostino De Marco, membro di GUIT, che ha pubblicato il suo risultato su ArsTEXnica[12]. La soluzione molto parziale che lo scrivente aveva trovato era basata su comandi di bassissimo livello del linguaggio primitivo dei programmi di composizione e sulla programmazione avanzata eseguita mediante il comando `\whiledo` del pacchetto *`ifthen`*; tuttavia essa è evidentemente troppo complicata per essere riportata qui. Si è voluto però esporre questo caso per far capire meglio la problematica relativa agli oggetti flottanti, alla loro composizione, ma soprattutto alla loro gestione.

20.8.2 Gli oggetti non flottanti

Sembra strano che qui si aggiunga un paragrafo sugli oggetti non flottanti, ma talvolta è necessario farne uso, sia pure con estrema moderazione.

Certe figure, fotografie, riproduzioni di stemmi o altri simboli del genere, spesso vengono trattati come delle piccole figure non flottanti, alle quali, se si dovesse apporre una didascalia, sarebbe preferibile usarne una senza numero, in modo da 'essere costretti' a riferirsi ad esse in via indiretta, come per esempio 'la fotografia della pagina 125', oppure la 'fotografia qui a lato'. Questo si rende necessario per

[12] Agostino De Marco e Massimiliano Dominici hanno pubblicato un articolo su ArsTEXnica di ottobre 2008 in merito a questo problema, ma non hanno ancora pubblicato il file di estensione.

evitare che questi oggetti possano essere trattati come delle figure, e quindi gestiti in automatico dalla stessa coda che LaTeX usa per le figure; questa gestione, infatti, non mette mai una figura con un numero d'ordine superiore prima di un'altra figura con un numero d'ordine inferiore; non deve succedere che una piccola foto inserita e contornata dal testo abbia una didascalia con una numerazione più alta di un'altra fotografia di maggiori dimensioni trattenuta in memoria fino a quando LaTeX trova la pagina giusta per aggiungerla ad una pagina in uscita. Contemporaneamente la piccola figura non può essere trattenuta in memoria da una grande figura che ne ritarda la collocazione, rendendone vano l'uso.

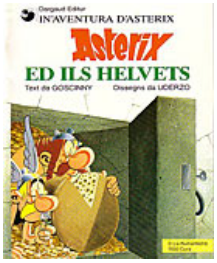
Ecco perché sarebbe opportuno evitare gli oggetti non flottanti, ma se proprio questi oggetti devono venire usati, allora è bene che non siano trattati alla stessa stregua degli oggetti flottanti della stessa natura.

Esiste però il pacchetto *wrapfig* che permette di gestire gli inserti contornati dal testo sia in modo fisso sia in modo più o meno flottante.

Questo pacchetto definisce due ambienti *wrapfigure* e *wraptable* simili nelle prestazioni ai normali ambienti flottanti *figure* e *table*, ma che richiedono due argomenti facoltativi e due obbligatori, il primo dei quali dice con una opportuna lettera di posizionamento se l'oggetto deve flottare o restare fisso; in entrambi i casi l'oggetto può essere collocato nel lato destro o sinistro del testo, oppure, componendo in bianca e volta, dal lato esterno o dal lato interno del testo.

La documentazione si trova in `.../doc/latex/wrapfig/wrapfig-doc.pdf` che contiene anche molti utili suggerimenti in merito alla gestione di questi oggetti e descrive bene le funzionalità del pacchetto nonché le sue limitazioni.

I due parametri facoltativi che l'ambiente accetta permettono di gestire con finezza lo spazio verticale da lasciare per l'inserto e di gestire l'eventuale sporgenza nel margine.



I due ambienti di *wrapfig* hanno nomi diversi così che quando si mette una didascalia, questa è preceduta con il nome giusto e numerata con lo stesso contatore degli oggetti flottanti con lo stesso nome. Il pacchetto è compatibile con il pacchetto *float* per cui *wrapfig* riesce a gestire come inserti anche i nuovi oggetti definiti con `\newfloat`.

In ogni caso uno dei vantaggi di *wrapfig* è che riesce a fare spazio per questo genere di inserti anche se devono essere contornati da diversi capoversi che, singolarmente, non siano sufficientemente lunghi da poter contornare l'oggetto inserito.

La sintassi (descritta per gli inserti di figure; quelli per le tabelle vengono gestiti nello stesso modo) è la seguente:

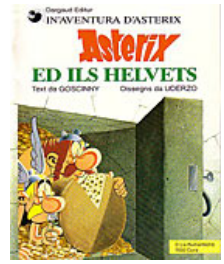
```
\begin{wrapfigure}[⟨numero di righe⟩]{⟨posizionamento⟩}[⟨sporgenza⟩]{⟨larghezza⟩}
⟨comandi per la figura⟩
\end{wrapfigure}
```

Il ⟨*numero di righe*⟩ può essere specificato quando sia necessario regolare con finezza lo spazio verticale da destinare all'inserto. Il codice di ⟨*posizionamento*⟩ è una lettera minuscola o maiuscola corrispondente a `r`, `l`, `i`, `o`; le lettere minuscole specificano che l'inserto deve essere trattato come oggetto non flottante,

mentre al contrario le lettere maiuscole classificano l'inserto come oggetto che può venire spostato. I codici riguardano i margini: `r`, margine destro; `l`, margine sinistro; `i`, margine interno; `o`, margine esterno. Componendo in bianca e volta conviene usare il codice `o` oppure `O`. Il parametro ⟨*sporgenza*⟩ indica di quanto l'inserto può sporgere nel margine; per impostazione predefinita questo valore è nullo. La ⟨*larghezza*⟩ rappresenta la larghezza della scatola dentro la quale il programma inserisce l'inserto; se si specifica un valore nullo, il programma usa la larghezza effettiva dell'inserto tenendo conto dell'eventuale ⟨*sporgenza*⟩. Per l'ammontare della ⟨*sporgenza*⟩ si può fare riferimento alla grandezza `\width` che è la larghezza dell'oggetto corrente da inserire. Infatti per inserire l'inserto della pagina 538 si è usato il codice:

```
\begin{wrapfigure}[10]{O}[.5\width]{0pt}
\includegraphics[height=8\baselineskip]{asterix-romancio}
\label{wfig:asterix}
\end{wrapfigure}
```

Volendo mettere una didascalia non numerata sotto o sopra all'inserto, basta mettere il corrispondente testo fra i ⟨*comandi per la figura*⟩ come nell'esempio a lato. Si tenga presente che per inserire una breve indicazione testuale è necessario specificare una ⟨*larghezza*⟩ non nulla; inoltre conviene specificare una forma di composizione a bandiera, perché la piccola larghezza dell'inserto non consente la composizione a pacchetto giustificata su entrambi i lati. Un inserto come questo a lato, alla fine di un paragrafo, può interferire con il titolo del paragrafo successivo e produrre difetti di composizione.



Copertina di *Asterix ed ils helvets*

20.9 Conclusioni

In questo capitolo sono stati illustrati diversi pacchetti per ottenere 'effetti speciali' nella composizione tipografica con LaTeX. Sta al compositore decidere se e come usare questi strumenti, perché, come è stato detto all'inizio, l'arte tipografica si basa su sottigliezze, non su vistosi effetti speciali.

Si è raccontato in breve come si produce un libro e come lo si disegna; ovviamente queste poche pagine non possono essere considerate nemmeno una breve introduzione alla tipografia, quindi non si raccomanderà mai abbastanza di documentarsi, non solo negli aspetti programmatori di LaTeX, ma specialmente sulla tipografia stessa, senza lasciarsi catturare dalle mode del momento, ma esercitando il proprio giudizio estetico, adeguatamente educato.

Lo scopo è quello di sviluppare una propria competenza nello scegliere le soluzioni compositive 'non acrobatiche', ma che siano effettivamente capaci di trasmettere il messaggio al lettore nel modo più chiaro possibile e senza sottoporlo a stress inutili causati da errori stilistici nella composizione.

Capitolo 21

Dove documentarsi

Invece di inserire un anonimo elenco bibliografico, che d'altronde è presente prima degli indici analitici, qui si cerca di esporre una lista ragionata di testi utili e/o necessari per approfondire la conoscenza di LaTeX e dei suoi fratelli. Le indicazioni precise di ogni riferimento sono riportate nella bibliografia, pagina 781.

21.1 La documentazione essenziale

Buona parte della documentazione relativa a LaTeX si trova già inclusa in qualunque distribuzione completa del sistema TeX.

Tutta la documentazione generale e quella dei singoli pacchetti si può leggere aprendo una "finestra comandi", che prende il nome di "Prompt dei comandi" nelle macchine Windows, e "terminale" o "console" o "xterm" nei sistemi UNIX: Mac e Linux. Aperta una di queste finestre basta dare il comando

`texdoc` ⟨*nome del documento*⟩

e poi premere il tasto Invio o Enter o ↩ . Il ⟨*nome del documento*⟩ è generalmente coincidente con il nome del pacchetto di cui si vuole leggere la documentazione o al nome della guida che si vuole leggere.

Qui elenco solo la documentazione essenziale:

usrguide generica guida iniziale per ogni utente di LaTeX.

fntguide generica guida per la manipolazione dei font con la descrizione dei comandi specifici sia per il modo testo sia per il modo matematico.

clsguide generica guida per la descrizione delle funzionalità delle classi e dei pacchetti, ma anche guida per chi si vuole scrivere delle classi o dei pacchetti personalizzati.

grfguide guida per usare correttamente i comandi per manipolare gli oggetti grafici, sostanzialmente legata all'uso dei pacchetti *graphics*, *graphicx* e *color*.

cnfguide per gestire e usare bene i file di configurazione.

cmfonts in cui la prima parte della guida indica quali siano i font con le codifiche standard della collezione Computer Modern, e poi come siano fatti i i file di descrizione dei font che LaTeX usa per selezionare il font giusto al momento giusto quando si compone un documento mediante quella collezione di font.

cyrguide descrive i font cirillici e i file che consentono di comporre in russo e in ucraino.

encguide descrive le codifiche dei font e come sceglierle quando si usa LaTeX.

inputenc descrive le codifiche di entrata usabili con i diversi sistemi operativi.

makeindx descrive l'estensione per creare convenientemente un file per l'indice analitico e definisce il comando `\printindex`.

modguide espone i problemi che si possono incontrare se si modificano i file della distribuzione del sistema TeX, in particolare quelli relativi all'uso e alle estensioni di LaTeX, e suggerisce come poter fare le modifiche funzionali senza modificare davvero i file.

Queste guide dovrebbero essere le prime ad essere lette e le prime a venire comprese, ma alcune sono difficilotte per chi si trova alle prime armi. Quindi nel seguito verrano elencate anche altre guide più "onnicomprensive", cioè meno legate ad uno specifico tema (le classi, i font, la grafica, eccetera). Ma queste guide devono essere sempre tenute presenti, cioè le si deve saper consultare senza esitazioni; non è che si debbano imparare e memoria, ci mancherebbe altro, ma sarebbe opportuno saperle consultare con il programma texdoc sapendo aprire al primo colpo la guida giusta, perché si conosce già la tipologia di ciò che si desidera controllare.

21.2 Documentazione sulla tipografia

Questa sezione avrebbe l'ambizione di fornire qualche idea sulla tipografia, le consuetudini tipografiche e il book design, senza limitarsi alla consuetudini italiane. Per ovvi motivi questa sezione è abbondantemente incompleta e la scelta dei riferimenti è frutto di un compromesso.

Una breve guida sulle consuetudini della tipografia italiana è contenuta nell'articolo di Gustavo Cevolani "Norme tipografiche", [16].

Questo articolo ha senz'altro il pregio della concisione, una decina di pagine, ma è piuttosto completo e indica anche come le varie consuetudini tipografiche italiane possono essere rese con i comandi e gli ambienti LaTeX.

Benché abbia i suoi anni, il manuale di Roberto Lesina [41] raccoglie in modo ben strutturato un certo numero di consuetudini tipografiche e raccomandazioni di vario genere, che molti scrittori tecnico–scientifici considerano assai valide.

Anche la documentazione accessoria della classe *memoir* [74] contiene non poche pagine di iniziazione alla tipografia e al book design. Anche se il testo è in inglese e si riferisce principalmente alla tipografia inglese, non mancano riferimenti alla tipografia europea continentale di varie nazioni e di vari secoli.

Precedentemente questa documentazione era contenuta nel manuale stesso di *memoir* [73], ma nel 2009 il curatore della documentazione ha ritenuto che il testo fosse diventato troppo lungo e ha separato questo documento dal manuale vero e proprio; viene installato insieme a TEX Live; se non fosse presente nella propria distribuzione bisogna scaricarlo da `http://www.ctan.org/tex-archive/info/memdesign/`.

Il testo elettronico di Peter Flynn [25] è sostanzialmente una guida all'uso di LATEX, ma l'autore è professore di Tipografia all'University College di Cork, in Irlanda. Fu lui il presidente e organizzatore della Conferenza Internazionale di Cork, dove furono gettate le basi per l'encoding esteso T1. Data la professione dell'autore, il testo è particolarmente curato nella sua forma grafica e compositiva; è vero, ci sono delle convenzioni insolite per la tipografia italiana; per esempio i richiami delle note seguono la punteggiatura, mentre nella maggior parte dei libri italiani i richiami di nota precedono la punteggiatura. Tuttavia il tipo di composizione è piuttosto curata, certamente può costituire un esempio da seguire.

L'Imprimerie Nationale francese, l'equivalente del nostro Poligrafico dello Stato, ha pubblicato un interessante libretto [48]. Si tratta di una specie di glossario dove sono elencate in ordine alfabetico le varie parole che comportano certe consuetudini tipografiche e viene specificato come quelle consuetudini siano applicate presso questo autorevolissimo ente.

Tipica della tradizione tipografica francese è la spaziatura uniforme fra le parole indipendentemente da quali segni di interpunzione le separino; questo effetto con LATEX si ottiene con la dichiarazione `\frenchspacing`. Vale la pena di segnalare che questo tipo di spaziatura è quello di default quando viene composta la bibliografia; in questo capoverso si è appunto specificata la spaziatura francese.

Bisogna osservare che le tradizioni tipografiche francesi sotto certi aspetti sono simili a quelle italiane, e sotto certi altri sono molto divergenti: non è solo una questione di lingua, ma si tratta proprio di consuetudini, una volta, molti decenni fa, usate anche in Italia. Per esempio i francesi inseriscono degli spazi prima di tutti i segni di interpunzione 'alti', dai due punti ai punti esclamativo e interrogativo; in questo libretto, anche se composto con cura, a me pare assai sgradevole il fatto che lo spazio che precede questi segni sia piuttosto grande e suscettibile di essere ulteriormente allargato per giustificare le righe; forse la sgradevole sensazione dipende dalla mancanza di abitudine. Questa sensazione mi si ripresenta quando vedo i numeri romani usati come ordinali ma con la desinenza ad apice, come in 'Lille est le siège de la II^{e} région militaire'; in italiano sarebbe certamente scorretto mettere ad apice di un numero romano la desinenza dell'aggettivo ordinale.

Bisogna però ammettere che questo è uno dei pochi manuali di composizione che tratta anche gli aspetti che non vengono mai trattati dagli autori più famosi;

per esempio la composizione della matematica, l'uso delle unità di misura, la scrittura della chimica, anche quella organica con le sue formule di struttura. Anche in matematica la tipografia francese si distingue con particolarità tutte sue, tuttavia le indicazioni fornite in questo libro sono certamente una buona guida anche per gli italiani.

Abbastanza documentato e facile anche per i neofiti è il libro di Michael Mitchel e Susan Wightman [43]. Esso copre tutti gli aspetti della produzione di un libro, e dà indicazioni assai utili a chi desidera fare o fa di professione il book designer, sia pure in un contesto britannico. In realtà, fuori dal Regno Unito cambiano i dettagli, ma i problemi da affrontare per la produzione di un libro sono praticamente gli stessi in ogni nazione. Un dilettante legge volentieri questo manuale, perché non è formato da una serie di prescrizioni, ma è un susseguirsi di descrizioni dei vari aspetti, inclusi quelli tipografici e compositivi, naturalmente, ma è molto istruttivo. Gli autori non fanno un mistero di avere usato un prodotto professionale e commerciale per la *mise en page*, come direbbero i francesi, ma si sforzano di dare indicazioni per non vincolare il lettore allo stesso programma di impaginazione. Molti dei problemi da risolvere a mano con gli impaginatori professionali sono risolti automaticamente con LaTeX, ma ce ne sono altri che con LaTeX sarebbe molto difficile affrontare e risolvere. È utile domandarsi via via: "Questo con LaTeX come potrebbe essere fatto?".

Fra i grandi book designer di questo secolo compare anche Robert Bringhurst; egli ha scritto un piacevole libro [11] dal titolo significativo: *The Elements of Typographic Style*, dove egli descrive i suoi principi tipografici e di book and page design; è incredibile come riesca a legare le proporzioni delle note musicali delle scale diatonica e cromatica con quelle delle figure geometriche e con quelle da usarsi in tipografia. Nelle sue mani quei principi per la scelta delle proporzioni diventano davvero musica, per gli occhi, invece che per le orecchie, ma con quella sobrietà che consente alla persona di gusto di apprezzarne le finezze discrete e sobrie, che però non si impongano fino a distrarre il lettore. Leggendo questo libro e comprendendo appieno i principi tipografici di Bringhurst, si capisce che il book design di qualità non è un gioco da apprendisti.

Gli ultimi riferimenti sono testi in inglese; è un peccato che non esistano o siano difficili da trovare opere simili in italiano[1]; d'altra parte è anche comprensibile che la pubblicazione di opere di questo genere in italiano per lettori italiani non sarebbe remunerativo, mentre lo è certamente se sono pubblicate in inglese per i lettori che usano l'inglese come prima o seconda lingua.

Tra le opere italiane, però, si può citare il manuale di Giorgio Fioravanti [24] dal titolo: *Il manuale del grafico — Guida alla progettazione grafica e all'impaginazione del prodotto editoriale.*

Come dice il titolo, questo libro è più rivolto agli aspetti grafici, piuttosto che a quelli compositivi; penso che si tratti di una buona lettura, ma certamente non ha il respiro dell'opera di Bringhurst o la copertura di argomenti di Mitchel

[1] Il testo di Bringhurst esiste anche in italiano, ma oggi viene stampato come *book on demand*.

e Wightman. Tuttavia merita notare la composizione moderna a gabbia su tre colonne, dove spesso due delle colonne sono unite assieme, così che le pagine sono mosse e attraenti; in questa apparente disomogeneità la struttura grafica si adegua al contenuto, senza però attrarre l'attenzione su di sé distraendo il lettore dalla comprensione del testo.

Si può citare anche il libro di Fabrizio Serra [57]; l'autore, oltre ad essere proprietario e amministratore di un certo numero di aziende tipografiche, è stato anche professore di Composizione della stampa presso la Scuola diretta ai fini speciali in Scienza ed Arti della Stampa, ora divenuto il corso di laurea in Progetto Grafico e Virtuale. Le sue aziende si occupano specialmente, ma non esclusivamente, di stampa di libri, di collane e di periodici nell'ambito delle discipline umanistiche. Il libro di norme che Fabrizio Serra ha pubblicato si riferisce prevalentemente alla stampa di documenti testuali, con numerosi inserti iconografici, tabellari, eccetera, ma in genere senza espressioni matematiche. Il libro contiene una prefazione, una postfazione e una appendice scritte da personaggi importanti nell'ambito della tipografia e del *book design*; basta citare fra tutti Jan Tschichold, famosissimo book designer svizzero.

21.3 Documentazione su LaTeX

La fonte iniziale di ogni sapere a proposito di LaTeX standard è il manuale di Leslie Lamport [39]. L'appendice C di questo testo è stata tradotta e commentata in italiano, [5]; assomiglia al capitolo 29 di questa Introduzione, ma ha una sua autonomia.

Per leggere un testo introduttivo più breve del manuale di Lamport, si può ricorrere al testo [4], inizialmente scritto in tedesco, poi tradotto in molte lingue, fra le quali l'italiano; la traduzione italiana è stata curata da un certo numero di membri del GuIt.

In italiano è disponibile il testo di Lorenzo Pantieri e Tommaso Gordini [54] dal titolo: *L'Arte di scrivere con LaTeX*. È un'ottima guida, e viene aggiornata abbastanza spesso; tipograficamente parlando è molto più bella di questa Introduzione; l'autore, infatti, si è servito del pacchetto di estensione `ClassicThesis`, che modifica le prestazioni delle classi della serie KOMA Script, di cui si è già parlato, producendo un layout assai elegante e composto con font scelti con cura; al di là del contenuto, interessantissimo e ben scritto, anche lo stile tipografico può costituire un buon modello a cui ispirarsi. Invece in questa Introduzione si sono usati la classe standard `book` e i pacchetti standard in modo da mostrare quello che LaTeX può fare da solo. Nel testo di Pantieri e Gordini, invece, è stato cambiato praticamente tutto; il risultato è eccellente, ma per i neofiti potrebbe risultare difficile sfruttare le possibilità di `ClassicThesis` per ottenere le stesse bellissime pagine.

Lorenzo Pantieri ha scritto anche un'altra breve guida, *LaTeX per l'impaziente*, che potrebbe sostituire la guida [4], ripubblicata e aggiornata nel 2019. La guida di Pantieri [52] contiene molti riferimenti ai pacchetti di estensione, in modo da

indirizzare subito il lettore impaziente verso le soluzioni compositive migliori; ovviamente espone anche le parti essenziali di LaTeX, ma non tratta quasi per niente gli argomenti che esulano dalla composizione del testo con il criterio di "quello che vedi è quello che intendi dire", tipico di LaTeX. Il testo è composto con la classe `book` con l'estensione `layaureo`. Il lettore curioso può vedere subito che effetto fa usare questo pacchetto, a confronto con questa Introduzione dove, invece, l'aspetto grafico della pagina è quella di default della classe.

Lorenzo Pantieri ha anche scritto un altro testo intitolato *LaTeXpedia*, [53]. Questa guida raccoglie ciò che Pantieri ha scritto in merito a LaTeX e ai pacchetti con i quali ha lavorato nell'arco di anni. L'impaginazione è identica a quella de *l'Arte*, il contenuto è molto più vasto e comprende anche argomenti molto avanzati.

Invece, per andare su un testo più completo anche della Guida di Lamport ci si può riferire all'eccellente libro di Kopka a Daly [37].

Ma il libro più completo (e pesante, non nel prezzo, ma nel numero di pagine), il libro dei libri su LaTeX, è certamente il testo *The LaTeX companion* [44]. Si tratta di un libro di quasi 1100 pagine dove c'è tutto lo scibile concernente LaTeX 2_ε e i più importanti pacchetti.

21.4 Documentazione sulla grafica

È stata pubblicata anche la seconda edizione del volume *The LaTeX graphics companion* [45]. Questo libro è utilissimo per apprendere come manipolare le immagini per poterle inserire al meglio all'interno di un documento composto con LaTeX. Sono citati e descritti in dettaglio molti dei programmi appena menzionati in questo testo, ma, quello che più importa, essi sono commentati e confrontati per esplorarne pregi e difetti o per indicare usi alternativi a quelli che ci si potrebbe aspettare[2].

Per quel che riguarda la creazione di grafica che rappresenta informazione quantitativa, si indica il libro di Edward Tufte, *The visual display of quantitative information*. Oltre ad essere un bellissimo libro in sé, esso mostra una quantità di rappresentazioni grafiche belle e brutte, tutte commentate; per quelle brutte i commenti indicano proprio perché sono mal riuscite. C'è molto da imparare, anche per chi si crede un abile creatore di diagrammi, cartogrammi, istogrammi, eccetera. Ciò non toglie che non si possano tenere in conto anche le raccomandazioni della norma UNI 2949, [64], che, guarda caso, sono coerenti con la maggior parte delle indicazioni di Tufte, anche se hanno un aspetto più tecnico dei bei diagrammi mostrati da Tufte nel suo libro.

[2] Tutti i libri della Addison Wesley sono acquistabili tramite il sito dell'associazione mondiale degli utenti di TeX con notevoli sconti; si suggerisce di esplorare il sito `http://www.tug.org/books/`.

21.5 Documentazione sui singoli pacchetti

La documentazione sui singoli pacchetti di estensione fa generalmente parte della distribuzione stessa del pacchetto; ogni volta che si scarica dalla rete un pacchetto e questo viene installato nella struttura standard di cartelle del sistema TEX, la documentazione va a finire in una cartella, che si chiama come il pacchetto, nel ramo dell'albero che comincia con `.../doc`; può poi finire nel ramo `latex` oppure `tex` oppure `generic` a seconda che l'uso del pacchetto sia specifico di LATEX, di TEX, oppure sia polivalente.

Queste cartelle sono le prime a dover essere esaminate quando si cerca della documentazione specifica; non lo si raccomanda mai abbastanza, tanto che molti utenti si rivolgono in rete, alla prima mailing list che capita, per avere delucidazioni su cose che sono spiegate chiarissimamente nella documentazione che arriva insieme al sistema TEX e che quindi è già installata nello stesso calcolatore dove l'utente si è trovato di fronte ad un problema. Per carità un po' di aiuto non lo si nega a nessuno, ma quando si chiede aiuto ad una mailing list competente, la 'netiquette' impone prima di tutto di aver fatto il possibile per documentarsi autonomamente.

Per i problemi relativi al sistema TEX è certamente competente il forum `comp.text.tex` in `groups.google.com`.

Anche il Gruppo degli utilizzatori Italiani di TEX ha un forum all'indirizzo `https://www.guitex.org/forum/` ed è meglio rivolgersi a questi gruppi, invece di scrivere alla mailing list dello shell editor che si sta usando, dimostrando quindi di non aver capito la distinzione fra lo shell editor e il sistema TEX.

Ma una cartella che bisogna avere sempre presente è `.../doc/latex/base/` perché quella cartella contiene alcune rapide guide dal titolo *xxx*`guide.pdf`, dove le tre lettere *xxx* indicano un po' stenograficamente di che cosa si tratti: *usr* riguarda il generico utente, *cls* riguarda le classi, eccetera; anche se sono abbreviazioni inglesi non è difficile decifrarle. Quelle guide sono già state elencate nel preambolo di questo capitolo.

Un problema per gli utenti italiani è costituito dal fatto che quasi tutta la documentazione è scritta in inglese; capisco che non tutti sono tenuti a conoscere l'inglese come la propria lingua madre, tuttavia se si usa un PC o un laptop, se si naviga in rete, una conoscenza di base dell'inglese è sicuramente presente; questa conoscenza di base dell'inglese scritto dovrebbe essere sufficiente.

21.6 Documentazione su TEX

Se si arriva a scrivere dei file di classe o dei pacchetti di estensione viene voglia di conoscere meglio il linguaggio TEX nativo; il libro dei libri è il TEXbook [35], scritto direttamente dal padre del sistema TEX. Questo libro è la fonte più autorevole su TEX; se si è in vena di spese, esiste anche l'*opera omnia* di Knuth relativa alla tipografia digitale pubblicata come "Millenium Edition" [36]. Essa contiene, in un elegante cofanetto cartonato, tutti e cinque i volumi del sistema

TEX: Volume A, *The TEXbook*; Volume B, *TEX: The Program*; Volume C, *The* METAFONT*book*; Volume D, METAFONT*: The Program*; Volume E, *Computer Modern Typefaces*. Tutti e cinque i volumi sono incassati (copertina rigida) e, a seconda dell'uso che se ne fa, sono molto meglio che non procurarsi separatamente le varie parti, per giunta brossurate o con legature a spirale.

Conviene visitare il sito di Enrico Gregorio: `http://profs.sci.univr.it/~gregorio/egtex.html`. Ci sono diversi documenti da scaricare, da quello sugli Orrori, a quello sugli Esempi. Quello che sembra più interessante, oltre alle altre Guide sui pacchetti più comuni, è il testo PDF *Appunti di programmazione in TEX e LATEX*; si tratta della seconda edizione aggiornata nel 2009 e contiene diverse note che descrivono molte macro sia scritte in linguaggio TEX, sia in linguaggio LATEX 2_ε. Si tratta di cose molto istruttive, alcune semplici, altre decisamente più complesse; ma come testo didattico, secondo chi scrive, è un ottimo esempio che consente all'apprendista di arrivare per gradi e secondo le sue necessità a livelli alti di programmazione, ottimi per scrivere macro complesse, così come per scrivere file di classe o di estensione. Non si dimentichino i molti articoli che Enrico Gregorio ha scritto epr ArsTEXnica; sono tutti scaricabili dalla sezione ArsTeXnica del forum del GUIT: `www.guitex.org`.

21.7 Documentazione sui simboli di LATEX

LATEX consente di inserire nei documenti una miriade di simboli che sono disponibili con certi pacchetti oppure con certi font; esiste un documento in rete, predisposto da Scott Pakin e tenuto costantemente aggiornato, che contiene tutti i simboli gestibili con LATEX, [49]. L'indirizzo in rete rimane sempre lo stesso anche dopo ogni aggiornamento.

21.8 Documentazione sulla composizione della matematica

In una lettera privata a un suo corrispondente Jean-Pierre Serre scrisse:

> Mi colpisce il fatto che la scrittura matematica sia simile alla scrittura di una lingua. Per essere capito devi seguire certe regole grammaticali. Tuttavia, nel nostro caso, nessuno si è preso la briga di scrivere una grammatica; noi l'afferriamo come un bimbo lo fa dai suoi genitori, per imitazione degli altri. Alcuni matematici hanno un buon orecchio; altri no (e alcuni preferiscono usare espressioni gergali come, per esempio, "iff"). Così è la vita.[3]

[3]It strikes me that mathematical writing is similar to using a language. To be understood you have to follow some grammatical rules. However, in our case, nobody has taken the trouble of writing down the grammar; we get it as a baby does from its parents, by imitation of others. Some mathematicians have a good ear; some not (and some prefer the slangy expressions such as "iff"). That's life.

Tuttavia non è proprio completamente esatta questa affermazione; alcune regole esistono, almeno per certi stili di scrittura matematica.

Le norme ISO per la composizione della matematica scritta per i fisici e i cultori delle scienze sperimentali e tecnologiche (la matematica delle grandezze) si trovano tradotte in italiano nel documento [23] pubblicato dall'Ente di Unificazione Italiano. Fra le guide tematiche del GUIT, compare anche *Regole e consigli per comporre la matematica delle scienze sperimentali*, [7]. Si ritiene che sia il minimo necessario per comporre bene la matematica delle scienze che usano grandezze. Alcune nozioni sono utili anche per chi scrive di matematica pura, ma essenzialmente il testo è rivolto ai fisici, agli ingegneri, ai chimici, e a tutti coloro che scrivono matematica delle grandezze.

Vale la pena di consultare anche le indicazioni fornite dalla Unione Internazionale di Fisica Pura e Applicata riportate per esempio nell'appendice F del testo [72].

Molto interessante e completo è il documento elettronico [62] pubblicato dal National Institute for Standardization and Technology; esso è estremamente dettagliato e vale la pena di disporne e di ricorrevi spessissimo; è anche molto aggiornato sul sistema SI; ovviamente si rivolge ai lettori statunitensi, quindi dove discute del separatore decimale indica il punto invece della virgola; usa anche l'ortografia americana invece di quella inglese per rispettare usi e costumi consolidati negli USA, per giunta fissati per legge da altre norme. La bibliografia è molto estesa; fra le appendici ci sono elenchi di fattori di conversione da unità "strane" a unità SI che risultano difficilissimi da reperire altrove; alcune di queste unità "strane", sono tanto inconsuete che se non fossero elencate in quell'appendice qui in Europa non si sarebbero mai potute incontrare. Le norme in questione, complete di check list per gli autori, sono destinate ai vari ricercatori di quell'Istituto affinché scrivano i loro testi tecnici in modo irreprensibile. Fatte le debite piccolissime tare (come il punto decimale e il plurale dei nomi che indicano le unità di misura; in italiano quei nomi sono quasi tutti invarianti al plurale) le indicazioni presenti in questa guida sono validissime per qualunque scrittore tecnico.

L'American Mathematical Society ha anche pubblicato due libretti su questo argomento; il primo [60] serve per dare adeguate istruzioni agli autori che le inviano lavori matematici da pubblicare; il secondo [38], invece, oltre a dedicare un adeguato spazio alla composizione della matematica con il calcolatore, serve anche per consigliare il lettore come scrivere di matematica, cioè come usare la prosa che può interessare un matematico.

Come appare chiaro dal titolo, i primi riferimenti si riferiscono esplicitamente alla matematica scritta dai fisici e dagli scienziati sperimentali, mentre i secondi si riferiscono alla matematica dei matematici. Come è stato giustamente osservato, sarebbe opportuno che fisici, matematici, ingegneri e scienziati sperimentali scrivessero la matematica nello stesso modo: non c'è dubbio. Ma non conoscendo se il lettore sia un matematico, o un fisico, o un ingegnere, o un economista, o un giurista, eccetera, allora...

In ogni caso si rinvia al capitolo 24 una lista di simboli e di notazioni tratta dalle norme o dalle pubblicazioni suddette.

Può essere utile il manualetto preparato da Michael Downes [20], della American Mathematical Society, che ripercorre un po' quanto si è già detto a proposito delle estensioni fornite dal pacchetto `amsmath`, ma scende in maggiori dettagli e fornisce utilissimi commenti e indicazioni di ulteriori pacchetti di estensione della matematica che qui non sono stati nemmeno nominati.

Unendo "l'utile al dilettevole" si segnala il pacchetto `cool` [58]; esso propone all'utente un certo numero di estensioni che consentono composizioni matematiche ben fatte e ben documentate.

Esso può essere utilissimo per comporre la matematica sia con lo stile dei matematici, sia con quello dei fisici e ingegneri; esso consente di usare una moltitudine di comandi ognuno dei quali realizza la corretta scrittura di ogni possibile piccola o grande operazione matematica, con la garanzia che essa sarà tradotta nel file di uscita in modo corretto e conforme alle norme; ma esso consente anche una maggiore leggibilità del file sorgente in quanto i nomi degli operatori o delle operazioni sono scritti per disteso o comunque sono poco abbreviati; il pacchetto contiene definizioni anche per gli operatori più specializzati e che non sono listati nel capitolo 24. Se ne consiglia l'uso a tutti gli utenti di LaTeX che abbiano necessità di comporre testi con una notevole dose di matematica, in particolare se questa è molto avanzata.

Si consiglia vivamente la lettura del documento di Herbert Voß, *Math mode*, [71], che espone tutti i modi possibili per comporre la matematica sia con plain TeX, sia con il mark-up LaTeX, senza o con le estensioni fornite principalmente dal pacchetto `amsmath`, e pacchetti annessi, oltre che con un'altra ventina di pacchetti specializzati per la composizione della matematica. Esamina i problemi di composizione per categorie e spessissimo mostra degli esempi, completi di codice, anche per realizzare strutture compositive matematiche per situazioni particolari. Sono 135 pagine densissime di informazioni preziose e costituiscono una vera miniera per trovare soluzioni o raccogliere idee nuove.

Per quel che riguarda la *vexata quaestio* della punteggiatura esterna delle formule in display, oltre a richiamare quanto annotato verso la fine del capitolo 14 si suggerisce qui l'esame dei fascicoletti di istruzioni per gli autori messi a disposizione su CTAN e nelle apposite cartelle delle varie distribuzioni del sistema TeX. Si ricorda il già citato fascicoletto della Elsevier [22] e l'analogo fascicoletto predisposto dalla Kluwer [34]. Queste case editrici specializzate nelle pubblicazioni di tipo accademico e scientifico non usano la punteggiatura esterna per le espressioni matematiche in display.

La bibliografia di questo testo contiene diversi riferimenti a libri o manuali che trattano di punteggiatura; sono tutti scritti da linguisti quindi non si occupano della punteggiatura in matematica. Tuttavia c'è il testo [56] che, secondo chi scrive, ha una visione più ampia degli altri sul significato di "segno di punteggiatura"; infatti oltre ai soliti segni convenzionalmente considerati per punteggiare un testo, ci sono altri segni, compresi gli spazi orizzontali e verticali che possono essere usati per una marcatura logico-sintattica del testo. Si tratta di un punto

di vista abbastanza insolito, ma utile per comprendere e valutare meglio anche la questione della punteggiatura esterna e, parzialmente, di quella interna, realizzata mediante spazi nel campo della matematica.

21.9 L'archiviazione dei documenti

L'argomento relativo all'archiviazione è ancora abbastanza nuovo e c'è poca documentazione disponibile. L'unico testo che sembra essere stato pubblicato è in vendita solo accedendovi dalla rete, [21].

In compenso nel sito `http://www.pdfa.org/doku.php?id=start:en` si trova una quantità di informazioni incredibile, che vanno dai documenti tecnici, alle *FAQ*, le *Domande frequenti*, dove si trovano una quantità di informazioni spicciole su argomenti precisi e delimitati.

Il sito viene continuamente aggiornato e vale la pena di accedervi regolarmente, specialmente se la produzione di documenti archiviabili è una parte importante del proprio lavoro.

Bibliografia

[1] LaTeX3 Project Team. "LaTeX 2_ε font selection". PDF document, 11 2020. Leggibile con `texdoc fntguide`.

[2] Artech House. "Style Manual". Artech House, Boston – London.

[3] A.V. *The Chicago Manual of Style*. The University of Chicago Press, 14 ed., 1994.

[4] A.V. "Una mica tanto breve introduzione a LaTeX 2_ε". PDF document, 10 2019. In `$TEXMF/doc/latex/lshort-italian/itlshort.pdf`. Traduzione a cura di C. Beccari, G. Pignalberi, M. Sacchetto, T. Gordini, G. Ruocco, G. Milanese.

[5] Claudio Beccari. *Il LaTeX Refernece Manual commentato*. GuIt, `http://www.guitex.org/home/images/doc/GuideGuIT/`, 2017.

[6] Claudio Beccari. "La classe TOPtesi". PDF document, 11 2017. In `$TEXMF/doc/latex/toptesi/toptesi.pdf`, `$TEXMF/doc/latex/toptesi/toptesi-it-xetex.pdf`.

[7] Claudio Beccari. "Regole e consigli per comporre la matematica delle scienze sperimentali". PDF document, 2019. Scaricabile da `http://www.guitex.org/home/images/doc/GuideGuIT/ComporreMatematica.pdf`.

[8] Gian Luigi Beccaria. *Il mare in un imbuto*. Giulio Einaudi editore, Torino, 2010.

[9] Francesco Biccari. "Documentation of the LaTeX class sapthesis.cls". PDF document, 3 2012. In `$TEXMF/doc/latex/sapthesis/sapthesis-doc.pdf`.

[10] Francesco Biccari. "Documentation of the LaTeX class unifith.cls". PDF document, 8 2019. Leggibile con `texdoc unifith`.

[11] Robert Bringhurst. *The elements of typographic style*. Hartley & Marks, Vancouver, BC, 2004.

[12] Christophe Caignaert. "A story of *kpfonts*". In: *TGGboat*, vol. 31, no. 3, (2010), pp. 161–174.

[13] CEI 24-1. *Unità di misura e simboli letterali da usare in elettrotecnica*. Comitato Elettrotecnico Italiano, Milano, 1986.

[14] CEI-S-646. *Dizionario della strumentazione nucleare. Primo complemento al fascicolo S-447.* Comitato Elettrotecnico Italiano, Milano, 1983.

[15] Gustavo Cevolani. "Libretti in LaTeX". In: *ArsTEXnica*, vol. 1, no. 2.

[16] Gustavo Cevolani. "Norme tipografiche". In: *ArsTEXnica*, vol. 1, no. 1.

[17] CNR-UNI 10003. *Sistema internazionale di unità (SI).* Ente Italiano di Unificazione, Milano, 1984.

[18] Massimiliano Dominici. "Utilizzo di caratteri TrueType con LaTeX – Un esempio pratico: i *FellTypes*". In: *ArsTEXnica*, vol. 2, no. 4, (2007), pp. 88–102.

[19] Massimiliano Dominici. "Una panoramica su Pandoc". In: *ArsTEXnica*, vol. 8, no. 15, (2013), pp. 31–38.

[20] Michael Downes. *Short math guide.* American Mathematical Society, Providence, Rhode Island, 1.09 ed., 2002. In `ftp://ftp.ams.org/pub/tex/doc/amsmath/short-math-guide.pdf`.

[21] Olaf Drümmer, Alexandra Oettler, Dietrich von Seggern. *PDF/A in a Nutshell. Long Term Archiving with PDF.* Callas Software gmbh, 2008. In: `http://www.pdfa.org/doku.php?id=pdfa:en:pdfa_in_a_nutshell`.

[22] Elsevier. "Preparing articles with LaTeX — Instructions to authors for preparing compuscripts". PDF document. In `instraut.pdf`.

[23] Ente Italiano di Unificazione. *Grandezze e unità di misura per la fiscia e le scienze sperimentali.* Unificazione Italiana, Milano, 2020. Le norme ISO 80000 sono divise in una quindicina di fascicoli aggiornati in anni diversi, dal 2013 al 2021. Sono acquistabili anche in rete dal sito `store.uni.com/catalogo`.

[24] Giorgio Fioravanti. *Il manuale del grafico — Guida alla progettazione grafica e all'impaginazione del prodotto editoriale.* Zanichelli, Bologna, 1987.

[25] Peter Flynn. *Formatting information — A beginner's introduction to typesetting with LaTeX.* In `CTAN/tex-archive/info/beginlatex/beginlatex-3.6.pdf`.

[26] Michel Goossens. *The XeTeX Companion – TeX meets OpenType and Unicode.* LaTeX Team, gennaio 2011. In `http://xml.web.cern.ch/XML/lgc2/xetexmain.pdf`.

[27] Enrico Gregorio. "Installare TeX Live 2010 su Ubuntu". In: *ArsTeXnica*, vol. ottobre, no. 10, (2010), pp. 7–13.

[28] Enrico Gregorio. "L'arte esoterica di scrivere in cirillico con LaTeX". In: *ArsTeXnica*, vol. aprile, no. 9, (2010), pp. 57–73.

[29] Enrico Gregorio. "Come comporre un frontespizio e vivere felici". PDF document, 9

2011. In `$TEXMF/doc/latex/frontespizio/frontespizio.pdf`.

[30] Enrico Gregorio. "Installing TeX Live 2010 on Ubuntu". In: *TUGboat*, vol. 32, no. 1, (2011), pp. 56–61.

[31] Enrico Gregorio. *Introduzione a XeLaTeX*, gennaio 2011. In `http://profs.sci.univr.it/~gregorio/introxelatex.pdf`.

[32] ISO 690. *Information and documentation – Guidelines for bibliographic references and citations to information resources*. International Organization for Standardization, Ginevra, 2010.

[33] ISO Handbook 2. *Units of measurement*. International Organization for Standardization, Ginevra, 1982.

[34] Kluver. "User manual for `kluwer.cls`". In `$TEXMF/doc/latex/kluwer/usrman.dvi`.

[35] Donald E. Knuth. *The TeXbook*. Addison Wesley, Reading, Mass., 16 ed., 1996.

[36] Donald E. Knuth. *Computers & typesetting*. Addison Wesley, Reading, Mass., 2001. Millenium edition.

[37] Helmut Kopka, Patrick W. Daly. *Guide to LaTeX*. Addison Wesley, Reading, Mass., 4 ed., 2004.

[38] Steven G. Krantz. *A primer in mathematical writing*. American Mathematical Society, Providence, Rhode Island, 2 ed., 1998.

[39] Leslie Lamport. *A document preparation system — LaTeX — User's guide and reference manual*. Addison Wesley, Reading, Mass., 2 ed., 1994.

[40] Jerónimo Leal, Gianluca Pignalberi. *Edizioni Critiche – Guida alla composizione con il proprio computer*. TeXnologie. Edizioni CompoMat, Configni (RI), 2012.

[41] Roberto Lesina. *Il nuovo manuale di stile*. Zanichelli, Bologna, 2 ed., 1994.

[42] André Miede. "A classic thesis style". PDF document, 12 2011. In `$TEXMF/doc/latex/classicthesis/classicthesis.pdf`.

[43] Michael Mitchel, Susan Wightman. *Book typography — A designer's manual*. Libanus Press, Marlborough, Wiltshire UK, 2005.

[44] Frank Mittelbach, Michel Goosens, *et al. The LaTeX companion*. Addison Wesley, Reading, Mass., 2 ed., 2004.

[45] Frank Mittelbach, Michel Goosens, *et al. The LaTeX graphics companion*. Addison Wesley, Reading, Mass., 2 ed., 2007.

[46] Bice Mortara Garavelli. *Prontuario di punteggiatura*. Editori Laterza, Bari, 11 ed., 2008.

[47] Bice Mortara Garavelli (cur.). *Storia della punteggiatura in Europa*. Editori Laterza, Bari, 2008.

[48] Imprimerie Nationale (cur.). *Règles typogrphiques*. Imprimerie Nationale, Parigi, 2006. ISBN 2-7433-0482-0.

[49] Scott Pakin. "The comprehensive LaTeX symbol list". PDF document. In `$TEXMF/doc/latex/comprehensive/symbols-a4.pdf`.

[50] Lorenzo Pantieri. "Tesi Classica". PDF document, 5 2012. In `http://www.lorenzopantieri.net/LaTeX_files/TesiClassica.zip`.

[51] Lorenzo Pantieri. "Tesi Moderna". PDF document, 5 2012. In `http://www.lorenzopantieri.net/LaTeX_files/TesiModerna.zip`.

[52] Lorenzo Pantieri. "LaTeX per l'impaziente – Un'introduzione all'Arte di scrivere con LaTeX". PDF document, 2017. In `http://www.lorenzopantieri.net/LaTeX_files/LaTeXimpaziente.pdf`.

[53] Lorenzo pantieri. "LaTeXpedia". PDF document, 2019. Scaricabile da `http://www.lorenzopantieri.net/LaTeX_files/LaTeXpedia.pdf`.

[54] Lorenzo Pantieri, Tommaso Gordini. "L'arte di scrivere con LaTeX". PDF document, 2017. In `http://www.lorenzopantieri.net/LaTeX_files/ArteLaTeX.pdf`.

[55] Will Robertson. "The XeTeX reference guide". PDF document, gennaio 2011. In `$TEXMF/doc/xetex/xetexref/XeTeX-reference.pdf`.

[56] Francesca Serafini. *Questo è il punto – Istruzioni per l'uso della punteggiatura*. Editori Laterza, Bari, 2012.

[57] Fabrizio Serra. *Regole editoriali, tipografiche & redazionali*. Istituti Editoriali e Poligrafici Internazionali, Pisa – Roma, 2004. Prefazione di Martino Mardersteig, postfazione di Alessandro Olschki, appendice di Jan Tschichold.

[58] N. Setzer. "The cool package". PDF document. In `$TEXMF/doc/latex/cool/cool.pdf`.

[59] Societa Italiana di Fisica e Il Nuovo Cimento, Bologna. *Guida per gli autori*, 1988.

[60] Ellen Swanson, Arlene O'Sean, Antoinette Schleyer. *Mathematics into type*. American Mathematical Society, Providence, Rhode Island, 2 ed., 1999.

[61] Till Tantau. "Ti*k*Z and pgf — Manual for version 1.10". PDF document. In `texlive/2012/texmf-dist/doc/generic/pdf/version-for-pdftex/en/pgfmanual.pdf`.

[62] A. Thompson, B. N. Taylor. "The NIST Guide for the use of the International System of Units", 2008. In `http://physics.nist.gov/cuu/pdf/sp811.pdf`.

[63] Edward R. Tufte. *The visual display of quantitative information*. Grapphics Press, Cheshire, Connecticut, 2 ed., 2007.

[64] UNI 2949. *Diagrammi e cartogrammi, Regole generali per l'elaborazione*. Ente Italiano di Unificazione, Milano, 1982.

[65] UNI 6015. *Segnaccento obbligatorio nell'ortografia della lingua italiana*. Ente Italiano di Unificazione, Milano, 1967.

[66] UNI 6461. *Divisione delle parole in fin di linea*. Ente Italiano di Unificazione, Milano, 1969.

[67] UNI 7090. *Metodo di scrittura numerica delle date*. Ente Italiano di Unificazione, Milano, 1973.

[68] UNI-ISO 5966. *Presentazione dei rapporti scientifici e tecnici*. Ente Italiano di Unificazione, Milano, 1989.

[69] Ivan Valbusa. "User's guide to suftesi". PDF document, 2015. In `$TEXMF/doc/latex/suftesi/suftesi.pdf`.

[70] Filippo Vomiero. *Utilizzare i software per la gestione della bibliografia con LaTeX*. GuIt, 7 2109. Scaricabile da `https://www.guitex.org/`.

[71] Herbert Voss. "Math mode". PDF document, 2010. In `$TEXMF/doc/latex/mathmode/`.

[72] R.C. Weast, M.J. Astle, W.H. Beyer (cur.). *CRC handbook of chemistry and physics*, CRC Press, Boca Raton, Florida, cap. "Symbols, units and nomenclature in physics", pp. F259–F293. 65 ed., 1984.

[73] Peter Wilson. "The memoir class for configurable typesetting — User guide". PDF document, 2004. In `\protect\T1\textdollarTEXMF/doc/latex/memoir/memman.pdf`.

[74] Peter Wilson. *A few notes on book design*. The Herries Press, Normandy Park, WA, 2009.

Indice analitico

Simboli

\!, 333, 727
\", 708
">, 703
\', 708, 755
', 703
\(, 311, 331, 724, 728, 742
\), 311, 331, 724, 728, 742
\+, 755
\,, 341, 703, 727, 790
\-, 67, 382, 457, 630, 632, 637, 650, 755
\., 708
\:, 338, 727, 790
\;, 727, 790
\<, 755
\=, 708, 754, 755
\>, 755, 790
@, 278
\@, 703, 704
\@dottedtocline, 469, 478
\@evenhead, 532
\@firstoftwo, 462
\@glossaryfile, 292
\@gobble, 476
\@ifdefinable, 171
\@ifundefined, 171
\@indexfile, 292
\@oddhead, 532
\@Roman, 467, 470
\@roman, 467, 470
\@secondoftwo, 462
\@sptoken, 473
\@tfor, 474
\[, 301, 312, 338, 724
\~, 708
\\, 186, 199, 201, 207, 329, 458, 674, 699, 701, 717, 755, 757, 764
*, 329, 701, 764
\^, 708
\], 301, 312, 338, 724
\|, 306, 307
\`, 708, 755
`, 477, 703
", 703
~, 457

", 338, 457
"<, 703
#, 142
$, 142
%, 141
&, 142
', 703
", 703
-, 703
--, 703
---, 703
<, 212, 745
<<, 703
=, 745
>, 212, 745

>>, 703
@, 457, 467, 479
@-espressione, 199
@-espressione, 198, 756
^, 142
_, 142
{, 141
}, 141
␣, 703
~, 674, 703, 717

A

\a', 755
\a=, 755
\a‘, 755
\AA, 708
\aa, 708
\abovedisplayshortskip, 725
\abovedisplayskip, 725
\accent, 629, 660
\addbibresource, 282, 283, 285
\addcontentsline, 462, 488, 490, 710, 711
\addto, 465, 631
\addtocounter, 699, 732
\addtolength, 735
\addvspace, 746
\advance, 736
\AE, 708
\ae, 708
\AfterEndPreamble, 689
\afterpage, 210, 218, 494, 495, 679, 680
\aleph, 307
\allowdisplaybreak, 329
\allowdisplaybreaks, 328, 329
\Alph, 733
\alph, 732, 733
\alpha, 301, 303
\amalg, 304
\and, 717, 742
\angle, 307
\ap, 314, 725
\appendix, 151, 161
\approx, 304
\arabic, 732, 734
\arc, 224, 225
\arc*, 224
\arccos, 305
\arcsin, 305
\arctan, 305
\arg, 305
\arraybackslash, 212
\arraycolsep, 758
\arrayrulewidth, 758
\arraystretch, 202, 758
\Arrowvert, 306
\arrowvert, 306
\ast, 304
\asymp, 304
\AtBeginDocument, 473, 743
\Author, 565, 566
\author, 377, 536, 701, 717
\autore, 536

B

\b, 708
\backmatter, 151, 154, 161, 533, 709, 710
\backslash, 306, 307
\baselineskip, 489, 521, 705, 765, 788
\begin, 147, 174–180, 182, 185, 190, 196, 198, 217, 221, 312, 313, 338, 398, 481, 484, 538, 566, 639, 686, 703, 718, 722–724, 730, 747, 754–756, 760–762, 767, 769, 770, 785
\begingroup, 185
\belowdisplayshortskip, 725
\belowdisplayskip, 725
\beta, 301, 303
Beta Code, 347
\beveljoin, 227
\bfseries, 213, 396, 406, 779
\bgroup, 185, 482, 771
\bibindent, 713

\bibitem, 182–184, 700, 760
\bibliography, 279, 285, 759
\bibliographystyle, 280
\bibname, 490
\bigbreak, 765
\bigcap, 305
\bigcirc, 304
\bigcup, 305
\Biggl, 310
\biggl, 310
\Biggr, 310
\biggr, 310
\Bigl, 310
\bigl, 310, 325
\bigodot, 305
\bigoplus, 305
\bigotimes, 305
\Bigr, 310
\bigr, 310, 325
\bigskip, 746
\bigskipamount, 765
\bigsqcup, 305
\bigtriangledown, 304
\bigtriangleup, 304
\biguplus, 305
\bigvee, 305
\bigwedge, 305
\binom, 330
\bm, 334, 412
\boldmath, 334, 412, 728
\boldsymbol, 334, 412
\boolean, 741
\BooleanFalse, 788
\BooleanTrue, 788
\bot, 307
\botfigrule, 218, 747
\botmark, 531, 533, 534
\botmrule, 790
\bottomfraction, 750, 752
\bottomrule, 203, 758
\bowtie, 304
\Box, 307
\box, 482, 772
\boxed, 335
\bracevert, 306
\break, 382
\bullet, 304
\buttcap, 227

C

\c, 708
\cap, 304
\capolettera, 459
\caption, 189, 190, 197, 210, 214, 484, 708, 709, 750, 772
\cbezier, 223, 224
\cdot, 304
\cdots, 726
\centering, 185–187, 190, 212, 701, 757
\cfrac, 330
\chapter, 138, 154, 461, 488, 531, 533, 591, 708, 709
\chapter*, 138, 139, 710
\chaptermark, 533
\chaptername, 708, 709
\char, 458, 660
\chi, 303
chiave
  angle, 271
  bb, 269
  clip, 271
  height, 270
  keepaspectratio, 270
  trim, 271
  viewport, 270
  width, 269
\circ, 304
\circle, 222, 223
\circle*, 222, 223
\cite, 183, 184, 281, 284, 362, 575, 760
\citeauthor, 284
\citedate, 284
\citetitle, 284
\clap, 789
\cleaders, 491, 493
\cleardoublepage, 742, 749, 765

\clearpage, 218, 673, 678–680, 749, 761, 765
\cline, 202, 337, 758
\clubpenalty, 677, 682
\clubsuit, 307
codice di allineamento
  B, 776
  b, 198, 756, 770, 776
  c, 198, 756, 770
  l, 767, 776
  r, 767, 776
  s, 767
  t, 198, 756, 770, 776
colophon, 536
\color, 777
\colorbox, 777
\columnbreak, 783
\columnsep, 705
\columnseprule, 705
\columnwidth, 270, 705, 737
\cong, 304
contatore
  bottomnumber, 750–752
  dbltopnumber, 751, 752
  footnote, 706
  MaxMatrixCols, 335
  secnumdepth, 709, 710
  section, 462
  table, 215
  tocdepth, 709, 710
  topnumber, 750–752
  totalnumber, 750–752
\coprod, 305
\copyright, 708
\cos, 305
\cosh, 305
\cot, 305
\coth, 305
\counterwithin, 783
\counterwithout, 783
\cs, 457, 458
\csc, 305
\csname, 501, 722, 783
\cspreto, 368
\cup, 304

D

\D, 203
\d, 708
\dag, 708
\dagger, 304
\dashbox, 222, 223
\dashv, 304
\date, 377, 536, 701, 717
\dbinom, 330
\dblfigrule, 218, 747
\dblfloatpagefraction, 751, 752
\dblfloatsep, 751, 752
\dbltextfloatsep, 751, 752
\dbltopfraction, 751, 752
\ddag, 708
\ddagger, 304
\ddots, 726
\DeclareDocumentCommand, 787
\DeclareDocumentEnvironment, 788
\DeclareExpandableDocumentCoommand, 788
\DeclareFontSeriesDefault, 788
\DeclareGraphicsRule, 256, 264
\DeclareMathOperator, 306, 329, 333
\DeclareMathOperator*, 306
\DeclareMathSizes, 779
\DeclareMathSymbol, 473, 689
\DeclareOperatorName, 308
\DeclareRobustCommand, 456, 457, 466, 479, 480
\DeclareRobustCommand*, 456
\def, 171, 203
\definecolor, 776
\deg, 305
\Delta, 301, 303
\delta, 303
\depth, 767
descrittore di colonna
  >, 784
  @, 199
  @, 756
  |, 199
  |, 756
  b, 201, 207, 701

C, 213
c, 199, 211, 213, 756
D, 211
E, 213
L, 213
l, 199, 211, 213, 756
m, 201, 207, 701
p, 199, 201, 207, 211, 701, 757
r, 199, 211, 213, 756
S, 207
s, 207
W, 211, 783
w, 211, 783
X, 206
\det, 305
\dfrac, 330, 331
\Diamond, 307
\diamond, 304
\diamondsuit, 307
\diff, 333
\dim, 305
\dimexpr, 736, 737, 788
\displaybreak, 328, 329
\displaystyle, 215, 300, 562, 729
\div, 304
\divide, 736, 738
\do, 474
\docolaction, 784
\documentclass, 135, 139, 147, 161, 165, 210, 400, 565, 566, 703, 711, 761
\doteq, 304
\dotfill, 491, 493
\Dotline, 228
\dots, 726
\doublehyphendemerits, 676
\doublerulesep, 758
\Downarrow, 304, 306
\downarrow, 304, 306
\dp, 774

E

\edef, 171
\egroup, 185, 482
\ell, 307
\else, 462, 463, 473, 475, 688
\em, 407, 631
\emegencystretch, 676
\emergencystretch, 676
\emph, 117, 215, 407, 630, 631, 704
\emptyset, 307
\end, 138, 147, 171, 176–180, 182, 185, 190, 196, 198, 217, 221, 312, 313, 481, 484, 538, 566, 639, 686, 703, 718, 723, 724, 730, 747, 749, 754–756, 760–762, 767, 769, 770, 785
\endcsname, 501, 722, 783
\endfirsthead, 214
\endfoot, 215
\endgroup, 185
\endhead, 214
\endinput, 169
\endlastfoot, 215
\endlist, 722
\endmulticols, 489
\enlargethispage, 524, 680, 706, 765, 766
\enlargethispage*, 706, 765
\enspace, 213, 338
\ensuremath, 724
\epsilon, 303
\eqref, 313
\equal, 741
\equiv, 304
\ergoup, 771
\errorstopmode, 693
\eta, 303
\etichettadescrizione, 722
\evensidemargin, 716
\exists, 307, 320
\exp, 305
\expandafter, 462, 474, 476, 692, 722
\externaldocument, 790
\extracolsep, 204, 215, 756, 757

F

\f@baselineskip, 467
\fbox, 497, 767, 769, 770, 777
\fboxrule, 482, 767, 777
\fboxsep, 482, 767, 777
\fcolorbox, 777
\fi, 462, 463, 473, 475, 688
\figurename, 708, 709
\filbreak, 682, 766
file
  di formato, 121
  di inizializzazione, 123
\filecontents, 787
\filestyle, 295
\fill, 204, 735, 757
\finalhyphendemerits, 382, 673
\firsthline, 790
\firstmark, 531, 533, 534
\flat, 307
\floatpagefraction, 751, 752
\floatsep, 751, 752
\fnsymbol, 733
font
  Adobe Utopia, 414
  Asana Math, 411
  Bitstream Charter, 414
  Latin Modern, 434
  Latin Modern Math, 411
  Libertinus, 414, 563
  Linux Libertine, 414
  Palladio, 414, 434
  Text Companion Font, 415
  URW Garamond, 414
  Utopia, 414, 434
  XITS Math, 411
\fontencoding, 398
\fontfamily, 786
\fontseries, 447, 786
\fontshape, 447, 786
\fontsize, 467
\fontspec, 427
\footcite, 284
\footnote, 188, 189, 284, 575, 576, 628, 700, 706
\footnotemark, 189, 190, 706
\footnoterule, 707, 747
\footnotesep, 706
\footnotesize, 208, 401, 402, 513, 779
\footnotetext, 189, 190, 700, 706
\footskip, 526
\forall, 307
\fpeval, 228, 464
\frac, 330, 726, 789, 790
\frakfamily, 355
\framebox, 222, 223, 482, 497, 767
\frenchspacing, 491, 543, 704
\frontmatter, 151, 154, 161, 468, 533, 709, 710
\frown, 304
\fussy, 764

G

\Gamma, 301, 303
\gamma, 301, 303
\gcd, 305
\gdef, 171, 473
\ge, 304
\genfrac, 790
\geq, 304
\GetDocumentCommandArgSpec, 788
\GetDocumentEnvironmentArgSpec, 788
\gets, 304
\gg, 304
\glossary, 290, 291, 711, 759, 763
\glossaryentry, 759
\glueexpr, 736
\goodbreak, 766
\goodpagebreak, 681, 682, 766
\goodpagepreak, 682
\gothfamily, 355
\graphicspath, 266–268
\greekfont, 365
\greektxt, 419
\guillemetleft, 783
\guillemetright, 783
\GuIT, 136

H

\H, 708
\halign, 216
\hangafter, 460
\hangindent, 460
\hat, 142
\hbadness, 636
\hbar, 307
\hbox, 466, 467, 498, 769, 771, 772
\hbox to, 771
\header, 526
\headsep, 526
\heartsuit, 307
\height, 767
\hfill, 492, 535, 747
\hfuzz, 636
\hline, 201, 202, 757, 758
\hom, 305
\hookleftarrow, 304
\hookrightarrow, 304
\hrule, 215, 492, 500
\hrulefill, 491, 492
\hsize, 482
\hskip, 491, 630
\hspace, 746
\hspace*, 746
\hss, 493, 722, 772
\ht, 483, 774
\Huge, 401, 774, 779
\huge, 401, 460, 779
\hypersetup, 564, 567
\hyphenation, 67, 632–634, 639, 699
\hyphenpenalty, 676
\hz, 631

I

\i, 578, 657, 708
\idotsint, 332
\if, 475
\ifbool, 744
\IfBooleanF, 788
\IfBooleanT, 788
\IfBooleanTF, 788
\ifcat, 476
\IfClassAtLeastTF, 789
\ifcsdef, 744
\ifcsempty, 745
\ifcsequal, 745
\ifcsname, 476, 783
\ifcsprefix, 744
\ifcsstring, 745
\ifcsundef, 744
\ifdef, 744
\ifdefempty, 745
\ifdefequal, 745
\ifdefined, 462
\ifdefprefix, 744
\ifdefstring, 745
\ifdim, 466
\iff, 304
\ifFamily, 362
\IfFormatAtLeastTF, 789
\iflanguage, 477
\iflettere, 471
\ifLuaTeX, 352
\ifmmode, 744
\IfNoValueF, 788
\IfNoValueT, 788
\IfNoValueTF, 788
\ifnum, 475, 688, 734
\ifnumcomp, 745
\ifnumequal, 745
\ifnumgreater, 745
\ifnumless, 745
\ifnumodd, 745
\ifodd, 495, 734, 744, 745
\IfPackageAtLeastTF, 789
\ifPDFTeX, 352
\ifthen, 746
\ifthenelse, 740, 742
\ifundef, 744
\IfValueF, 788
\IfValueT, 788
\IfValueTF, 788
\ifx, 473, 475, 783
\ifXeTeX, 352
\iiiint, 332

\iiint, 332
\iint, 332
\Im, 307
\imath, 307, 310
\in, 304, 320
\include, 144–146, 148, 686, 759, 761, 765, 784, 788, 790
\includegraphics, 144, 235, 264–267, 269, 271, 379, 483, 498, 500, 583, 752, 776
\includegraphics*, 269, 271
\includeonly, 144, 146, 148, 151, 759, 761, 784, 788, 790
\includesvg, 266
\indent, 186, 705
\index, 287, 288, 291, 293, 294, 458, 711, 714, 759, 762, 763, 783
\indexentry, 288, 759
\indexname, 487
\indexprologue, 294
\indexspace, 486
\indici, 161
\inf, 305
\infty, 307, 476
\input, 144–146, 148, 210, 289, 761, 762, 784
\InputIfFileExist, 785
\insertpenalties, 678
\institute, 377
\int, 305, 332
\IntelligentComma, 174
interprete TEX, 125
\intertext, 330, 331, 705, 778
\intextsep, 751, 752
\iota, 303
\isodd, 741
\isundefined, 741
\it, 405
\item, 179, 180, 182, 379, 488, 630, 699, 718, 719
\itemindent, 719
\itemsep, 719, 721
\itshape, 213, 396, 406, 779

J

\j, 708
\jmath, 307, 310
\jobname, 292, 565–567, 759
\Join, 304
\jot, 725
\justify, 185

K

\kappa, 303
\ker, 305
\Keywords, 565, 566
\kill, 755

L

\L, 708
\l, 708
\l@chapter, 470
\l@figure, 478
\l@table, 478
\label, 181, 184, 189, 190, 192, 197, 313, 321, 328, 462, 484, 706, 719, 725, 730, 732, 750, 759, 760, 783, 784
\labelformat, 785
\labelsep, 719
\labelwidth, 719
\Lambda, 303
\lambda, 303
\langle, 306
\LARGE, 401, 402, 779
\Large, 401, 402, 779
\large, 401, 402, 779
larghezza del font, 384
\lasthline, 790
\LaTeX, 704
\LaTeXe, 704
\lceil, 306
\ldots, 726
\le, 304
leader, 491
\leaders, 491, 493

\leadsto, 304
\leavevmode, 492
\left, 309, 310, 325, 411
\Leftarrow, 304
\leftarrow, 304
\lefteqn, 725
\leftharpoondown, 304
\leftharpoonup, 304
\lefthypHenmin, 639
\lefthyphenmin, 639
\leftmargin, 719, 721
\leftmargini, 719, 721
\leftmarginii, 719
\leftmarginiii, 719
\leftmarginiv, 719
\leftmark, 532–534
\Leftrightarrow, 304
\leftrightarrow, 304
legature e segni diacritici, 385
\lemma, 534
\lengthtest, 741
\leq, 304
\let, 476, 477, 722
\let@token, 473
\lfloor, 306
\lg, 305
\lgroup, 306
\lhd, 304
\lim, 305
\liminf, 305
\limsup, 305
\Line, 224
\line, 222
linea guida, 491
\linebreak, 382, 699, 764
\linepenalty, 676
\lineskip, 678, 705
\lineskiplimit, 705
\linespread, 403, 705
\linethickness, 223
\linewidth, 208, 270, 705, 770
\list, 722
\listfigurename, 465
\listfiles, 762
\listoffigures, 219, 465, 709
\listoftables, 219, 709
\listparindent, 719
\ll, 304
\llap, 460, 722, 789
\lmoustache, 306
\ln, 305
\log, 305
\logo, 378
\long, 729, 744, 745
\Longleftarrow, 304
\longleftarrow, 304
\Longleftrightarrow, 304
\longleftrightarrow, 304
\longmapsto, 304
\Longrightarrow, 304
\longrightarrow, 304
\looseness, 524, 680, 681, 766
\lower, 466
\LTleft, 215
\LTright, 215

M

\m@thcomma, 473
\MacTeX, 457
\mainmatter, 151, 154, 161, 468, 533, 709
\makeatletter, 467, 495
\makebox, 208, 222, 223, 227, 497, 767–769, 772
\makeglossary, 290, 711, 759, 763
\makeindex, 287, 293, 295, 711, 759, 762
\makelabel, 719
\MakeLowercase, 357
\maketitle, 535, 717, 733
\MakeUppercase, 357, 488, 533
\mapsto, 304, 320, 561, 562
\mapstocar, 563
\marginpar, 189, 190, 706, 753
\marginparpush, 707, 753
\marginparsep, 707, 753
\marginparwidth, 707, 753
\markboth, 488, 532–534, 710, 715

\markright, 463, 488, 532–534, 711, 715
\mathbb, 319
\mathbf, 334, 728
\mathcal, 338, 409, 728
\mathchoice, 561, 562
\mathindent, 713, 725, 789
\mathit, 303, 413, 728
\mathop, 328, 333
\mathord, 473
\mathpunct, 473
\mathrm, 311, 728, 778
\mathsf, 413, 728
\mathtt, 413, 728
\max, 305
\maxdepth, 787
\maxdimen, 692, 738
\mbox, 311, 497, 704, 767, 768, 771
\mdseries, 396, 406, 779
\medbreak, 765
\medskip, 215, 746
\medskipamount, 765
\medspace, 790
METAPOST, 132
\mho, 307
\mid, 304
\middle, 411
\midrule, 203, 758
\min, 305
\miterjoin, 227
\mkern, 727
\models, 304
\mp, 304
\mskip, 727
\mu, 303
\muexpr, 736
\multicolumn, 200, 757
\multiply, 736, 738
\multiput, 221, 222, 224, 228

N

\nabla, 307
\natural, 307
\ne, 304
\nearrow, 304
\NeedsTeXFormat, 169
\neg, 307
\negmedspace, 790
\negthickspace, 790
\negthinspace, 790
\neq, 304, 561
\newbool, 744
\newboolean, 741, 742
\newbox, 772
\newcolumntype, 213, 757
\newcommand, 170, 171, 328, 333, 456, 457, 459, 465, 479–481, 491, 729, 764
\newcommand*, 456
\newcount, 745
\newcounter, 699, 731, 745
\NewDocumentCommand, 787
\NewDocumentEnvironment, 788
\newenvironment, 480, 484, 487, 490, 730, 771
\NewExpandableDocumentCommand, 788
\newfloat, 538
\newfontfamily, 427
\newif, 741, 744
\newlength, 699, 735
\newline, 199, 701, 764
\newpage, 524, 681, 682, 743, 765, 766
\newsavebox, 497, 699, 768, 769, 772
\newtheorem, 159, 699, 730
\nexists, 320
\ni, 304
\noalign, 215
\nobreak, 382, 631
\nocite, 280, 284, 285
\noexpand, 476
\nofiles, 759
\noindent, 186, 187, 482, 705
\NoIntelligentComma, 174
\nolinebreak, 699, 764
\nonfrenchspacing, 704
\nonumber, 725
\nopagebreak, 699, 765

\normalbaselineskip, 521
\normalfont, 405, 779
\normalmarginpar, 707, 753
\normalsize, 401, 779
\not, 304, 320, 561, 563, 742
\notag, 322
\notbool, 744
\notin, 320
\nu, 303
\null, 237, 535
\numberline, 462, 711
numerazione
  Alph, 715, 733
  alph, 715, 732
  arabic, 715, 732
  fnsymbol, 715, 733
  Roman, 715, 732
  roman, 715, 732
\numexpr, 736, 745
\nwarrow, 304

O

\O, 708
\o, 708
occhiello, 536
\oddsidemargin, 716
\odot, 304
\OE, 708
\oe, 708
\ohm, 314, 479
\oint, 305
\oldstylemums, 786
\oldstylenums, 412
\Omega, 303, 314
\omega, 303
\ominus, 304
\onecolumn, 716
\oplus, 304
opzione
  10pt, 459, 513, 522, 712, 721
  11pt, 400, 403, 513, 712, 778
  12pt, 400, 513, 712
  a-1b, 564
  a4paper, 712
  a5paper, 712
  ancient, 420
  ansinew, 107, 137, 650
  applemac, 137, 650, 652, 661, 662
  b5paper, 712
  babelshorthands, 67
  backend, 283
  charter, 414
  columns, 294
  config, 388
  cp1250, 650
  cp1252, 650
  cp437, 650
  cyrillicfont, 418
  draft, 712
  dvipdfm, 582
  ecclesiastic, 421
  enable-write18, 261
  executivepaper, 712
  final, 712
  fleqn, 713, 725, 789
  garamond, 414
  greek, 347, 349
  hyperref, 283
  ibycus, 347
  intoc, 294, 489
  italian, 165, 174, 175, 314–316, 338, 376, 472, 577, 725, 726
  landscape, 370, 712
  latin, 356, 421
  latin1, 110, 137, 581, 650, 661, 663, 668, 671
  latin9, 137, 582, 650
  legalpaper, 712
  leqno, 713
  letterpaper, 712
  LGR, 360, 362, 641, 786
  ltxarrows, 220
  LY1, 434
  main, 416
  mono, 420
  name, 294
  noheader, 762, 785
  nosearch, 785

notitlepage, 713
oldstyle, 354
onecolumn, 712
oneside, 712
openany, 712
openbib, 490, 713
openright, 712
options, 294
origin, 776
original, 220
OT1, 137, 353, 660
ot1, 656
OT2, 416, 417
overwrite, 762, 784
pdfa, 563
polutoniko, 349
poly, 420
pstarrows, 220
shell-escape, 261
style, 282
T1, 136, 137, 353, 362, 366, 368, 376, 416, 434, 445, 447, 641, 643, 652, 658–660, 669, 786
T2, 434
T2A, 416, 418
T2B, 416
T2C, 416
title, 294
titlepage, 713, 716
TS1, 434, 652
TU, 368
twocolumn, 712
twoside, 712, 715
utf8, 100, 102, 108, 110, 136, 137, 351, 358, 376, 418, 575, 578, 582, 642, 643, 650, 651, 656, 661, 663, 668, 669, 671, 782
utopia, 414
veryoldstyle, 353, 354
x-1a, 564
X2, 30, 416–418
`\or`, 742
`\Org`, 565, 566
`\oslash`, 304
`\otimes`, 304
`\oval`, 222, 223
`\overbrace`, 310, 311
`\overline`, 727
`\overset`, 727, 785
`\overunderset`, 727, 785

P

`\P`, 708
`\packstyle`, 295, 463
`\pagebreak`, 329, 673, 699, 765, 783
`\pagecolor`, 699, 777
`\pagenumbering`, 699, 715
`\pageref`, 181, 184, 197, 313, 725, 759
`\pagestyle`, 715
`\paperheight`, 738
`\paperwidth`, 738
`\par`, 186, 187, 398, 459, 482, 489, 705, 729, 789
`\paragraph`, 341, 708, 709
`\parallel`, 304
`\parbox`, 237, 497, 500, 729, 757, 770, 772
`\parencite`, 284
`\parindent`, 705
`\parsep`, 719, 721
`\parskip`, 678, 705
`\part`, 153, 154, 461, 708, 709
`\partial`, 307
`\partname`, 708, 709
`\partopsep`, 719
`\pdfoutput`, 126
`\ped`, 314, 726
`\perp`, 304
`\pgfdeclareimage`, 377, 378
`\pgfuseimage`, 378, 379
`\phantom`, 783
`\Phi`, 303
`\phi`, 303
`\Pi`, 303
`\pi`, 303
piedino, 56

\pm, 304
\pmb, 334
\polygon, 224, 225
\polygon*, 224, 225
\polyline, 224, 225, 227
\poptabs, 755
posizione degli oggetti flottanti
  !, 196, 218, 747
  b, 196, 218, 679, 747
  c, 211
  H, 218, 219
  h, 196, 218, 679, 747–749
  l, 211
  p, 196, 218, 679, 680, 747, 751
  r, 211
  t, 196, 218, 679, 747–749
\pounds, 708
\Pr, 305
\prec, 304
\preceq, 304
\pretocmd, 631
\pretolerance, 383, 492, 675, 676
\prime, 307
\printbibliography, 489, 759
\printglossary, 290
\printindex, 289, 290, 293, 294, 296, 542, 762
\ProcessList, 788
\prod, 305
\progstyle, 295
\propto, 304
\protect, 462, 480, 700, 711, 744, 745
protrusione dei segni grafici, 384
protrusione della punteggiatura, 384
\providebool, 744
\provideboolean, 741, 742
\ProvideClass, 787
\providecommand, 456, 457, 479, 485, 729
\providecommand*, 456
\ProvideDocumentCommand, 787
\ProvideDocumentEnvironment, 788
\ProvideExpandableDocumentCommand, 788
\ProvideFile, 787
\ProvidePackage, 787
\ProvidesClass, 170
\ProvidesPackage, 169
\ProvideTextCommandDefault, 656
\Psi, 303
\psi, 303
\pushtabs, 755
\put, 221–225, 228, 236

Q

\qbezier, 223, 224
\qquad, 324, 341, 394, 727
\quad, 324, 341, 394, 462, 727

R

\r, 708
\raggedleft, 185, 190, 199, 212, 701, 753, 757
\raggedright, 185, 190, 199, 212, 701, 757, 787
\raisebox, 460, 771
\rangle, 306
\rBrace, 237
\rceil, 306
\Re, 307
\Ref, 785
\ref, 181, 184, 192, 197, 313, 462, 706, 719, 725, 730, 732, 759
\reflectbox, 466, 776
\refstepcounter, 462, 732, 760, 785
refuso, 61
\relax, 171, 467, 688, 736, 745, 747, 783
\renewcommand, 170, 171, 202, 464–466, 485, 707, 729, 751, 758
\renewcommand*, 464
\RenewDocumentCommand, 787
\RenewDocumentEnvironment, 788
\renewenvironment, 480, 484, 487, 490, 730

\RenewExpandableDocumentCommand, 788
\RequirePackage, 161, 169, 495
\resizebox, 208, 484, 776
\ReverseBoolean, 788
\reversemarginpar, 190, 707, 753
\rfloor, 306
\rgroup, 306
\rhd, 304
\rho, 303
\right, 309, 310, 325, 411
\Rightarrow, 304
\rightarrow, 304
\rightharpoondown, 304
\rightharpoonup, 304
\righthyphenmin, 639
\rightleftharpoons, 304
\rightmargin, 719
\rightmark, 532–534
risposta all'errore
  e, 152
  h, 152
  i, 152
  Invio, 151
  q, 152
  s, 152
  x, 152
\rlap, 562, 789
\rm, 405
\rmfamily, 396, 405, 406, 779
\rmoustache, 306
\robustify, 480
\Roman, 732
\roman, 467, 732
\romannumeral, 470
\rotatebox, 776
\roundcap, 227
\roundjoin, 227
\rule, 203, 500, 562, 771

S

\S, 708
\savebox, 497, 768
\sbox, 497, 768, 772
\scalebox, 775, 776
\scriptscriptstyle, 300, 562, 729
\scriptsize, 311, 401, 779
\scriptstyle, 300, 562, 729
\scshape, 396, 406, 779
\searrow, 304
\sec, 305
\section, 117, 154, 461, 488, 531, 533, 591, 708, 709, 783
\section*, 461, 462, 710, 711
\selectfont, 398, 447, 467, 705
\selectlanguage, 465
\sep, 565, 566
\seriesdefault, 788
\setactivedoublequote, 67, 174, 338
\setbeamercolor, 377
\setbeamercovered, 377
\setboolean, 741
\setbox, 482, 769, 772, 773
\setcounter, 335, 699, 710, 731, 751
\setfontsize, 463, 464
\setISOcompliance, 174, 175, 314
\setlength, 221, 713, 735, 751
\setmainfont, 427
\setmainlanguage, 174
\setmathfont, 412, 427
\setminus, 304
\setmonofont, 427
\setsansfont, 427
\settodepth, 735, 773
\settoheight, 735, 773
\settowidth, 735, 773
\sf@size, 467
\sfcode, 490, 491
\sfdefault, 368
\sffamily, 396, 406, 779
\sharp, 307
shell editor, 104
\shipoutAnswer, 193
\shorthandoff, 338, 339
\shorthandon, 338, 339
\shortstack, 701
\show, 783

\showboxbreadth, 690, 692
\showboxdepth, 691, 692
\ShowCommand, 480, 789
\showhyphens, 629, 635
\SI, 316
\Sigma, 303
\sigma, 303
\sim, 304
\simeq, 304
\simulatedSC, 467, 470, 471
\sin, 305
\sinh, 305
\sloppy, 764
\slshape, 396, 406, 779
\small, 208, 401, 402, 513, 779
\smallbreak, 765
\smallskip, 215, 746
\smallskipamount, 765
\smartcite, 284
\smash, 783
\smile, 304
\spadesuit, 307
spazieggiatura, 384
\special, 121
\SplitArgument, 788
\SplitList, 788
\sqcap, 304
\sqcup, 304
\sqrt, 305, 726
\sqsubset, 304
\sqsubseteq, 304
\sqsupset, 304
\sqsupseteq, 304
\squarecap, 227
\ss, 708
\stackrel, 727
\star, 304
\stepcounter, 732
stile della pagina
  empty, 488, 715
  headings, 488, 532, 533, 715
  myheadings, 488, 532, 533, 715
  plain, 488, 715
\stretch, 735
\stretchandshrink, 215
\string, 476
\strut, 201–203, 702, 706, 758
\subparagraph, 153, 708, 709
\subsection, 531, 591, 708, 709, 783
\subset, 304
\subseteq, 304
\substack, 332
\subsubsection, 591, 708, 709
\subtitle, 377
\succ, 304
\succeq, 304
\sum, 305
\sup, 305
\Supaginadispari, 495
\Supaginapari, 495
\suppressfloats, 699, 750
\supset, 304
\supseteq, 304
\surd, 307
\swabfamily, 355
\swarrow, 304
\symbol, 779

T

\t, 708
\T1/lmr/m/n/10, 447
\tabcolsep, 206–208, 758
\tablename, 708, 709
\tableofcontents, 709
\tabucline, 337
\tabularimage, 500, 501
\tabularnewline, 201, 207
\tabulinestyle, 337
\tag, 313, 322
\tan, 305
\tanh, 305
\tau, 303
testatina, 56
\testlettere, 471, 472
\TeX, 457, 704
\texorpdfstring, 462, 463
\text, 311, 327, 331, 705, 778
\textasciicircum, 142

\textasciitilde, 142
\textbackslash, 142, 458, 787
\textbf, 334, 396, 406, 779
\textcelsius, 577
\textcent, 657
\textcite, 284
\textcolor, 777
\textcurrency, 657
\textcyrillic, 418
\textdegree, 576, 657
\textdollar, 142
\texteuro, 213, 434
\textfloatsep, 751, 752
\textfraction, 751, 752
\textfrak, 355
\textgoth, 355
\textheight, 270, 521, 706, 716, 738
\textit, 396, 728, 779
\textlangle, 785
\textmd, 396, 779
\textmho, 415
\textnormal, 779
\textnumero, 74
\textohm, 314
\textormath, 479, 724
\textrangle, 785
\textrm, 396, 405, 406, 705, 779
\textsc, 396, 467, 468, 779
\textsf, 396, 406, 779
\textsl, 396, 406, 779
\textstyle, 234, 300, 562, 729
\textsubscript, 725, 788
\textsupercript, 788
\textsuperscript, 725
\textswab, 355
\texttt, 396, 458, 779
\textunderscore, 142
\textup, 396, 779
\textvisiblespace, 648
\textwidth, 270, 512, 705, 716, 737, 738
\tfrac, 330
\thanks, 700, 717, 733
\the, 730, 733, 734, 774
\thepage, 468
\Theta, 303
\theta, 303
\thetable, 215
\thicklines, 223
\thikspace, 790
\thinlines, 223
\thinspace, 338, 783, 790
\thispagestyle, 488, 699, 715
thumbnail, 120
\times, 304
\tiny, 401, 779
\Title, 565, 566
\title, 377, 536, 701, 717
\titlepage, 378
\titolo, 536
\to, 304
\today, 704
\tolerance, 383, 492, 639, 675, 676
\top, 307
\topcaption, 210
\topfigrule, 218, 747
\topfraction, 750, 752
\topmargin, 716
\topmark, 531, 533, 534
\toprule, 203, 758, 790
\topsep, 719, 721
\topskip, 521, 522
\totalheight, 767
\traceoff, 691, 692, 697
\traceon, 691, 697
\tracingassigns, 690
\tracingcommands, 690, 691
\tracinglostchars, 690
\tracingmacros, 690, 691
\tracingonline, 690
\tracingoutput, 690
\tracingpages, 690, 694
\tracingparagraphs, 690, 694
\tracingrestores, 690
\tracingstats, 690
\triangle, 307
\TrimSpaces, 788
\ttfamily, 396, 406, 779
\twocolumn, 487, 699, 716, 748

\typein, 700, 763, 764
\typeout, 700, 763, 764, 789

U

\U, 203, 306
\u, 306, 708
\uimm, 307, 328
\uishape, 406
\unboldmath, 334, 412, 728
\underbrace, 310, 311
\underline, 727
\underset, 727, 785
\unhbox, 772, 773
\unhcopy, 773
\unit, 174, 175, 314–316
unità di misura
  bp, 735
  cc, 735
  cm, 735
  dd, 735
  em, 466, 735
  ex, 735
  in, 735
  mm, 735
  mu, 727
  pc, 735
  pt, 735
  sp, 735
\unita, 316
\unitlength, 220, 221, 237, 788
\unitre, 438
unità di misura, 49
  big point, 50, 735
  cicero, 51
  pica, 51
  punto didot, 50
  punto PostScript, 50
  punto tipografico anglosassone, 49
\unlhd, 304
\unrhd, 304
\unvbox, 482, 773
\unvcopy, 773
\Uparrow, 304, 306
\uparrow, 304, 306
\uplus, 304
\upshape, 396, 406, 779
\Upsilon, 303
\upsilon, 303
\usebox, 497, 769, 773
\usecounter, 719
\usefont, 398
\usefonttheme, 378
\useoutertheme, 377
\usepackage, 136, 164, 169, 183, 184, 220, 282, 293, 378, 398, 412, 427, 467, 495, 713, 761
\usetheme, 376, 380

V

\V, 203, 306
\v, 708
\value, 495, 732, 734, 736, 740, 741
\varDelta, 302
\varepsilon, 303
\varGamma, 302
\varphi, 303
\varpi, 303
\varrho, 303
\varsigma, 303
\varTheta, 302
\vartheta, 303
\vbadness, 636
\vbox, 482, 498, 772
\vbox to, 772
\vcenter, 498, 772
\vcenter to, 772
\vdash, 304
\vdots, 726
\vector, 222
\vee, 304
\verb, 723
\verb*, 723, 783
\vert, 306
\vfill, 747
\vfuzz, 636
\virgola, 473, 474, 476, 689

virgola intelligente, 175
\virgoladecimale, 473, 474, 477
\vline, 199, 756
\vref, 785
\vspace, 187, 458, 489, 746
\vspace*, 187, 746
\vsplit, 773
\vss, 772
\vtop, 498, 772, 773
\vtop to, 772

W

\wd, 774
\wedge, 304
\whiledo, 537, 742
\widowpenalty, 677, 682
\width, 539, 767
\wp, 307
\wr, 304
\write18, 292, 293

X

\X, 228
\xdef, 171
\XeLaTeX, 466
\XelaTeX, 466
\Xi, 303
\xi, 303
\xleaders, 491, 493
\xleftarrow, 332
\xrightarrow, 332

Y

\Y, 228

Z

\z@, 631
\zeta, 303

Indice degli ambienti

A

abstract, 154, 717
align, 325, 326, 328, 341
alignat, 327
aligned, 321–323, 328, 790
alignedat, 328
alltt, 713, 723
amsmath, 324
Answer, 192
array, 201, 207, 337, 700, 701, 755, 756

B

Bmatrix, 335
bmatrix, 335
bulgarian, 418

C

cases, 309, 331
CD, 338
center, 185, 187
checkhyphens, 629

D

description, 181, 182, 718, 722
descrizione, 722
displaymath, 301, 312, 313, 322, 724
document, 139, 210, 574

E

enumerate, 179, 719, 722
eqnarray, 321, 322, 701, 724, 725
*eqnarray**, 321, 724
equation, 312, 322, 325, 724, 725
*equation**, 313, 322
Exercise, 192

F

FIGURA, 483
figure, 187, 217, 219, 220, 265, 538, 750
filecontents, 565, 567, 703, 784
flalign, 326
flushleft, 185, 580
flushright, 185
frame, 372, 378
fussy, 676

G

gather, 325
gathered, 328, 790
glossary, 711
greek, 419

I

index, 711
itemize, 180

L

letter, 700
list, 718, 719, 721, 722
longtable, 215
longtabu, 207
lrbox, 769, 770

M

math, 724
matrix, 335
medaglione, 22, 481, 482
minibox, 630
minipage, 189, 497, 501, 630, 706, 757, 769–771
multicols, 386, 487, 489
*multicols**, 784
multline, 323, 324, 328

P

picture, 219–221, 223, 225, 228, 229, 231, 233, 235–237, 370, 379, 456, 509, 714, 774, 775, 788
pmatrix, 335

Q

quotation, 176
quote, 176

R

riquadro, 481, 769, 770
russian, 418

S

sidewaystable, 209
sintassi, 22, 481
sloppy, 676
sloppypar, 639
split, 321–323, 328
subarray, 790
subequations, 321, 327, 328

T

tabbing, 701, 754
table, 187, 196, 198, 202, 203, 217–219, 237, 538, 750
tabu, 207, 212, 216, 337
tabular, 189, 198, 206, 207, 219, 235, 700, 701, 755–757
*tabular**, 198, 206, 207, 756
tabularx, 206, 207
tela, 221
thebibliography, 22, 182, 183, 275, 276, 489, 760
theglossary, 291
theindex, 22, 289, 291, 485, 486, 488, 762
tikzpicture, 230, 231, 233, 775
titlepage, 535, 536, 717, 733

U

ukranian, 418

V

verbatim, 178, 723
*verbatim**, 783
verse, 176
Vmatrix, 335
vmatrix, 335

W

wrapfigure, 538
wraptable, 538

X

xy, 339

Indice delle classi

A

article, 135, 154, 155, 531, 635, 708, 712, 713

B

beamer, 20, 155, 371
book, 76, 135, 153–156, 161–163, 170, 179, 183, 210, 292, 346, 349, 463, 486, 489, 490, 512, 527, 533, 535, 537, 545, 546, 709, 712, 713, 719, 721, 751, 752
book.cls, 489

L

layaureo, 346
letter, 154, 700, 708, 712
ltnews, 154
ltxdoc, 154, 712
ltxguide, 154

M

memoir, 158, 196, 292, 347, 445, 505, 527, 530, 533, 535, 543
minimal, 155, 712

N

ncc, 159

O

octavo, 76, 155, 346, 517, 520
ottavo, 346

P

proc, 155, 712

R

report, 135, 154, 161, 712, 713

S

scrartcl, 157
scrbook, 157
scrlttr2, 157
scrreprt, 157
slides, 155, 369, 370, 378, 712
suftesi, 76

T

topesi, 161
toptesi, 161

Indice dei file

Estensioni

.aff, 109
.afm, 424, 440
.aux, 85, 146, 279, 759, 761
.bat, 131
.bbl, 279, 281, 759
.bib, 276, 279, 282
.bmp, 262
.bst, 280, 282
.dat, 643
.def, 658
.dic, 109
.doc, 584, 668
.docx, 572–575, 578–582, 584
.dtx, 154, 502
.dvi, 291, 450, 451, 712, 759
.engine, 259
.eps, 254, 260, 262, 263, 266
.fd, 398, 436, 444, 445
.fmt, 122
.fodt, 588
.gif, 684
.glo, 292, 759
.gls, 292
.icc, 558
.icm, 558
.idx, 288, 292, 759
.ind, 289, 292, 759
.ini, 123
.jpg, 263, 268
.lof, 709, 759
.log, 85, 122, 132, 146, 273, 307, 448, 480, 629, 630, 686, 689, 692–694, 753, 754, 759, 762, 784
.log , 629
.lot, 709, 759
.ltx, 123, 144, 758
.lyx, 582
.map, 436, 438
.mf, 132, 452
.mp, 256
.mps, 133, 256
.odt, 584
.otf, 354, 423, 424, 426
.pdf, 254, 256, 260, 262, 263, 266, 291, 451, 573, 649, 668, 712
.pfa, 440
.pfb, 423, 438, 440, 441, 561
.png, 263, 684
.prj, 111
.ps, 254, 262, 451, 712
.rtf, 573, 579, 584
.scriv, 590, 591
.sty, 169, 380, 436, 467
.svg, 254, 261, 266
.sxw, 584
.tex, 88, 89, 97, 119, 123, 134, 144, 267, 508, 565, 574, 578, 580–584, 648, 649, 652, 658, 661–664, 668, 670, 684, 700, 755, 758, 780
.tfm, 423, 424, 436, 439, 440, 452

.toc, 709, 759
.ttf, 423, 424, 426, 441
.txt, 573
.vf, 436, 440
.vpl, 440
.xml, 584
.xmpdata, 565, 567
.zip, 118

A

auctex, 86
Autorun.inf, 83

B

babel-italian, 704
babel-italian.ld, 174
beamerexample2.article.pdf, 372
beamerexample2.beamer.pdf, 372
beamerexample5.pdf, 372
beameruserguide.pdf, 372
bk10.clo, 512, 522, 721
bohahyph.tex,, 642
book.cls, 170, 478

C

clsguide, 155
cmr10.300pk, 439

D

dvipdfm.map, 425, 441
dvipsnam.def, 777

E

encguide.pdf, 652
endnote.tex, 188
endnotes.sty, 188
esempio1.pdf, 140
esempio1.tex, 139
euler.pdf, 414

F

fntguide.pdf, 413, 466, 778
fontmath.ltx, 413
fonts/map/, 443
fourier-doc-en.pdf, 414

G

gfsartemisia, 359
gfsbaskerville, 359
gfsbodoni, 359
gfscomplutum, 359
gfsdidot, 359
gfsporson, 360
gfssolomos, 360
gloss-italian, 704
gloss-italian.ld, 174
greekfont, 420
grfguide.pdf, 775
GuidaGuIT.tex, 565
GuidaGuIT.xmpdata, 565

H

hyph-it.tex, 635
hyph-quote-it.tex, 635

I

ind.pdf, 288
it_IT.aff, 109
it_IT.dic, 109
italian.ldf, 641, 689
ithyph.tex, 635

K

kpfonts, 388

L

l3fp, 129
language-local.dat, 642
language-local.def, 642
language.dat.lua, 642

latex.fmt, 122, 126
latex.ltx, 121, 124, 455, 722
latin.ldf, 641
latin1.def, 652, 656
libretto.pdf, 508
loadhyph-it.tex, 635
local.map, 443, 444, 454

M

MacroGuida.sty, 76, 228, 231, 459, 472, 481, 494, 495
makebst.tex, 280
makeindex.pdf, 288
mathdesign-doc.pdf, 414
Mathmode.pdf, 343
memdesign, 158
merlin.mbs, 280
mf, 439
microtype.cfg, 388
miofont.mf, 439
missfont.log, 132
mt-MinionPro.cfg, 388
mymacros.sty, 170, 171, 486, 489

N

neohellenic, 360
nuovateoria.tex, 147

O

ot1enc.def, 660

P

pag.fd, 366
pagk8r.pfb, 366
pdftex.map, 366, 425, 435, 441
pict2e.cfg, 220
plain.bst, 280
plain.tex, 122, 124, 455
protext-setup-en.pdf, 83
ps4pdf.engine, 259
psfonts.map, 425, 441
psnfss2e.pdf, 433

S

scrguien.pdf, 530
segnature.pdf, 508
source2e.pdf, 786

T

t1enc.def, 652, 659
t1lmr.fd, 445
tds.pdf, 89, 94
tex, 24, 589–591, 593, 596
texlive2016-20160530.iso, 82
TeXworks.app, 118
tiger.eps, 260
topfront.sty, 160
toptesi.cls, 161
toptesi.sty, 160, 161
tttomia.cls, 170
txfonts, 360
txfontsx, 360
txt, 24, 587–589

U

uni3.map, 438
uni3.pfb, 438
uni3.tfm, 438
unsrt.bst, 280
updmap.cfg, 442–444, 453, 560
updmap.map, 454

Indice dei pacchetti

A

afterpage, 210, 218, 494, 495
alltt, 713, 723
amscd, 338
amsfonts, 302, 319, 413, 428, 714
amsmath, 159, 216, 299, 302, 306, 309, 311–313, 317, 319, 321, 325, 328–330, 332, 334–336, 338, 343, 412, 414, 419, 428, 550, 705, 713, 714, 724–727, 778, 781, 784, 785, 787, 789, 790
amssymb, 319, 320, 412, 428, 714
array, 199, 201, 211–213, 216, 498, 757, 783–785, 790
auctex, 107, 112, 117
auto-pst-pdf, 230
avant, 366

B

babel, 67, 136, 137, 141, 145, 159, 165, 174–176, 192, 282, 314–316, 338, 339, 347–349, 356, 362, 365, 376, 382, 416, 418, 419, 421, 457, 465, 472, 477, 479, 487, 490, 577, 629, 632, 640–642, 689, 708, 714, 716, 724–726, 781, 790
beamer, 155, 371, 372, 377–380
begingreek, 419
bgreek, 349
biblatex, 276, 277, 279, 282–285, 463
bm, 334, 412
bohairic, 421
bookktabs, 498
booklet, 508, 509
booktabs, 203, 207, 757, 758, 790

C

calc, 464, 731, 788
caption, 156, 196, 197, 531
chngcntr, 783
circuitikz, 231
ClassicThesis, 76, 157, 160, 545
cleveref, 785
color, 370, 541, 559, 714, 775, 787
combelow, 659
cool, 550
coptic, 363, 421
crop, 511
csquotes, 282
cureve2e, 228
curve2e, 165, 222, 228, 229, 374, 376, 379

D

dcolumn, 211

E

ecclesiastic, 421
edmac, 361
edstanza, 361
eledmac, 361
endnotes, 188, 361
epsconversion, 260
epstopdf, 260
etex, 735
etoolbox, 26, 368, 480, 488, 631, 689, 740, 743, 744, 746, 781
euclideangeometry, 229
euler, 414
exercise, 192
expl3, 786

F

fancyhdr, 23, 156, 513, 514, 526, 531, 715
fancyvrb, 723
filecontents, 784
fixltx2e, 748, 749
float, 218, 219, 536, 538
floatrow, 219
fncylab, 785
fontenc, 376, 628, 649, 651, 669, 780
fontsize, 463
fontspec, 128, 368, 420, 427–429, 431, 432, 669, 785, 786
fourier, 414, 434, 472, 525
frontespizio, 160, 161, 535

G

geometry, 156, 527, 530, 716
ghostscript, 96
glossaries, 290, 291, 763
glossary, 291
graphics, 541, 714, 775
graphicx, 208, 253, 256, 265, 379, 460, 466, 498, 541, 559, 714, 775, 781, 787
guit, 136, 376

H

hyperref, 461, 463, 468, 563, 564, 567, 582, 759, 784, 785, 788
hyperrref, 564
hyphenator.js, 627

I

icomma, 175
IEEEtrantools, 216
iftex, 352
ifthen, 26, 537, 714, 740–742, 744
imakeidx, 292, 293, 295, 296, 486, 564, 714, 761–763
indentfirst, 524
indextools, 293, 714, 761–763
inputenc, 100, 102, 117, 376, 577, 628, 649, 651, 656, 661, 663, 669, 707, 780, 782

K

koma-script, 157
kpfonts, 353, 354, 414, 434
kuvio, 338

L

latexsym, 302, 714
layaureo, 162, 163, 519, 546
layout, 165
ledmac, 361
lettrine, 459, 460, 525
libertinus, 414
longtable, 209, 211
ltxsymb, 319
luatex-math, 790
lxslides, 378

M

MacTeX.pkg, 87
makeidx, 289, 714, 762
mathdesign, 414, 416, 433
mathenv, 216
mdwtab, 216
metalogo, 466
metre, 363
mfnfss, 355
microtype, 381, 382, 385–389, 636, 639, 674
morefloats, 750, 754
multicol, 486, 487, 716, 748, 783–785, 787
multirow, 200

N

natbib, 276, 279, 280
ncccomma, 175
newpx, 432
newpxmath, 408, 414
newpxtext, 408, 414
newtx, 432
newtxmath, 408, 414, 432
newtxtext, 408, 414, 432

O

opcit, 361

P

pdfcomment, 258
pdfpages, 23, 259, 508, 509, 563, 566
pdfscreen, 371
pdfslide, 371
pdfx, 558, 559, 561, 563–565, 567, 568
pgf, 229, 230, 232, 253, 371, 372, 376, 775
pgfplots, 134
pic2e, 165
pict2e, 220, 224, 225, 229, 376, 379, 714, 775, 781
Pifonts, 433
poemscol, 361
polyglossia, 67, 137, 141, 174, 192, 282, 365, 366, 420, 421, 427, 640–642, 708, 714, 716, 724–726, 781
powerdot, 370, 380
ppower4, 371
prosper, 370, 380
psfrag, 233
pst-pdf, 230
PSTricks, 230, 258, 259, 261, 380, 775
pxfonts, 408, 414, 432

R

reledmac, 361
remreset, 783
rotating, 209
rotfloat, 219

S

sapthesis, 160
scrextend, 157
sectsty, 531
seminar, 371
showidx, 714
SIstyle, 316
SIunits, 315, 316
siunitx, 175, 207, 314–316
stanalone, 269
standalone, 265
subfigure, 482
suftesi, 160, 347
supertabular, 209, 211
svg, 261, 266

T

tabmac, 361
tabu, 206, 207, 211, 216, 337

tabularx, 205, 206, 211
tagpdf, 556, 564
technica, 347, 349
TesiClassica, 160
TesiModerna, 160
testhyphens, 629
teubner, 349, 358, 360, 362, 363, 419–421
texcomp, 786
texpower, 371
textcomp, 74, 416, 582, 650–652, 657, 786
tikz, 134, 231, 338
titlesec, 531, 532
tocloft, 469, 472, 531
topcapt, 210
TOPtesi, 160, 161
toptesi.sty, 161
trace, 691, 692, 696, 697, 784
txfonts, 408, 414, 432
type1ec, 459, 712
typearea, 156, 157, 520, 527, 530

U

unicode-math, 128, 411, 412, 428, 728
unifith, 160
units, 315
utopia, 414

V

varioref, 165, 184, 785

W

widetable, 206
wrapfig, 538

X

xcolor, 207, 371, 372, 531, 559, 775
xfp, 129, 228, 464, 736, 739, 740
xparse, 171, 455, 456, 461, 464, 485, 700, 701, 786–788
xpatch, 170, 171, 321
xr, 784, 785, 790
xr-hyper, 785
xypic, 175, 338, 339

Indice dei programmi e delle distribuzioni

A

AbiWord, 580, 583, 584
Acrobat Pro, 569
Acrobat Pro XI, 568
Acrobat Professional, 556, 563
Adobe Acrobat, 96, 557
Adobe Reader, 96, 111, 559, 573
afm2tfm, 440
Aleph, 129, 130
aleph, 134, 555
Anteprima, 96, 265, 583
Aquamacs, 117, 118, 652, 662, 663, 669
arara, 133
arlatex, 133
asy, 134
Asymptote, 134
auctex, 112, 118
autoinst, 440, 441

B

bash, 454
BibDesk, 87, 131, 276, 284
biber, 117, 131, 183, 276, 277, 279, 282–285, 290, 759, 760
BibTeX, 19, 114, 117, 131, 183, 276, 277, 279, 281–285, 290, 361, 468, 759, 760
bibtex8, 131, 277
bibtexu, 131, 277

C

Character map, 98
charco, 668
context, 134, 555, 585, 673
context-mk-iv, 661
convert, 264
CygWin, 112, 439, 580, 668

D

dbcontext, 585
dblatex, 585
Draw, 583
dvipdfm, 120, 132, 258, 259, 380, 441, 555, 560
dvipdfmx, 132, 258, 259, 425, 439
dvips, 96, 120, 132, 220, 230, 234, 258, 259, 380, 425, 439, 441, 451, 560, 585, 777

E

ebb, 258
elatex, 125
emacs, 86, 107, 110, 112, 117, 118, 585, 652, 662
eps2jpg, 263

eps2pdf, 262
eps2png, 263
epspdf, 262
epstopdf, 132, 262
ε-TEX, 471, 782, 783
etex, 122, 124, 125, 129, 133, 292, 411, 498, 522, 532, 731, 736
EurKey, 100
Excel, 244, 583

F

fmtutil, 133
fmtutil-sys, 133
FontBook, 426
FontBook.app, 354
FontForge, 129, 436, 438, 440
fontinst, 440

G

gedit, 114
gedit-LaTeX-plugin, 114
ghostscript, 84, 96, 262, 265, 555, 556, 563, 564, 567
ghostview, 84, 96, 262
gimp, 97, 264, 560
gnome, 114
gnuplot, 240, 242, 244
gs, 258, 260
gv, 96, 262
gview, 96, 262, 264, 273

I

iconv, 664, 668
ImageMagik, 264
InDesign, 42
indextools, 296
inkscape, 261, 262, 560

J

JabRef, 276, 284
Jabref, 131
java, 454
jpegtops, 97, 263

K

Kile, 86, 111–115, 118
kpathsea, 129
kpseaccess, 133
kpsepath, 133
kpsereadlink, 133
kpsestat, 133
kpsetool, 133
kpsewhere, 133
kpsewhich, 89, 133, 442
kpsexpand, 133
Kword, 580, 583, 584

L

lamed, 134
latex, 96, 107, 120, 121, 123, 124, 126, 133, 230, 256, 258–260, 380–382, 384, 385, 423, 425, 450, 582, 585, 782
LaTeXiT, 87–89
latexmk, 133
LED, 110, 111, 126
lialatex, 359
Libre Office, 591, 647
LibreOffice, 627
LibroFont, 426
lua, 129
lualatex, 23, 35, 67, 117, 120, 121, 123, 124, 126, 129, 130, 132–134, 137, 162, 174, 230, 234, 256, 262, 293, 352, 354, 357, 359, 362, 365, 366, 368, 377, 381, 382, 385, 397, 398, 400, 411, 418, 421, 423–426, 428, 432, 434, 466, 561, 563–565, 567, 568, 572, 575, 576, 578, 629, 633,

641–643, 649, 661, 669, 673, 704, 714, 728, 750, 754, 780
luatex, 122, 127, 129, 130, 133, 134, 291, 424, 455, 498, 523, 555, 661, 673
LyX, 119, 120, 582, 584

M

MacTeX, 87, 131, 276
makeindex, 131, 132, 288–294, 458, 759, 762, 763
Mappa dei caratteri, 98
Marked2, 589
METAPOST, 129, 233, 234, 254, 256, 258, 260, 263
mftrace, 439, 440, 452
MiKTeX, 33, 83, 84, 89–94, 108–111, 131, 140, 293, 435–437, 444, 449, 451, 640, 684, 685, 740
MiKTeX Maintainance, 94
MiKTeX Settings, 94
mktexlsr, 133
mllatex, 133
mltex, 133
mptopdf, 132, 263
MultimarkdownComposer, 589

N

Notepad, 85, 111, 453
Notepad++, 85
Notepade++, 85

O

Okular, 140
Omega, 129
Open XML converter, 584
OpenOffice, 591, 627
OpenOffice.org, 583
otfinfo
  -T, 430
  -V, 431
  -a, 429
  -f, 429
  -g, 430
  -h, 431
  -i, 429
  -p, 429
  -q, 431
  -s, 429
  -t, 430
  -u, 430
  -v, 429
  -z, 429
  --script, 430
  --version, 431
otfinfo, 354, 428, 429

P

Pages, 590, 591
Paint, 263
Pandoc, 571, 588
patgen, 643, 644
pdfcrop, 133, 265
pdfelatex, 125
pdfetex, 122, 125
pdfimport, 583
pdflatex, 35, 87, 107, 113, 117, 118, 120, 121, 123, 124, 128, 130, 133, 137, 174, 230, 231, 234, 256, 259–262, 266, 291, 351–353, 355, 356, 358–360, 362, 363, 366, 368, 377, 378, 381, 382, 385, 397, 412, 413, 416, 418–420, 423, 425, 428, 431, 432, 439, 441, 445, 450, 451, 466, 471, 508, 556, 561, 563–565, 567, 568, 572, 577, 578, 580–582, 585, 629, 643, 649, 652, 656–658, 660, 669, 704, 712, 714, 725, 726, 750, 752, 754, 759

pdftex, 122–127, 129, 133, 259, 260, 291, 293, 382, 385, 411, 425, 441, 455, 498, 522, 555, 560, 566, 673–677, 679, 680, 731, 736, 737, 739, 782
pdftops, 263
pdfx, 564
perl, 454
potrace, 439
Preflight, 557
Preview, 96, 265, 583
ProTEXt, 83, 110
ps2pdf, 97, 120, 259, 262, 380
ps4pdf, 132, 259, 261
pstopdf, 132
ptex, 133
python, 439, 454

Q

QuarkXPress, 42

R

rtf2latex2e, 579

S

Saxon, 585
Scribus, 41, 42
Scrivener, 24, 32, 589–593, 596
Skim, 96, 117, 118, 583
StarOffice, 583
SumatraPDF, 84, 107, 108, 140

T

Task Manager, 685
tex, 122, 124, 126, 129, 133, 259, 292, 423, 455, 641, 673, 736, 782
TeX Live Manager, 94
TeX Live Utility, 87
TeX4ht, 571
texdoc, 90, 134, 542, 782
TEX Live, 33, 34, 81, 82, 84, 86, 87, 89, 91, 93–95, 108, 109, 112, 113, 123, 154, 260–262, 283, 290, 293, 343, 355, 361, 363, 408, 417, 435–437, 442–444, 449, 450, 543, 568, 640, 642–644, 685, 714, 724, 782
TeXmacs, 119
Texmaker, 86, 108, 110, 114, 119, 652, 671
TeXnicCenter, 109–111
Texpad, 596
TeXShop, 87, 105, 108, 109, 114–119, 126, 140, 151, 259, 263, 364, 596, 652, 658, 662, 663, 669–671, 685, 686, 688
TeXstudio, 84, 86, 107, 108, 110, 111, 114, 119, 151, 276, 652, 663, 671
TeXstudio, 364
TextEdit, 584, 591
TextPad, 111
TeXworks, 86, 87, 108–111, 114, 117–119, 364, 596, 652, 658, 662–664, 668–671, 685
tlmgr, 83, 86, 94
ttf2tfm, 441

U

uninstl.bat, 82
updmap, 133, 436, 442, 560
updmap-sys, 133, 442, 443, 560

V

VeraPDF, 557, 563, 568, 570
vim, 112, 453
vptovf, 440

W

Win-iconv, 668
WinCDEmu, 82
WinEdt, 110, 111, 263, 576
Word, 590, 591
Word2LyXMacro, 582
word2tex, 579
Wordpad, 85
Writer, 581, 582, 584, 590, 591, 647
Writer2LaTeX, 581, 584
Writer4LaTeX, 581

X

xdvi, 116, 425, 451
xdvipdfmx, 132, 234
xelatex, 18, 35, 67, 117, 120, 121, 123, 124, 126, 128, 130, 132–134, 137, 162, 174, 230, 234, 256, 260, 262, 293, 352, 354, 357, 359, 362, 363, 365, 366, 368, 378, 381, 382, 385, 397, 398, 400, 411, 418, 420, 421, 423–426, 428, 432, 434, 466, 566, 572, 575, 629, 641–643, 649, 661, 669, 704, 714, 728, 750, 754
Xemacs, 112, 113
xetex, 122, 128, 133, 134, 291, 455, 498, 555, 566, 661, 673
xfig, 242
xindy, 131, 293
xpdf, 129

Y

YAP, 425, 451

Z

Zotero, 284