I calcoli matematici in pdftex

Claudio Beccari

GUIT *meeting* 2015

TEX e i calcoli

È ovvio che TEX è nato sapendo fare qualche calcolo adatto all'uso in tipografia.

I suoi comandi nativi sono:

Comandi nativi

\advance per eseguire una somma algebrica
\multiply per eseguire una moltiplicazione
\divide per eseguire una divisione intera

I conti venivano e vengono tuttora fatti solo con numeri interi.

Le lunghezze

Le lunghezze di TEX sono gestite e memorizzate come un numero intero di scaled point (unità **sp**).
Uno scaled point è la frazione $1/2^{16}$ di un punto tipografico; questo a sua volta è la frazione $1/72.27$ di un pollice, che a sua volta contiene esattamente 25.4 mm. Ne segue che un punto tipografico equivale a 0.35146 mm, poco più di un terzo di millimetro. Uno scaled point, quindi vale 5.36285 nm, molto meno della più corta lunghezza d'onda visibile dall'occhio umano.

Registri del calcolatore

TEX conserva i suoi dati numerici in registri di 4 byte, quindi in registri di 32 bit. Due bit sono riservati al segno e, per le lunghezze, a impostazioni speciali; quindi solo 30 bit possono indicare il numero di scaled point di una lunghezza, il cui valore massimo è quindi di $2^{30} - 1 = 1\,073\,741\,823$ scaled point, un numero enorme, ma di cose piccolissime; tradotto in punti tipografici diventa 16 383.999 98 pt, e tradotto in metri diventa circa 5.758 m, poco meno di sei metri. Per gli usi tipografici sembra che tutto sia adeguato.

E i numeri fratti?

TEX traduce ogni misura fratta di lunghezza in un numero intero di scaled point, e non c'è verso di eseguire calcoli con numeri fratti. Volendo si possono immaginare le lunghezze espresse in scaled point come se fossero espresse in punti tipografici con 16 cifre binarie fratte. Il comando nativo **\the** permette di rappresentare in forma decimale fratta il valore di una lunghezza in punti: per esempio qui si può scrivere con **\the\paperwidth** la larghezza di questo quadro: 364.19536pt. In realtà essa è scritta internamente come:

0001 0110 1100 0011 0010 0000 0011

di cui gli ultimi 16 bit (gli ultimi 4 gruppi di 4 bit) rappresentano la parte binaria fratta del numero di punti tipografici.

Operazioni con numeri fratti

TEX, dunque, non fa operazioni con numeri fratti.
È in grado di trasformare una indicazione di lunghezza con un valore fratto della sua misura in un numero intero di scaled point, e di convertire un numero di scaled point una una stringa decimale fratta seguita dall'unità di misura 'pt'.
Ma non è in grado in nessun modo di eseguire direttamente calcoli sui numeri fratti.

Vecchie proposte e recenti estensioni

Già nel 1997 avevo scritto un articolo su *TUGboat* (18:1) per lamentare l'assenza da ogni motore di composizione di un'aritmetica *floating point*.
Nulla è stato fatto da allora direttamente nei motori di composizione se non queste due importanti operazioni.

1. Lo sviluppo del New Typesetting System (NTS), che ha fruttato l'importante estensione ε-TEX incorporata da una decina d'anni in tutti i motori di composizione allora esistenti.
2. Lo sviluppo di *luatex* che è andato ben al di là delle mie proposte.

Pacchetti per calcolare con numeri fratti

Indirettamente però si può aggirare questa mancanza con alcuni pacchetti.
Il pacchetto *calc* tradizionalmente permette di aggirare l'ostacolo e molti altri pacchetti si appoggiano su *calc*.
Il pacchetto *fp* permette di fare calcoli con numeri fratti indicati anche in notazione scientifica (numero ed esponente: per esempio: 364.19536 viene scritto nella forma 3.6419536×10^2, e codificato nella forma `3.6419536E02`).
Esegue i calcoli nella rappresentazione decimale e allunga i tempi di calcolo in modo molto significativo. I pacchetti *tikz* e *pgfplots* si servono di questi algoritmi e chi ha usato *pgfplots* ne ha constatato la lentezza di elaborazione.

Il linguaggio L3

Il linguaggio del futuro LaTeX 3 è in via di sviluppo, ma già contiene una libreria *l3fp* che è in grado di elaborare numeri fratti in notazione scientifica, ma non tutti sono in grado di usare questa libreria.

Non ho sufficiente esperienza, ma ho il sospetto che la lentezza dei calcoli sia solo marginalmente ridotta rispetto all'esecuzione eseguita con le macro definite nel pacchetto *fp*.

Le estensioni di ε-TEX

Da quasi 10 anni le estensioni del progetto ε-TEX sono incorporate in ogni motore di composizione del sistema TEX, tranne nella versione originale di *tex*.

Praticamente *tex* non è più usato da nessuno (tranne che dal suo creatore Knuth) ma continua a rappresentare la pietra di paragone per sapere se un dato motore può contenere la stringa **tex** nel suo nome.

Le estensioni di ε-TEX

Fra le tante cose utili introdotte da ε-TEX, ci sono le espressioni in particolare quelle numeriche e quelle dimensionali (sulle lunghezze).

Espressioni numeriche e dimensionali

\numexpr esegue calcoli sui numeri interi
\dimexpr esegue calcoli sulle lunghezze

Una espressione dimensionale può contenere una espressione numerica.

Le espressioni

Ci sono alcune particolarità nei calcoli eseguiti con queste espressioni.
I comandi **\numexpr** e **\dimexpr** iniziano una espressione numerica o dimensionale che termina con il primo token non valido in una espressione; conviene terminare ogni espressione con il comando **\relax**.

Ogni espressione può contenere i simboli + per la somma, - per la sottrazione, * per la moltiplicazione e / per la divisione.

Ogni espressione può contenere sottoespressioni racchiuse fra parentesi tonde.

Gli arrotondamenti

Le divisioni fra numeri interi continuano a produrre solo la parte intera del quoziente, ma arrotondandola all'intero più vicino, non troncandola al valore intero immediatamente inferiore in modulo rispetto al quoziente fratto.

Per cui la divisione 6/7 eseguita con il comando **\divide** produce il valore nullo, mentre la stessa divisione eseguita con **\numexpr** porta al valore 1, visto che il valore esatto sarebbe 0.857.
L'arrotondamento avviene anche per le divisioni fra lunghezze.

Gli scalamenti

Uno scalamento in questo contesto indica un fattore di scala espresso mediante una frazione; quindi per scalare b con il fattore di scala n/d al fine di ottenere a bisogna eseguire il calcolo

$$a = b\frac{n}{d}$$

In termini di ε-TEX, se i simboli **\dima**, **\dimb**, **\dimn** e **\dimd** sono i nomi di registri dimensionali che contengono gli operandi, allora l'espressione suddetta con la sintassi nativa per le assegnazioni di valori alle lunghezze, si traduce in:

Esempio

```
\dima = \dimexpr \dimb * \dimn / \dimd \relax
```

Gli scalamenti

Ecco: quella sequenza di una moltiplicazione e di una divisione si chiama scalamento ed ε-TEX la esegue in un modo particolare.

1. Prima esegue la moltiplicazione senza troncare il risultato e quindi conservando questo risultato intermedio in un registro di 64 bit (virtualmente con 32 cifre binarie fratte).

Gli scalamenti

Ecco: quella sequenza di una moltiplicazione e di una divisione si chiama scalamento ed ε-TEX la esegue in un modo particolare.

1. Prima esegue la moltiplicazione senza troncare il risultato e quindi conservando questo risultato intermedio in un registro di 64 bit (virtualmente con 32 cifre binarie fratte).
2. Poi esegue la divisione per un divisore (virtualmente con 16 cifre binarie fratte); il risultato di questa divisione viene assegnato ad una parola di 32 bit; se esso è una lunghezza valida la assegna alla dimensione a primo membro, oppure se la parte che scarta non è nulla alza un flag d'errore senza interrompere l'elaborazione ma assegnando al primo membro il massimo valore consentito per una lunghezza.

Esempio di scalamento

Supponiamo di disporre della lunghezza `\textwidth` e di voler costruire la gabbia del testo con le stesse proporzioni della pagina. Allora deve essere:

$$\frac{\texttt{\textheight}}{\texttt{\textwidth}} = \frac{\texttt{\paperheight}}{\texttt{\paperwidth}}$$

Quindi dobbiamo calcolare:

$$\texttt{\textheight} = \texttt{\textwidth} \cdot \frac{\texttt{\paperheight}}{\texttt{\paperwidth}}$$

Questa è una tipica operazione di scalamento.

Esempio di scalamento

Se in questi quadri la larghezza del testo fosse di 108 mm, allora il codice seguente ci permetterebbe di determinare l'altezza della gabbia:

```
\textheight= \dimexpr \textwidth *
\paperheight / \paperwidth \relax
```

Infatti il risultato sarebbe

```
\textheight = 230.46748pt   (= 81.00003mm)
```

Come si vede i decimali ci sono tutti insieme alle unità di misura di default 'pt'.
(Il valore in millimetri è stato calcolato dal programma stesso.)

Come impostare i calcoli

Quanto esposto finora ci permette di dire che possiamo trasformare qualunque numero fratto senza troppi decimali in una lunghezza che abbia proprio quel valore come misura in punti. Su questa ed altre simili lunghezze possiamo fare tutti i calcoli usando le quattro operazioni elementari; alla fine possiamo ottenere una lunghezza la cui misura in punti è il risultato fratto che volevamo ottenere. Al massimo dobbiamo togliere l'unità di misura 'pt' che generalmente non ci interessa.

Il comando per "strappare via" i 'pt' è, senza troppe sorprese, proprio `\strip@pt` già presente nel nucleo di LaTeX.

Trasformare le unità di misura

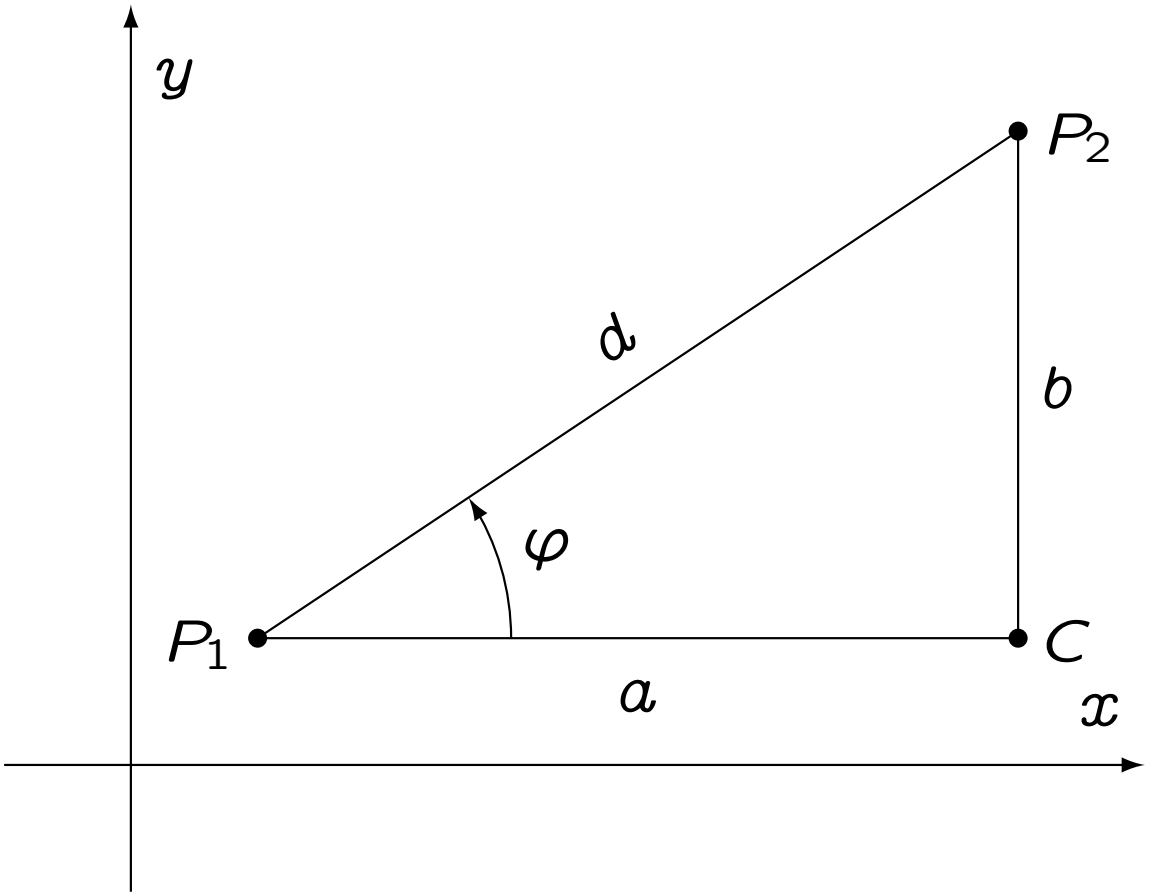
Ecco un piccolissimo esempio. La larghezza in punti di queste griglie non è di facile comprensione. È vero che sarebbe molto facile dividere a mente per 3 la misura in punti: 307.28987pt, stimando un valore di 102 mm. Ma se vogliamo qualcosa di più preciso possiamo usare un'espressione di scalamento:

```
\newcommand*\changepttomm[1]{\bgroup
\dimendef\lungh= 2222 \lungh=#1\relax
\dimendef\pttoin=2223 \pttoin=72.27\p@
\dimendef\mmtoin=2224 \mmtoin=25.4\p@
\strip@pt\dimexpr\lungh*\mmtoin/\pttoin\relax\,mm%
\egroup}
```

e il risultato è: \textwidth=108 mm.

La distanza pitagorica

Siano dati i punti P_1 e P_2 come in figura.



La distanza pitagorica è la lunghezza d del segmento P_1P_2.

Impostazione dei calcoli

Indipendentemente dalla figura è banale calcolare la differenza assoluta delle ascisse a e quella delle ordinate b. Supponiamo che sia $a \geq b$.
Il teorema di Pitagora ci dice che è

$$d = \sqrt{a^2 + b^2}$$

Per fare i conti conviene mettere in evidenza a; posto dunque $x = b/a$, è sempre $0 \leq x \leq 1$. Dobbiamo calcolare allora

$$d = a\sqrt{1 + x^2}$$

Il metodo iterativo di Newton

Nemmeno le espressioni di ε-TEX riescono a calcolare una radice quadrata.
Ma Newton ci ha fornito un ottimo metodo iterativo che si basa sul calcolo della media fra un valore approssimato della radice e il rapporto fra il radicando e quel valore approssimato; se tale valore è approssimato per difetto il rapporto è in eccesso rispetto al valore "esatto" della radice. La loro media è dunque più vicina la valore esatto.
Posto $R = 1 + x^2$ ed r_0 un valore approssimato iniziale, si tratta di eseguire la sequenza di calcoli seguente:

$$r_{i+1} = 0.5(r_i + R/r_i) \qquad \text{per } i = 0, 1, 2, \ldots$$

La divisione di due numeri fratti

Se gli operandi sono già stati assegnati a certi registri dimensionali, la divisione può essere eseguita come uno scalamento dove uno dei due operandi da moltiplicare vale la lunghezza unitaria `\p@`.
Inoltre, visto che la nostra macchina calcolatrice è piuttosto modesta e non ci può dare più di 16 cifre binarie fratte, corrispondenti a meno di 5 cifre decimali fratte, è inutile spingere le iterazioni oltre un certo limite. Poiché l'algoritmo ha convergenza quadratica[1] poche iterazioni sono più che sufficienti.

[1] le distanze fra i successivi valori approssimati e il valore teorico esatto va diminuendo con legge quadratica almeno finché il "rumore" degli arrotondamenti alla quinta cifra decimale non impedisce ulteriori miglioramenti dell'approssimazione

La sequenza per il calcolo della radice di 2

Supponiamo che sia $a = b$; la radice da calcolare è quindi $\sqrt{R} = \sqrt{2}$. Partendo dal valore iniziale $r_0 = 1$ la sequenza delle prime iterazioni risulta essere la seguente.

i	r_i	$\lvert r_\infty - r_i \rvert$
0	1.00000	0.4142135
1	1.50000	0.0857864
2	1.41667	0.0024564
3	1.41422	0.0000064
4	1.41421	0.0000035

Un paio di altre iterazioni ci metterebbero dalla parte della ragione, ma è evidente che se proseguissimo l'algoritmo iterativo continueremmo all'infinito con l'alternanza degli ultimi due valori visto che TEX non ci può dare di più.

Le funzioni trigonometriche

Ci sono già alcuni pacchetti di uso comune che forniscono i valori delle funzioni trigonometriche con l'angolo indicato in radianti.
Siccome un angolo giro contiene un numero irrazionale di radianti, mentre contiene un numero intero di gradi, la riduzione dell'angolo al suo intervallo principale $-180° < \theta \leq +180°$ è più precisa da eseguire se si usano i gradi.

Non solo, ma la tangente di un angolo è particolarmente facile da calcolare usando le frazioni continue.

Le formule parametriche travestite

Normalmente a scuola (secondaria superiore) si imparano le formule parametriche nella forma:

$$\sin\theta = \frac{2\tan\theta/2}{1+\tan^2\theta/2}$$
$$\cos\theta = \frac{1-\tan^2\theta/2}{1+\tan^2\theta/2}$$
$$\tan\theta = \frac{2\tan\theta/2}{1-\tan^2\theta/2}$$

Le formule parametriche travestite

Quelle formule sono meno adatte al calcolo con i mezzi di ε-TEX rispetto alle seguenti dove si è posto $x = \pi\theta/360 = \theta/114.591556$ che rappresenta il valore in radianti dell'angolo $\theta/2$ espresso in gradi:

$$\sin\theta = \frac{2}{\cot x + \tan x}$$
$$\cos\theta = \frac{\cot x - \tan x}{\cot x + \tan x}$$
$$\tan\theta = \frac{2}{\cot x - \tan x}$$

È chiaro che la riduzione di θ all'intervallo principale viene fatta con i calcoli in gradi e poi vengono usati i radianti nell'espressione finale.

La frazione continua della tangente

Le approssimazioni delle funzioni trascendenti mediante frazioni continue sono particolarmente comode visto che i coefficienti sono tutti numeri interi; inoltre si prestano ad essere programmate con algoritmi iterativi, talvolta con algoritmi ricorsivi.

Per la tangente circolare vale lo sviluppo seguente:

$$\tan x = \cfrac{1}{\cfrac{1}{x} - \cfrac{1}{\cfrac{3}{x} - \cfrac{1}{\cfrac{5}{x} - \cfrac{1}{\cfrac{7}{x} - \cfrac{1}{\cfrac{9}{x} - \ldots}}}}}$$

Il calcolo della frazione continua

Nell'articolo c'è il codice per il cuore del calcolo della frazione continua troncata al termine $11/x$; qui non lo si commenta di nuovo; l'unica cosa che merita sottolineare è come sia semplice quel calcolo e il suo codice.
È tanto semplice e abbastanza preciso, nonostante le 5 cifre decimali scarse che l'algoritmo fornisce, che altrove l'ho ritenuto utile per calcolare iterativamente l'arcotangente cercando, appunto, l'angolo che fornisce la tangente data, piuttosto che usare la frazione continua di Lagrange specifica per l'arcotangente.

Il calcolo dell'arcotangente

L'arcotangente è una delle funzioni della libreria di calcolo del pacchetto *pgfplots*.
Ho esaminato il codice per trarne ispirazione, ma ho deciso di non appoggiarmi a quel pacchetto; è vero che usa il calcolo floating point, preciso e lentissimo, ma si basa sull'interpolazione di una tabella di dati che fornisce l'arcotangente per un migliaio di valori con più di una ventina di cifre significative della variabile indipendente. Ottimo. . . ma lentissimo.

Confesso di non avere provato con la libreria *l3fp* in linguaggio L3. Non mi sento ancora a mio agio nell'usare quel linguaggio.

Naturalmente si potrebbe usare il linguaggio Lua incorporato nel programma di composizione *luatex*. Ma quando si scrivono macro per pacchetti da usare possibilmente in modo indipendente da questo o quel programma di composizione l'uso di funzionalità specifiche per un dato programma diventa impossibile.

Alcune considerazioni finali

Quanto esposto in questa presentazione e nell'articolo è molto utile, direi utilissimo, per chi scrive pacchetti e classi.
È probabile che l'utente "normale" non pensi mai ad usare questa matematica per scrivere i suoi documenti; tuttavia non c'è bisogno di scomodare *calc* per fare i calcoletti direttamente accessibili con le espressioni di ε-TEX.
Se l'utente deve disegnarsi alcuni diagrammi e deve programmarne il loro disegno, certamente può ricorrere ai potentissimi pacchetti *tikz* e *pgfplots*. Ma a seconda del tipo di disegno potrebbe essere applicabile quella frase di "sparare ad una zanzara con un cannone".

Le curve di Bézier

Non voglio annoiarvi con ulteriori dettagli e quindi non parlo delle curve di Bézier di cui parlo nell'articolo.

Ammetto di essermi divertito mentre scrivevo l'articolo provando a vedere quanto in là potevo spingermi con il disegno programmato che ho codificato nel pacchetto *curve2e*.

Presentai la prima versione di *curve2e* al TUG meeting di Marrakesh nel 2006. Simultaneamente veniva distribuito TeXLive con la prima versione di *tikz*. Avrei potuto ritirare il mio pacchetto da CTAN; ma poi ho constatato che diverse persone lo usavano in giro per il mondo. Perciò l'ho conservato e arricchito di nuove funzionalità, ma sempre usando le possibilità di calcolo di *pdftex*.

Happy pdfTeXing