Numero 11
Aprile 2011

ArsTEXnica

Rivista italiana di TEX e LATEX

GUIT

http://www.guit.sssup.it/arstexnica

GUIT – Gruppo Utilizzatori Italiani di TEX

ArsTEXnica è la pubblicazione ufficiale del GUIT

ArsTEXnica è la prima rivista italiana dedicata a TEX, a LATEX ed alla tipografia digitale. Lo scopo che la rivista si prefigge è quello di diventare uno dei principali canali italiani di diffusione di informazioni e conoscenze sul programma ideato quasi trent'anni fa da Donald Knuth.

Le uscite avranno, almeno inizialmente, cadenza semestrale e verranno pubblicate nei mesi di Aprile e Ottobre. In particolare, la seconda uscita dell'anno conterrà gli Atti del Convegno Annuale del GUIT, che si tiene in quel periodo.

La rivista è aperta al contributo di tutti coloro che vogliano partecipare con un proprio articolo. Questo dovrà essere inviato alla redazione di ArsTEXnica, per essere sottoposto alla valutazione di recensori. È necessario che gli autori utilizzino la classe di documento ufficiale della rivista; l'autore troverà raccomandazioni e istruzioni più dettagliate all'interno del file di esempio (`.tex`). Tutto il materiale è reperibile all'indirizzo web della rivista.

Gli articoli potranno trattare di qualsiasi argomento inerente al mondo di TEX e LATEX e non dovranno necessariamente essere indirizzati ad un pubblico esperto. In particolare tutorials, rassegne e analisi comparate di pacchetti di uso comune, studi di applicazioni reali, saranno bene accetti, così come articoli riguardanti l'interazione con altre tecnologie correlate.

Di volta in volta verrà fissato, e reso pubblico sulla pagina web, un termine di scadenza per la presentazione degli articoli da pubblicare nel numero in preparazione della rivista. Tuttavia gli articoli potranno essere inviati in qualsiasi momento e troveranno collocazione, eventualmente, nei numeri seguenti.

Chiunque, poi, volesse collaborare con la rivista a qualsiasi titolo (recensore, revisore di bozze, grafico, etc.) può contattare la redazione all'indirizzo:

`arstexnica@sssup.it`.

Associarsi a GUIT

Fornire il tuo contributo a quest'iniziativa come membro, e non solo come semplice utente, è un presupposto fondamentale per aiutare la diffusione di TEX e LATEX anche nel nostro paese. L'adesione al Gruppo prevede una quota di iscrizione annuale diversificata: 30,00 € soci ordinari, 20,00 (12,00) € studenti (junior), 75,00 € Enti e Istituzioni.

Indirizzi

Gruppo Utilizzatori Italiani di TEX:
c/o Ufficio Statistica
Scuola Superiore Sant'Anna
Piazza Martiri della Libertà 33
56127 Pisa, Italia.
`http://www.guit.sssup.it`
`guit@sssup.it`

Redazione ArsTEXnica:
`http://www.guit.sssup.it/arstexnica/`
`arstexnica@sssup.it`

Codice ISSN 1828-2369

Stampata in Italia
Pisa: 15 Aprile 2011

ArsTEXnica

Rivista italiana di TEX e LATEX

Numero 11, Aprile 2011

Gruppo Utilizzatori Italiani di TEX

Editoriale

Gianluca Pignalberi

Non ho fatto in tempo ad accomiatarmi nell'ultimo editoriale che mi ritrovo qui, voluto dal GUIT, di nuovo alla direzione di ArsTEXnica (con in più la carica di vicepresidente dell'associazione — verrò denunciato per accumulazione di cariche —), perciò eccomi qui a riaprire dal punto lasciato lo scorso anno: i ritardi.

Mi scuso spiegandovi perché avete ricevuto il numero di aprile 2011 così in ritardo. I contributi sono stati inizialmente un po' pochi per cui ho posticipato informalmente la *deadline*. Oltre a fare i solleciti di rito, ho deciso di scrivere un articolo in gestazione da lungo tempo chiedendo la complicità di Jerónimo Leal e di Salvatore Schirone. Purtroppo sono stato uno dei ritardatari nel consegnare il materiale, insieme alla traduzione dell'articolo di Jean-Michel Hufflen. Sia io che Massimiliano Dominici (il traduttore dell'articolo nominato) siamo stati impegnati su varii fronti di TEX, non ultimo l'avventura con la CompoMat di Silvia Maschio per la collana TEXNOLOGIE, a breve nelle librerie. Un altro contributo lungamente atteso, ma non più arrivato, ha contribuito al ritardo nell'uscita. Me ne scuso personalmente con tutti gli associati.

Prima di passare a parlare di quanto succede nel GUIT, mi rammarico di non poter pubblicare alcun articolo di Enrico Gregorio, nostro nuovo presidente. Questo numero di ArsTEXnica cammina in pratica su una gamba sola (quella di Claudio Beccari), ben supportata dalle stampelle di tutti gli autori (tra cui due studenti universitari) che hanno presentato dei lavori più che pregevoli.

Cosa fa intanto il consiglio direttivo del GUIT? Per prima cosa abbiamo formalizzato il passaggio di consegne tra il vecchio e il nuovo direttivo, cosa fondamentale per la corretta gestione dell'associazione. Il fatto che proprio il giorno che abbiamo raggiunto Pisa da varie parti d'Italia Poste Italiane abbia cambiato tutta la modulistica all'insaputa degli impiegati è stata la proverbiale "ciliegina sulla tor..." no, troppo banale, l'arancia candita sul cannolo siciliano. Ma arriveremo al risultato voluto.

Su segnalazione di alcuni associati abbiamo chiesto e ottenuto che ai mirror da cui scaricare TEX Live si aggiungesse il GARR. Ciò ha comportato uno scambio di informazioni tra noi, il TUG e il GARR, cosa risolta in pochissimi giorni. Ho trovato il mirror del GARR come mirror predefinito quando poco tempo dopo ho installato TEX Live 2011. Non posso testimoniare sull'aumentata velocità perché ho lasciato lavorare il computer impresenziato, e comunque non ricorderei il tempo occorso per installare TEX Live 2010.

Il nuovo sito sta prendendo vita: una parte è già stata fatta, altre parti verranno aggiunte nel tempo (sempre di lavoro volontario si tratta), e potremo lasciare il server della Sant'Anna gentilmente concessoci per tanti anni.

Dovremmo iniziare una specie di collaborazione con il Prof. Manuel Gonzalez Suarez dell'università di Oviedo (Spagna). Costui ci ha chiesto di avere i sorgenti di alcune delle opere scritte da alcuni associati del GUIT e rilasciate tramite il nostro sito così da poterle tradurre in spagnolo e riprodurre nel loro formato originario. Quest'opera meritoria è un riconoscimento al lavoro del GUIT e di quanti hanno scritto direttamente o collaborato in varie forme alle opere richieste.

Passiamo ora agli articoli pubblicati su questo numero.

Antonello Pilu ci spiega in *La creazione di una prova d'esame* l'uso di LATEX per ottenere delle prove d'esame di qualità ineccepibile. Ci mostra diversi brani di codice per diverse esigenze e stili di test.

Gianluca Pignalberi, Salvatore Schirone e Jerónimo Leal ci propongono un'*Introduzione agli strumenti per grecisti classici* e ci introducono all'uso di compositori, editor e pacchetti utili alla loro opera tutt'altro che semplice.

Marco Crivellaro, studente universitario, ci racconta la sua esperienza nell'integrazione tra R e LATEX per la stesura di report statistici di alta qualità. *Leggetelo in Un dialogo tra GNU-R e LaTeX: xtable e Sweave.*

Un altro studente universitario, Giovanni Mascellani, ci presenta *Uno strumento per gestire progetti LATEX cooperativi*. Usa Git per realizzare un portale per la scrittura cooperativa di appunti (LATEX è ovviamente il compilatore degli appunti).

In *Passare da LATEX a XSL-FO* Jean-Michel Hufflen (con l'aiuto di Massimiliano Dominici in qualità di traduttore) introduce i lettori a XSL-FO. Mostra i rudimenti di questo linguaggio e ne descrive affinità e differenze con LATEX.

La virgola intelligente di Claudio Beccari, articolo breve ma densissimo, spiega come risolvere il problema della spaziatura dopo il separatore decimale se questo è diverso dal punto.

Con *The unknown* picture *environment*, sempre Claudio Beccari ci mostra degli aspetti insoliti e delle particolarità dell'ambiente picture, rendendoci edotti del fatto che questo pacchetto può rivelarsi

insostituibile nonostante l'esistenza di pacchetti più moderni e gettonati.

Infine Luigi Scarso ci presenta *Extending ConTEXt MkIV with PARI/GP* e ci fa capire come integrare ConTEXt con il citato sistema di computer algebra. Propone un nutrito numero di esempi per rendere la comprensione più profonda.

Buona lettura e happy TEXing.

▷ Gianluca Pignalberi
`g dot pignalberi at alice dot it`

La creazione di una prova d'esame

Antonello Pilu

Sommario

Questo articolo mostrerà come realizzare in modo facile e veloce una prova d'esame con LaTeX 2ε, utilizzando lo specifico pacchetto *exam*. In tale ambiente si potranno predisporre dalle più semplici alle più complesse batterie di domande, con la consueta qualità tipografica propria di LaTeX 2ε.

Abstract

This article will show us how to realise, in a fast and simple way, a test with LaTeX 2ε, using its *exam* package.

With this package we will be able to elaborate many different tests, from the easiest to the more complexe ones, always keeping the high graphical quality of LaTeX 2ε.

1 Introduzione

La predisposizione di test, questionari, prove d'esame è molto importante nella scuola superiore, all'università e nelle aziende. La scelta della tipologia di domande può essere molto varia. Si potrebbero utilizzare quesiti a risposta aperta, a risposta chiusa, vero o falso, ecc.

La stesura di una prova di verifica non banale, se effettuata con gli usuali programmi di videoscrittura, potrebbe implicare difficoltà di un certo rilievo e un grande utilizzo di tempo. Tale difficoltà è sicuramente accentuata se la prova riguarda discipline di tipo scientifico o matematico, nelle quali sia necessario inserire molte formule matematiche.

Con LaTeX 2ε e l'utilizzo della classe *exam* il lavoro di preparazione della prova può essere semplificato ed il tempo per la sua predisposizione ridotto in modo significativo. Oltre a ciò, comunque, la qualità del lavoro prodotto è elevata, rientrando negli standard qualitativi delle produzioni LaTeX 2ε.

Il file *exam.cls* fornisce la classe documento *exam*, la quale consente di semplificare la preparazione di un questionario d'esame anche a novizi di LaTeX 2ε. La maggior parte di quello che si può fare con la classe *exam*, in verità, può essere svolto utilizzando altre classi quali *fancyheadings* e *fullpage*, aggiustando l'aspetto della pagina LaTeX; tuttavia, la classe *exam* rende il lavoro molto più semplice.

Per esempio, la classe *exam* imposta il layout della pagina in modo tale da lasciare un margine fisso (di un pollice) tutto intorno alla pagina (indipendentemente dalla dimensione della pagina) e fornisce i comandi che semplificano la formattazione delle domande, la creazione flessibile degli header e dei footer ed il cambio dei margini. In particolare:

- La classe formatta e numera automaticamente le domande, i gruppi di domande, le sottodomande e rende semplice riferirsi a specifiche domande attraverso il loro numero, secondo le esigenze dell'autore del questionario.
- Si può includere il punteggio di ogni domanda (o gruppo, o sottogruppo), scegliendo se visualizzare tale valore all'inizio del testo della domanda, sul lato sinistro del foglio o sul lato destro, all'inizio o alla fine della domanda.
- La classe aggiunge il totale del punteggio di ogni domanda o gruppo di domande, oppure il totale per pagina. Alla fine del questionario si può stampare una tabella riassuntiva dei punteggi raggruppati per domanda o per pagina, a seconda delle necessità.
- Si può specificare l'intestazione suddivisa in tre parti: l'intestazione sinistra, quella centrale e quella destra. L'intestazione sinistra è giustificata a sinistra; l'intestazione centrale è giustificata centralmente, mentre l'intestazione destra è giustificata a destra. Una o più intestazioni possono essere omesse.
- Anche il piè di pagina è suddiviso in tre parti, le cui giustificazioni seguono le regole dell'intestazione.
- L'intestazione ed il piè di pagina della prima pagina possono essere diversi da quelli delle pagine successive.
- Sia l'intestazione che il piè di pagina possono contenere più di una riga. La gestione delle varie righe è fatta con semplici comandi, attraverso i quali è possibile modificare la dimensione di ogni singola parte.
- Le macro sono definite in modo da consentire di disporre nell'intestazione o nel piè di pagina del numero totale di pagine del questionario e di cambiare quanto compare nell'ultima pagina.
- Vi sono macro che consentono di indicare nell'intestazione o nel piè di pagina che una domanda è iniziata nella pagina precedente, o continua nella pagina successiva.

- È possibile disegnare una linea alla base dell'intestazione o sopra il piè di pagina.

2 Il comando `documentclass`

Per utilizzare la classe documento *exam* si deve specificare `exam` come classe documento principale. Ad esempio, se si vuole utilizzare caratteri di 12 punti, il `documentclass` sarà:

```
\documentclass[12pt]{exam}
```

Per poter usufruire delle caratteristiche di AMSLaTeXsi devono utilizzare i comandi seguenti:

```
\documentclass[12pt]{exam}
\usepackage{amsmath}
```

Se sono state inserite le soluzioni all'interno del questionario e si vuole stamparle, bisogna includere nella classe documento l'opzione *answer*, come in:

```
\documentclass[answer]{exam}
```

oppure, come nella riga seguente:

```
\documentclass[answer,12pt]{exam}
```

Tuttavia si tratta di una scelta opzionale; infatti le soluzioni possono essere stampate ugualmente con il comando:

```
\printanswers
```

3 Richiesta del nome dello studente

Benché non sia qualcosa di specifico della classe `exam`, vale la pena menzionarlo per via della sua non ovvietà. Se si sta lasciando spazio per le risposte nelle pagine del questionario, probabilmente si vuole lasciare anche lo spazio per il nome dello studente. Scrivendo:

```
\begin{center}
\fbox{\fbox{\parbox{5.5in}{\centering
Rispondete alle domande del questionario.
Se lo spazio è insufficiente, allora
continuate nel retro del foglio.}}}
\end{center}
\vspace{0.1in}
\hbox to \textwidth{Studente:
\enspace\hrulefill}
\vspace{0.2in}
\hbox to \textwidth{Classe e sezione:
\enspace\hrulefill}
```

dopo il comando `\begin{document}` e prima del comando `\begin{questions}`, si ottiene il seguente frammento di documento:

> Rispondete alle domande del questionario. Se lo spazio è insufficiente, allora continuate nel retro del foglio.
>
> Studente: _____
>
> Classe e sezione: _____

4 Le domande del questionario

Per inserire le domande nel questionario si utilizza l'ambiente `questions`. Ogni domanda, allora, inizierà con il comando `\question`, e la domanda sarà numerata in automatico.

Per esempio, se si scrive

```
\begin{questions}
\question
Cos'è l'insieme vuoto?
\question
Dato un insieme avente N oggetti, qual è
la cardinalità dell'insieme delle parti?
\question
Calcola
$\displaystyle\int_0^1 x^2 \, dx$.
\end{questions}
```

si ottiene il seguente pezzo di questionario:

1. Cos'è l'insieme vuoto?
2. Dato un insieme avente N oggetti, qual è la cardinalità dell'insieme delle parti?
3. Calcola $\displaystyle\int_0^1 x^2\,dx$.

Per facilitare la lettura del sorgente (file *.tex), si possono lasciare linee vuote tra il comando `\question` e l'inizio della domanda, o prima del primo comando `\question` nell'ambiente in quanto esse saranno ignorate.

5 La scomposizione delle domande

Talvolta è utile suddividere una domanda in parti. In tal caso si deve utilizzare l'ambiente `parts`. Per esempio, il seguente frammento di codice:

```
\begin{questions}
\question
Cos'è l'insieme vuoto?
\question
Sia dato un insieme con N oggetti.
\begin{parts}
\part
Definire cosa si intende con insieme
delle parti.
\part
Determinare la cardinalità dell'insieme
delle parti.
\end{parts}
\question
\begin{parts}
\part
Calcola
```

```
$\displaystyle\int_0^1 x^2 \, dx$.
\part
Calcola
$\displaystyle\int_0^1 (x^2+1) \, dx$.
\end{parts}
\end{questions}
```

produce:

1. Cos'è l'insieme vuoto?
2. Sia dato un insieme con N oggetti.
   (a) Definire cosa si intende con insieme delle parti.
   (b) Determinare la cardinalità dell'insieme delle parti.
3. (a) Calcola $\displaystyle\int_0^1 x^2\,dx$.
   (b) Calcola $\displaystyle\int_0^1 (x^2+1)\,dx$.

Una domanda può essere suddivisa in 4 livelli differenti: `question`, `part`, `subpart` e `subsubpart`.

6 Punteggio delle domande

I comandi `\question`, `\part`, `\subpart` e `\subsubpart` prendono un argomento opzionale, che rappresenta il punteggio di ogni domanda, parte, sottoparte, sotto sottoparte. Per default i punti sono racchiusi tra parentesi tonde, ma esse possono essere sostituite da parentesi quadre o un riquadro.

Per default il punteggio è inserito all'inizio della domanda (o parte), ma può essere messo a sinistra o a destra con i comandi `\pointsinmargin` e `\pointsinrightmargin`.

I due comandi seguenti `\nopointsinmargin` e `\nopointsinrightmargin` sono equivalenti e riportano allo stato di default.

Ognuno dei precedenti comandi stampa il punteggio nella prima riga della domanda (o della parte, subparte, subsubparte). C'é anche la possibilità di stampare il punteggio nell'ultima riga della domanda (o delle sue parti).

Riprendendo l'esempio precedente, per inserire i punteggi si può scrivere:

```
\begin{questions}
\question[1]
Cos'è l'insieme vuoto?
\question[4]
Sia dato un insieme con N oggetti.
\begin{parts}
\part
Definire cosa si intende con insieme
delle parti.
\part
Determinare la cardinalità dell'insieme
delle parti.
\end{parts}
\question
\begin{parts}
\part[15]
Calcola
$\displaystyle\int_0^1 x^2 \, dx$.
\part[15]
Calcola
$\displaystyle\int_0^1 (x^2+1) \, dx$.
\end{parts}
\end{questions}
```

ottenendo:

1. (1 point) Cos'è l'insieme vuoto?
2. (4 points) Sia dato un insieme con N oggetti.
   (a) Definire cosa si intende con insieme delle parti.
   (b) Determinare la cardinalità dell'insieme delle parti.
3. (a) (15 points) Calcola $\displaystyle\int_0^1 x^2\,dx$.
   (b) (15 points) Calcola $\displaystyle\int_0^1 (x^2+1)\,dx$.

È utile provare ad utilizzare i comandi `\pointsinmargin` e `\pointsinrightmargin`, facendoli precedere al codice indicato sopra.

6.1 I mezzi punti

I punteggi delle domande includono anche i mezzi punti. Per specificare un mezzo punto si utilizza il comando `\half`, subito dopo la parte intera, oppure da solo.

Ad esempio:

scrivendo	0	`\half`	1	1`\half`	...
si ottiene	0	½	1	1½	...

6.2 Il contorno dei punteggi

I punteggi possono essere contornati con parentesi tonde, parentesi quadre oppure da una cornice rettangolare. Gli effetti si ottengono con i seguenti comandi:

`\bracketedpoints` racchiude il punteggio tra parentesi quadre;

`\boxedpoints` racchiude il punteggio all'interno di una cornice rettangolare.

Ad esempio, il codice:

```
\bracketedpoints
\begin{questions}
 \question[1]
 Cos'è l'insieme vuoto?
\end{questions}
```

produce:

1. [1 point] Cos'è l'insieme vuoto?

mentre il codice:

```
\boxedpoints
\begin{questions}
 \question[1]
 Cos'è l'insieme vuoto?
\end{questions}
```

produce:

1. 1 point Cos'è l'insieme vuoto?

Come si vede, la differenza consiste nel differente contorno del punteggio della domanda.

6.3 Sostituzione della parola point

In modo predefinito, il punteggio di una domanda è inserito all'inizio del testo, seguito dalla parola *point* se il numero è 1, oppure seguito dalla parola *points* negli altri casi. Si può personalizzare il testo mostrato dopo il numero del punteggio utilizzando il comando

```
\pointpoints{Singolare}{Plurale}
```

Il comando sostituisce a *point* quanto scritto al posto di Singolare e a *points* il testo messo al posto di Plurale.

Ad esempio, usualmente per un questionario in italiano, si scriverà:

```
\pointpoints{punto}{punti}
```

Se non si ha la necessità di differenziare tra singolare e plurale, si può utilizzare il comando:

```
\pointname{Testo}
```

Facciamo due esempi. Il codice:

```
\pointname{\%}
\begin{questions}
 \question[25]
 Cos'è l'insieme vuoto?
\end{questions}
```

produce:

1. (25%) Cos'è l'insieme vuoto?

mentre il codice:

```
\pointpoints{punto}{punti}
\begin{questions}
 \question[25]
 Cos'è l'insieme vuoto?
\end{questions}
```

produce:

1. (25 punti) Cos'è l'insieme vuoto?

7 Numero di domande e somma dei punteggi

L'ambiente `exam`, automaticamente, conta il numero di domande, di parti, di sottoparti e di sotto sottoparti rendendo tali numeri disponibili attraverso le seguenti macro:

```
\numquestions
\numparts
\numsubparts
\numsubsubparts
```

Invece, il comando `\addpoints` calcola la somma tutti i punteggi delle singole domande del questionario (o sue sottoparti), rendendola disponibile con la macro `\numpoints`.

Il comando `\addpoints` va scritto dopo `\documentclass`, ma prima di `begin{document}`. In questo modo, si può scrivere il seguente frammento di codice:

```
\begin{center}
Il questionario è formato
da \numquestions\ domande,
per un totale di \numpoints\
punti.
\end{center}
```

dopo il comando `\begin{document}`, ottenendo quanto segue:

```
Il questionario è formato da 3 domande,
per un totale di 35 punti.
```

7.1 Riferimenti al numero di specifiche domande (riferimenti incrociati)

Nell'ambiente *exam* si possono utilizzare i comandi standard di LaTeX, `\label` e `\ref` per riferirsi a domande (o parti, o sottoparti o sotto sottoparti) attraverso il numero. Ad esempio, se si scrive:

```
La domanda numero \ref{dom:a} è
facilissima. La domanda
\ref{dom:b} è facile.
La domanda \ref{dom:c} è formata
da due parti (parte \ref{part:a}
e parte \ref{part:b}).

\begin{questions}
 \question[1]
  \label{dom:a}
  Cos'è l'insieme vuoto?
 \question[4]
  \label{dom:b}
  Sia dato un insieme con N oggetti.
  \begin{parts}
   \part
    Definire cosa si intende con
    insieme delle parti.
   \part
    Determinare la cardinalità
```

```
    dell'insieme delle parti.
  \end{parts}
 \question
  \label{dom:c}
  \begin{parts}
   \part[15]
    \label{part:a}
    Calcola
    $\displaystyle\int_0^1 x^2\, dx$.
   \part[15]
    \label{part:b}
    Calcola
    $\displaystyle\int_0^1 (x^2+1)\, dx$.
  \end{parts}
\end{questions}
```

nel questionario, prima delle domande, comparirà il seguente testo:

La domanda numero 1 è facilissima. La domanda 2 è facile. La domanda 3 è formata da due parti (parte a e parte b).

Com'è noto, affinché LaTeX 2ε abbia la possibilità di mettere i riferimenti corretti è bene *compilare* due volte il file.

7.2 Inserimento di istruzioni aggiuntive per un gruppo di domande

Ci sono due appositi comandi utilizzati per inserire speciali istruzioni per specifiche domande: `\uplevel` e `\fullwidth`. Questi comandi consentono di dare istruzioni che saranno impostate con una particolare indentazione a sinistra.

Per esempio, all'interno di un ambiente `parts` si possono dare le istruzioni nel modo seguente:

```
\begin{questions}
 \uplevel{Per rispondere alle domande
 \ref{dom:c} si utilizzino i teoremi
 sugli integrali.}
 \question[1]
  \label{dom:a}
  Cos'è l'insieme vuoto?
 \question[4]
  \label{dom:b}
  Sia dato un insieme con N oggetti.
  \begin{parts}
   \uplevel{Le due domande seguenti
   richiedono di fornire due definizioni}
   \part
    Definire cosa si intende con
    insieme delle parti.
   \part
    Determinare la cardinalità
    dell'insieme delle parti.
  \end{parts}
 \question
  \label{dom:c}
  \begin{parts}
   \part[15]
    Calcola
    $\displaystyle\int_0^1 x^2\, dx$.
   \part[15]
    Calcola
    $\displaystyle\int_0^1 (x^2+1)\, dx$.
  \end{parts}
\end{questions}
\fullwidth{Alla fine della prova, prima
di riconsegnare, verificare nuovamente
le risposte.}
```

ottenendo:

Per rispondere alla domanda 3 si utilizzino i teoremi sugli integrali.

1. (1 point) Cos'è l'insieme vuoto?
2. (4 points) Sia dato un insieme con N oggetti.

   Le due domande seguenti richiedono di fornire due definizioni

   (a) Definire cosa si intende con insieme delle parti.

   (b) Determinare la cardinalità dell'insieme delle parti.

3. (a) (15 points) Calcola $\displaystyle\int_0^1 x^2\, dx$.

   (b) (15 points) Calcola $\displaystyle\int_0^1 (x^2+1)\, dx$.

Alla fine della prova, prima di riconsegnare, verificare nuovamente le risposte.

In questo esempio sono state inserite delle istruzioni aggiuntive sia prima, che dopo il testo delle domande.

8 Questionari lunghi. Nomi delle parti

Se il questionario è molto lungo, lo si potrebbe suddividere in parti, assegnando ad ognuna di esse un nome. La classe `exam` consente di svolgere questo compito in due modi semplici, utilizzando i comandi `\fullwidth` e `\uplevel`, oppure utilizzando gli usuali comandi `\part` e `\section`.

8.1 Utilizzo di `fullwidth` e di `uplevel`

Per impostare il nome di una sezione del questionario si può utilizzare il comando `\fullwidth` ed impostare il font che si intende utilizzare. Ad esempio, se si vuole suddividere il questionario in una parte generale e una parte specifica, si scrive:

```
\begin{questions}
 \fullwidth{\large\bf Parte generale}
 \question Domanda uno...
 \question Domanda due...
 \fullwidth{\large\bf Parte specifica}
 \question Domanda tre...
 \question Domanda quattro...
\end{questions}
```

produce il seguente questionario:

Parte generale

1. Domanda uno...
2. Domanda due...

Parte specifica

3. Domanda tre...
4. Domanda quattro...

Si noti che la numerazione delle domande è unica per le due parti.

8.2 Utilizzo dei comandi standard di sezionamento

La classe `exam` è costruita sulla classe `article`, perciò è possibile utilizzare tutti i comandi di sezionamento propri di quella classe. In particolare, si possono utilizzare `part`, `part*`, `section` e `section*`.

9 Lo spazio per le risposte

9.1 Lasciare spazio bianco

Per lasciare una certa quantità di spazio bianco dove poter dare le risposte alle domande si deve utilizzare il comando `\vspace*`. Ad esempio, il comando `\vspace*{1cm}` inserisce un centimetro di spazio bianco dopo la linea nel quale è immesso. È possibile utilizzare anche il comando `\vspace` (senza l'asterisco), il cui effetto è quello di non avere effetti, se è posto all'inizio di una nuova pagina.

Si può lasciare la stessa quantità di spazio bianco per ognuna delle risposte di una pagina. Lo si fa utilizzando il comando `\vspace*{\fill}` dopo ogni domanda; alla fine della pagina si mette il comando `\newpage`.

9.2 Stampare delle linee per le risposte

Se si vogliono stampare delle linee bianche per le risposte è possibile utilizzare il comando:

```
\fillwithlines{length}
```

che riempie uno spazio verticale di altezza `lenght` con linee orizzontali. La distanza tra le linee orizzontali è per default di 0.25 pollici, tuttavia tale distanza può essere impostata a piacere utilizzando il comando:

```
\setlength\linefillheight{altezza}
```

dove, al posto di altezza, va inserita la misura della distanza tra le linee.

Anche lo spessore della linea può essere impostato utilizzando il comando:

```
\linefillthickness
```

come sempre, si inserirà nel codice la seguente riga:

```
\setlength\linefillthickness{0.1pt}
```

Se si vuole riempire lo spazio che segue una domanda fino alla fine della pagine con delle linee bianche è necessario utilizzare i seguenti comandi:

```
\fillwithlines{\fill}
\newpage
```

Se, invece, si vuole lasciare lo stesso numero di linee dopo ogni domanda di una pagina è necessario far seguire il comando `\fillwithlines` ad ogni domanda e chiudere la pagina con `\newpage`.

10 Risposte a scelta multipla

Negli esempi precedenti abbiamo visto come inserire nel questionario domande a risposta aperta. In alcuni casi, nei questionari, si ha bisogno di inserire domande a cui si associa un elenco di possibili risposte, tra le quali individuare quella corretta. Tali domande sono dette a risposta chiusa.

La classe *exam* viene incontro a questa necessità, mettendo a disposizione due diversi ambienti denominati `choices` e `oneparchoices`. Il primo dispone le risposte come una lista, cioè una per riga; il secondo ambiente dispone le risposte in un unico paragrafo, scrivendole una di seguito all'altra, nella stessa riga.

In modo predefinito entrambi gli ambienti enumerano le risposte con le lettere maiuscole "A", "B", "C" ecc.

Vediamo come funzionano i due ambienti con un esempio.

Se si scrive il codice:

```
\begin{questions}
 \question
   Dato l'insieme formato dai numeri
   dispari divisori di 30, quale dei
   seguenti numeri naturali è
   un suo elemento?
 \begin{choices}
  \choice
   10
  \choice
   7
  \choice
   5
  \choice
   9
  \choice
   25
 \end{choices}
\end{questions}
```

si produrrà il seguente risultato:

1. Dato l'insieme A formato dai numeri dispari divisori di 30, quale dei seguenti numeri naturali è un suo elemento?

A. 10
B. 7
C. 5
D. 9
E. 25

Invece, se si sostituisce a `choices` l'ambiente `oneparchoices`, si produrrà il seguente frammento di questionario:

1. Dato l'insieme A formato dai numeri dispari divisori di 30, quale dei seguenti numeri naturali è un suo elemento?

   A. 10 B. 7 C. 5 D. 9 E. 25

Evidentemente, la scelta fra le due soluzioni sarà fatta sulla base delle lunghezze delle risposte.

11 Soluzioni alle risposte

È possibile inserire nel questionario anche le risposte ai quesiti proposti. In questo modo si potrà avere a disposizione la chiave di correzione, o correttore, del questionario. Per farlo basterà utilizzare due ambienti della classe *exam*, denominati `solutions` e `solutionsorlines`.

Entrambi questi ambienti stampano le soluzioni, se presenti, altrimenti non fanno nulla. La stampa o meno delle soluzioni è controllata dai due comandi `\printanswers` e `\noprintanswers`. Per default l'ambiente non stampa le risposte. Un'alternativa all'uso del comando `\printanswers` è l'uso dell'opzione `answers` della classe *exam*, come nel seguente esempio:

```
\documentclass[answers]{exam}
```

che equivale ad utilizzare il comando `\printanswers` all'inizio del documento.

I due ambienti possono prendere come opzione l'indicazione dello spazio da riservare tra due domande se le risposte non vengono stampate. La dimensione può essere data in pollici o in centimetri. La differenza tra i due ambienti consiste nel fatto che col primo, `solutions`, lo spazio viene lasciato bianco, mentre col secondo, `solutionsorlines`, lo spazio viene riempito con linee orizzontali.

Ad esempio, con il seguente frammento di codice:

```
...
\printanswers
...
\begin{questions}
 \question
   Cosa si intende per intersezione
   di due insiemi?
 \begin{solution}
   Si intende quell'insieme
   formato dai soli elementi
   comuni ai due insiemi
   dati.
 \end{solution}
 ...
\end{questions}
...
```

si ottiene il frammento di questionario

1. Cosa si intende per intersezione di due insiemi?

   Solution: Si intende quell'insieme formato dai soli elementi comuni ai due insiemi dati.

Come già detto, se le risposte alle domande non vengono stampate, è possibile lasciare tra i quesiti un certo spazio, che può essere vuoto o riempito con linee orizzontali. A questo scopo bisogna assegnare la quantità di spazio da riservare per le risposte ad ogni singola domanda. L'esempio seguente mostra come fare.

```
...
\noprintanswers
...
\begin{questions}
 \question
   Cosa si intende per intersezione
   di due insiemi?
 \begin{solution}[3cm]
   Si intende quell'insieme
   formato dai soli elementi
   comuni ai due insiemi
   dati.
 \end{solution}
 ...
\end{questions}
...
```

In questo caso viene riservato uno spazio bianco alto 3 cm dopo il testo della domanda.

12 Tabella dei risultati

La classe *exam* offre la possibilità di inserire nel questionario una "tavola dei risultati" nella quale indicare i punteggi ottenuti in ogni singola domanda.

La tavola contiene tre serie di valori: il numero delle domande, il punteggio attribuito ad ogni domanda e il punteggio ottenuto con la risposta ad ogni domanda. Essa può essere disposta orizzontalmente oppure verticalmente.

Il comando che stampa la tabella è `\gradetables` e ad esso possono essere associati due argomenti opzionali. Il primo per determinare il tipo di orientamento e può assumere i valori `h` o `v` (orizzontale o verticale); il secondo determina se il raggruppamento dei punteggi va fatto per pagina o per singola domanda e può assumere i valori `pages` e `questions`.

Ecco alcuni esempi che ne chiariscono l'utilizzo:

`\gradetables[h][questions]` stampa la tabella orizzontalmente mostrando i numeri delle domande;

`\gradetables[v][questions]` stampa la tabella verticalmente mostrando i numeri delle domande;

`\gradetables[h][pages]` stampa la tabella orizzontalmente mostrando i numeri delle pagine;

`\gradetables[v][pages]` stampa la tabella verticalmente mostrando i numeri delle pagine.

13 Intestazioni e piè di pagina

Le intestazioni ed i piè di pagina del questionario possono essere personalizzate a piacere in modo semplice ed efficace.

L'intestazione è suddivisa in tre parti: la parte sinistra, quella centrale e quella destra. La parte sinistra è allineata a sinistra; la parte centrale è centrata; la parte destra è allineata a destra.

Il comando per gestire l'intestazione è:

```
\header{sinistra}{centro}{destra}
```

`sinistra` sarà l'intestazione scritta a sinistra, `centro` sarà l'intestazione scritta al centro, mentre `destra` sarà scritto a destra. Al posto di `sinistra`, `centro` e `destra` si può inserire qualunque testo, anche su più righe utilizzando il comando `\\`.

In modo del tutto analogo la gestione del piè di pagina si fa attraverso il comando:

```
\footer{sinistra}{centro}{destra}
```

14 Pagine di copertina

È possibile inserire una o più pagine prima del questionario vero e proprio, attraverso le quali fornire il titolo, le spiegazioni su come rispondere alle singole domande, le informazioni sui criteri di correzione e sui punteggi, ecc. Queste pagine iniziali non saranno numerate e sono limitate solo dalla fantasia del progettista. Ovviamente la copertina potrà essere riutilizzata per tutti i questionari che si vogliono creare.

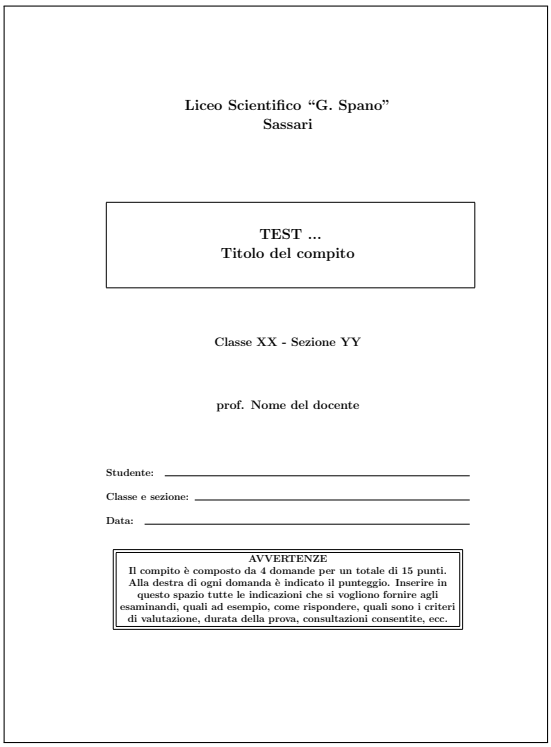
15 Esempio completo

Queste brevi note sulle potenzialità e sull'utilizzo della classe `exam` si concludono con uno schema di modello di questionario (vedi Figura 1, Figura 2 e Figura 3) che ho predisposto per la mia organizzazione – il Liceo Scientifico "Giovanni Spano" di Sassari – e che può essere facilmente utilizzato, modificato e ampliato in base alle proprie esigenze.

Il modello, infatti, non è esaustivo di tutte le possibilità offerte da `exam`. Chi è interessato ad approfondire la conoscenza di questo interessante pacchetto può leggere l'ottima guida **Using the exam document class** di *Philip Hirschhorn*.

Partendo dal codice di questo modello, se ne può creare un'altro, personalizzato, che risponde alle proprie esigenze, con una copertina che riporta i dati della propria organizzazione, le opportune indicazioni per lo svolgimento della prova, ecc.

Per la creazione di ogni nuovo questionario, quindi, basterà recuperare il modello ed aggiungervi, di volta in volta, le domande. La compilazione produrrà un questionario di qualità eccellente. Sulla qualità delle domande, invece, è tutta un'altra storia!


Liceo Scientifico "G. Spano"
Sassari

TEST ...
Titolo del compito

Classe XX - Sezione YY

prof. Nome del docente

Studente: _____

Classe e sezione: _____

Data: _____

AVVERTENZE
Il compito è composto da 4 domande per un totale di 15 punti. Alla destra di ogni domanda è indicato il punteggio. Inserire in questo spazio tutte le indicazioni che si vogliono fornire agli esaminandi, quali ad esempio, come rispondere, quali sono i criteri di valutazione, durata della prova, consultazioni consentite, ecc.

Figura 1: Frontespizio

Ecco il sorgente del modello per la creazione di un questionario.

```
% Esempio di questionario creato con la
% classe exam. I parametri del document-
% class possono essere modificati in ba-
% se alle esigenze.

\documentclass[a4paper,10pt]{exam}

% Pacchetti necessari per l'inserimento
% di formule matematiche e altri simbo-
% li di uso corrente

\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}
\usepackage{graphicx}
\usepackage{amsthm}
\usepackage{amsmath}
\usepackage{amssymb}
\usepackage{eurosym}
\usepackage{vmargin}
\usepackage{color}
```

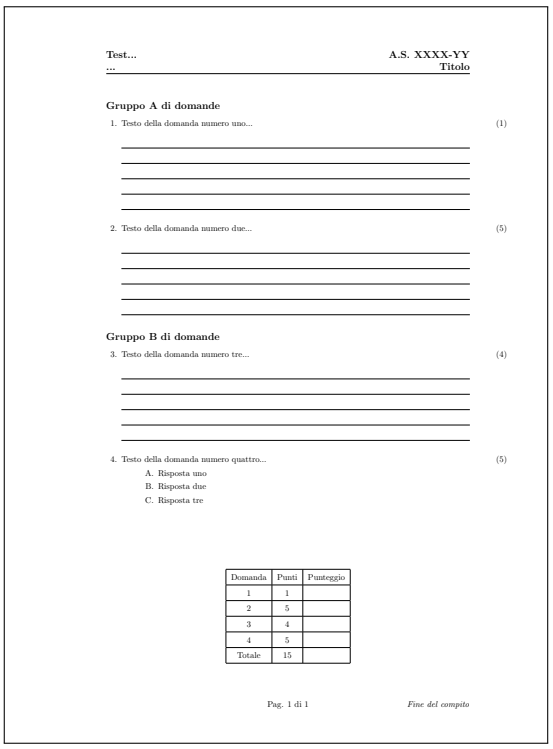


FIGURA 2: Pagina delle domande

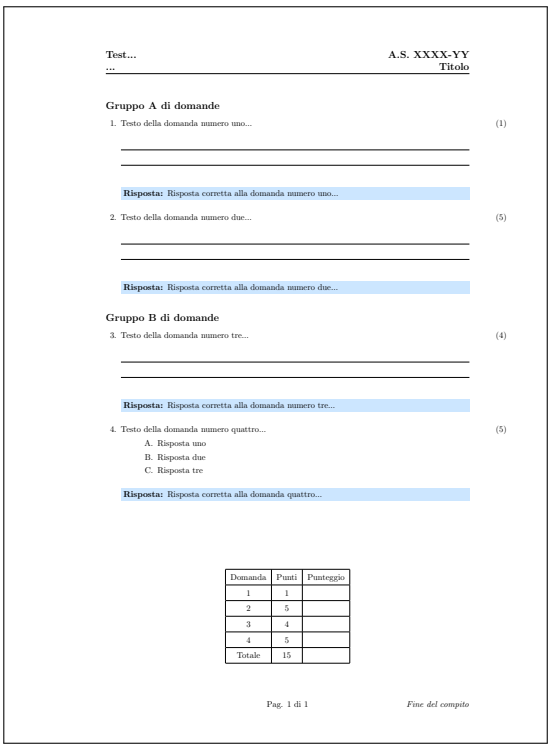


FIGURA 3: Pagina delle domande e risposte

```
% impostazioni relative alle soluzioni

\printanswers
% visualizza le soluzioni

\shadedsolutions
\definecolor{SolutionColor}
{rgb}{0.8,0.9,1}
\renewcommand{\solutiontitle}
{\noindent\textbf{Risposta:}\enspace}

% definizioni per la visualizzazione in
% italiano della tabella riepilogativa
% dei punteggi
\vqword{Domanda}
\hqword{Domanda}
\vpword{Punti}
\hpword{Punti}
\vsword{Punteggio}
\hsword{Punteggio}
\vtword{Totale}
\htword{Totale}

\pagestyle{headandfoot}
\firstpageheader{\bf\large Test...\\...}
{}{\bf\large A.S. XXXX-YY\\ Titolo}
\runningheader{\bf\large Test...\\...}
{}{\bf\large A.S. XXXX-YY\\ Titolo}
\headrule
\footer{}{Pag. \thepage\ di \numpages}
{\textit{\iflastpage{Fine del
compito}{segue \dots}}}
```

```
% titolo del compito
% inserire il titolo corretto del com-
% pito e la classe opzionalmente si può
% inserire la data del compito

\pointsinrightmargin
\pointpoints{ punto}{punti}

\begin{document}

% frontespizio del compito
\begin{coverpages}
\begin{center}

\LARGE \bf Liceo Scientifico ``G. Spano''
\\Sassari\\
\vspace{3cm}
\fbox{\parbox{15cm}{\vspace{1cm}
\centering \LARGE \bf TEST ... \\
Titolo del compito \vspace{1cm}}}\\
\vspace{2cm}
\Large Classe XX - Sezione YY
\\
\vspace{2cm}
\Large prof. Nome del docente
\\
\vspace{2cm}
\vspace{0.1in}
\large
\hbox to \textwidth{Studente:
\enspace\hrulefill}
\vspace{0.2in}
```

```
\hbox to \textwidth{Classe e
sezione:\enspace\hrulefill}
\vspace{0.2in}
\hbox to \textwidth{Data:
  \enspace\hrulefill}
\vspace{1cm}
\large
%\begin{center}
 \fbox{\fbox{\parbox{5.5in}
  {\centering \bf AVVERTENZE \\
   Il compito è composto da
   \numquestions\ domande per un
   totale di \numpoints\
   \points. Alla destra di ogni
   domanda è indicato il punteggio.
   Inserire in questo spazio tutte le
   indicazioni che si vogliono
   fornire agli esaminandi,
   quali ad esempio, come
   rispondere, quali sono i
   criteri di valutazione,
   durata della prova,
   consultazioni consentite, ecc.
  }}}
%\end{center}

\end{center}
\end{coverpages}

\addpoints

%% da questo punto iniziano le domande
\begin{questions}

% da utilizzare per raggruppare
% le domande rispetto ad un argomento
 \fullwidth{\large\bf Gruppo A}

% domanda e soluzione, duplicare
% questo blocco per ogni domanda
 \question[1] Testo della domanda
   numero uno...
 % il numero 1 indica il punteggio
\begin{solution}
 Risposta corretta alla domanda
   numero uno...
\end{solution}
 \question[5] Testo della domanda
   numero due...
\begin{solution}
 Risposta corretta alla domanda
   numero due...
\end{solution}
 \fullwidth{\large\bf Gruppo B}
 \question[4] Testo della domanda
   numero tre...
\begin{solution}
 Risposta corretta alla domanda
   numero tre...
\end{solution}
 \question[5] Testo della domanda
   numero quattro...
  \begin{choices}
   \choice
    Risposta uno
   \choice
    Risposta due
   \choice
    Risposta tre
  \end{choices}
 \begin{solution}
 Risposta corretta alla domanda
   numero quattro...
\end{solution}
\end{questions}

% tabella dei punteggi
\vspace{2cm}
\begin{center}
 \gradetable[v][questions]
\end{center}

\end{document}
```

▷ Antonello Pilu
Liceo Scientifico “G. Spano” - Sassari
antonello dot pilu at gmail
dot com

Introduzione agli strumenti per grecisti classici

Gianluca Pignalberi, Salvatore Schirone, Jerónimo Leal

Sommario

Il lavoro del grecista classico è parimenti difficile, o anche di più, di quello di un qualunque altro letterato. Per questo ha bisogno di strumenti di scrittura piuttosto sofisticati e potenti. Questo articolo presenta due editor, due compositori e alcune tecniche immediatamente utili per quel tipo di lavoro, e tutti utilizzati proficuamente dagli autori del presente articolo.

Abstract

Hellenists' work is as hard, or even harder, than every other literary men's work. That is why they need powerful and versatile tools. This paper shows two editors, two typesetters and some useful techniques for that kind of job. The authors of this paper effectively use all of them.

1 Introduzione

La vita dei grecisti classici può essere infernale: digitazione del testo in greco e nella propria lingua (almeno), controllo dei refusi e delle sviste, delle note e delle eventuali traduzioni a fronte. Se uno strumento come TEX può rendere quella vita un po' più accettabile, forse vale la pena affrontarne lo studio una volta per tutte. In questo articolo viene esposta parte dell'esperienza degli autori relativa a questo settore.

L'articolo è suddiviso come segue. La sezione 2 spiega l'uso di pdfLATEX e XELATEX per comporre dei documenti in italiano e in greco politonico. Per ognuno dei due compositori espone anche i metodi di composizione dei numeri secondo lo stile greco classico e del testo traslitterato. La sezione 3 introduce all'uso di Emacs e Yudit, due editor specializzati nella stesura di testi scritti in alfabeti anche diversi da quello latino. Dopo aver enucleato l'utilità di questi due editor, la stessa sezione elenca per puro spirito di completezza alcuni editor specifici per l'uso con TEX ma non specializzati nella digitazione di testi multilingue. L'elenco serve a ribadire sia la bontà delle due soluzioni proposte che la proficuità di scrivere un documento `.tex` con gli editor via via più adatti. Infine la sezione 4 affronta il problema della composizione di testi con traduzione a fronte.

2 I compositori

Questa sezione pone l'attenzione su due particolari implementazioni di LATEX: pdfLATEX e XELATEX. Il primo è probabilmente il più diffuso e completo pacchetto di macro per TEX; il secondo facilita l'uso dei font di sistema, sia True Type che Open Type, e lavora nativamente in Unicode. In entrambi i casi viene mostrato come scrivere testi in italiano e greco. Il lettore può sperimentare da sé che la sostituzione dell'italiano con un'altra lingua basata sull'alfabeto latino non comporta maggiori difficoltà.

Da questo punto in poi il termine *compositore* verrà usato come sinonimo di pdfLATEX o di XELATEX; ciò implica, alternativamente, indicare la sola funzione di composizione o che il contesto chiarisce quale dei due programmi si sta usando.

2.1 pdfLATEX

pdfLATEX è, probabilmente, il programma ormai più diffuso tra gli utenti di LATEX in virtù della sua compatibilità totale con quest'ultimo e la sua capacità di generare direttamente documenti in PDF. Questo compositore è in grado di comporre facilmente dei documenti contenenti testo in greco, anche classico, grazie all'uso dei pacchetti inputenc e babel.

Il primo dei due pacchetti serve a istruire il compositore su come interpretare il documento di input. Scrivere un file `.tex` contenente caratteri latini e greci implica ricorrere a codifiche "complesse", cioè con maggiore capacità espressiva rispetto alla codifica ASCII. La classica codifica ASCII a 7 bit permette di distinguere, e quindi di inserire in un file, tutti e soli i caratteri mostrati nella Tabella 1. Quando però il testo contiene anche delle lettere latine accentate, la codifica ASCII non è più sufficiente.

L'editor in uso dovrebbe accorgersi di ciò e usare una codifica adeguata, per esempio la ISO-8859-1, che aggiunge ad ASCII 7 bit i caratteri della Tabella 2. Dovendo scrivere un testo contenente lettere latine non accentate (come succede in inglese, a meno delle solite eccezioni dovute alle parole di origine straniera) e greche, l'editor dovrebbe scegliere da sé la codifica ISO-8859-7, che aggiunge ad ASCII 7 bit i caratteri mostrati nella Tabella 3. Infine, se l'intenzione è scrivere in italiano (lettere latine anche accentate) e in greco nello stesso documento, le codifiche appena viste sono soluzioni parziali, ed è naturale che l'editor codifichi il documento secondo lo standard Unicode (in genere UTF-8),[1] essendo

1. TEX è in grado di "simulare" le accentate latine (ad es. `\`e` → è), quindi è sempre possibile salvare un file `.tex` italiano e greco in un unico file codificato ISO-8859-7.

Tabella 1: Tabella dei codici ASCII 7 bit

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x																
1x																
2x		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	h	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	

Tabella 2: Tabella dei codici ISO-8859-1

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8x																
9x																
Ax		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯
Bx	°	±	2	3	´	µ	¶	·	¸	1	º	»	¼	½	¾	¿
Cx	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
Dx	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
Ex	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
Fx	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

questo una codifica multibyte. Quale che sia la codifica scelta, bisognerà specificarla a inputenc. Al caricamento del pacchetto si specificherà l'opzione adatta: iso-8859-1, iso-8859-7, oppure utf8 o utf8x.[2] Ecco il sorgente di un documento esemplificativo:

```
\documentclass{minimal}

\usepackage[T1]{fontenc}
\usepackage[utf8x]{inputenc}
\usepackage[polutonikogreek, italian]{babel}

\begin{document}
In principio era il Verbo, il Verbo era presso Dio
e il Verbo era Dio.

\textgreek{Ἐν ἀρχῇ ἦν ὁ Λόγος, καὶ ὁ Λόγος ἦν πρὸς
τὸν Θεόν, καὶ Θεὸς ἦν ὁ Λόγος.}
\end{document}
```

e il relativo Pdf risultante (Figura 1).

Quanto detto finora vale sia nel caso in cui la parte di testo in greco è scritta insieme alla parte in italiano che nel caso in cui la parte di testo in greco è scritta in un documento a se stante e inserita in un altro documento con in comando `\input`. Per semplificare le cose da ora in poi tutti i file verranno supposti codificati UTF-8.

L'uso delle codifiche appena viste (definite *multipiattaforma*) è preferibile all'uso delle codifiche locali legate al sistema operativo (latin1, applemac, ansinew...) e semplifica la vita a chi debba scambiare documenti tra diverse piattaforme. Se capita di avere un documento codificato localmente, può essere meglio fare una conversione verso una codifica multipiattaforma con strumenti come `iconv` o `recode` piuttosto che modificare lo specificatore nell'inclusione di inputenc.

Il sorgente precedente ha visto l'uso del comando `\textgreek`; questo indica quale parte del testo è in greco. Naturalmente esiste il comando equivalente `\textlatin`. È decisamente scomodo dover specificare quale parte del testo è scritta con quale alfabeto; in particolare è scomodo quando il testo greco è intercalato a quello italiano con una frequenza alta (ad esempio, una riga sì e una no). Il pacchetto auto-greek esiste per risolvere situazioni difficili come quella appe-

2. La codifica Unicode da usare richiede attenzione: se viene usato utf8 bisogna caricare anche il pacchetto ucs, altrimenti sarà possibile scrivere solo testi traslitterati come col metodo spiegato nella sezione 2.1.2.

Tabella 3: Tabella dei codici ISO-8859-7

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8x																
9x																
Ax		‘	’	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	―
Bx	°	±	2	3	΄	µ	¶	·	Έ	Ή	Ί	»	Ό	½	Ύ	Ώ
Cx	ΐ	Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι	Κ	Λ	Μ	Ν	Ξ	Ο
Dx	Π	Ρ	?	Σ	Τ	Υ	Φ	Χ	Ψ	Ω	Ϊ	Ϋ	ά	έ	ή	ί
Ex	ΰ	α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο
Fx	π	ρ	ς	σ	τ	υ	φ	χ	ψ	ω	ϊ	ϋ	ό	ύ	ώ	΄

In principio era il Verbo, il Verbo era presso Dio e il Verbo era Dio.
Ἐν ἀρχῇ ἦν ὁ Λόγος, καὶ ὁ Λόγος ἦν πρὸς τὸν Θεόν, καὶ Θεὸς ἦν ὁ Λόγος.

Figura 1: Testo compilato con pdfLaTeX

Tabella 4: Corrispondenze tra numeri arabi e numeri greci

1	α′	2	β′	3	γ′
4	δ′	5	ε′	6	ϝ′
7	ζ′	8	η′	9	ϑ′
10	ι′	20	κ′	30	λ′
40	μ′	50	ν′	60	ξ′
70	ο′	80	π′	90	ϟ′
100	ρ′	200	ς′	300	τ′
400	υ′	500	φ′	600	χ′
700	ψ′	800	ω′	900	ϡ′
1000	͵α	2000	͵β	3000	͵γ
4000	͵δ	5000	͵ε	6000	͵ϝ
7000	͵ζ	8000	͵η	9000	͵ϑ
10000	͵ι	20000	͵κ	30000	͵λ
40000	͵μ	50000	͵ν	60000	͵ξ
70000	͵ο	80000	͵π	90000	͵ϟ
100000	͵ρ	200000	͵ς	300000	͵τ
400000	͵υ	500000	͵φ	600000	͵χ
700000	͵ψ	800000	͵ω	900000	͵ϡ

na paventata. Non si trova su CTAN ma su `http://www.eelvex.net/latex-auto-greek/` insieme alle istruzioni di installazione e uso.

2.1.1 Numerazione

Il greco classico ha il proprio metodo di numerazione non basato sulla notazione posizionale, simile a quello romano. Il pacchetto babel fornisce gli strumenti per scrivere i numeri secondo tale notazione; fornisce anche un comando per convertire i numeri "arabi" in numeri greci: `\greeknumeral`. Il comando prende come unico argomento un numero tra 1 e 999.999.

La Tabella 4 riporta le corrispondenze tra numeri arabi e numeri greci e mostra le cifre sotto forma di lettere greche minuscole. Il comando `\Greeknumeral` mostra le cifre usando le lettere maiuscole. I simboli sono quelli definiti nel pacchetto teubner di Claudio Beccari.

In base alla notazione mostrata, il numero 987.654 diventa ͵ϡ͵π͵ζχνδ′.

Tre dei simboli usati per denotare le cifre in greco non assomigliano per niente alle lettere greche usualmente visibili: sono, rispettivamente, *stigma* per il 6, *qoppa* per il 90 e *sampi* per il 900. Il simbolo qoppa ha due forme diverse: la prima, più antica, a forma di 'q' o 'Q' (ϟ, Ϙ), la seconda, più moderna, con una forma a zig-zag (ϙ). Esiste anche la versione maiuscola di qoppa ma teubner non la definisce, e non la definiscono neanche ucs e babel. Il pacchetto arev[3] definisce quel carattere, ma solo sans serif. I simboli sampi e Sampi (rispettivamente ϡ e Ϡ) rimangono invariati, mentre stigma e Stigma assumono forme totalmente diverse: ϛ (o ϲ) e Ϛ se usati come simboli, oppure ϝ e F se usati come numeri.

L'approfondimento di questo argomento (e non solo) è possibile consultando Beccari (2010); Braams (2010); Pantieri (2008).

2.1.2 Traslitterazione

L'estensore di documenti TEX può essere interessato a scrivere del testo in greco senza dover digitare caratteri greci. La cosa è ragionevole (basti pensare a singoli termini o brevi frasi) e possibile ricorrendo alla traslitterazione. La traslitterazione è la trascrizione di un testo in un alfabeto diverso da quello originale, effettuata lettera per lettera secondo la corrispondenza dei suoni da esse rappresentati.

Siccome è normale per la maggior parte degli europei traslitterare il greco usando l'alfabeto latino, non serve codificare l'input (si assume implicitamente che sia ASCII), ma serve il supporto di babel caricato con l'opzione polutonikogreek per avere gli accenti e gli spiriti, dopodiché bisogna dimenticare i comandi `\textlatin` e `\textgreek`. O meglio, usarli è ancora possibile, ma gli spiriti e alcuni accenti andranno nelle posizioni sbagliate. La mossa corretta è ricorrere a `\selectlanguage` con argomento la lingua da usare da quel punto del documento fino al prossimo `\selectlanguage`.[4] Come il compositore traslittera in greco il testo latino? Il sorgente relativo all'esempio visto precedentemente in diversi stili aiuta la comprensione:

```
\documentclass{minimal}

\usepackage[T1]{fontenc}
\usepackage[utf8x]{inputenc}
\usepackage[polutonikogreek, italian]{babel}

\begin{document}
In principio era il Verbo, il Verbo era presso Dio
e il Verbo era Dio.

\selectlanguage{greek}
>En >arq{\~h|} \~>hn <o L'ogos, ka`i <o L'ogos \~>hn
pr`os t`on Je'on, ka`i Je`os \~>hn <o L'ogos.
\end{document}
```

Pur in apparenza differente, il sorgente garantisce un risultato uguale a quello della Figura 1.

3. Il pacchetto arev definisce i caratteri matematici e greci a complemento di bera, la riscrittura per LaTeX del font Bitstream Vera.

4. Nessuno vieta di inserire un bel comando in sostituzione della selezione, ad esempio `\newcommand\greco[1]{\selectlanguage{greek}#1\selectlanguage{italian}}` e poi scrivere `\greco{>En >arq~h| ~>hn <o L'ogos, ka'i <o L'ogos ~>hn pr'os t'on Je'on, ka'i Je'os ~>hn <o L'ogos.}`.

La Tabella 5 riporta le equivalenze tasto-carattere o, meglio, $_{\text{tasto}}$carattere, dove "tasto" indica quello premuto sulla tastiera e "carattere" indica quello visibile nell'editor o nel documento finale.

La facilità di inserire il testo in greco e la codifica Unicode (o più codifiche ISO) rendono obsoleto il metodo appena visto. Resta valido l'assunto di partenza: l'esiguità del testo greco da scrivere può giustificare il ricorso alla traslitterazione.

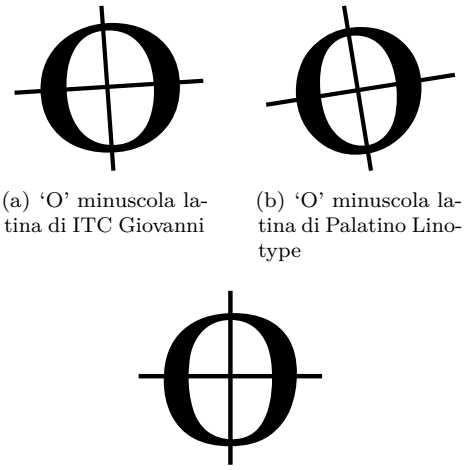
2.2 XeLaTeX

Lavorare professionalmente nell'editoria comporta regolarmente l'uso di font commerciali o comunque non sempre disponibili a LaTeX e a pdfLaTeX. Molti dei font usati dagli editori non hanno un corrispettivo utilizzabile con TeX. Salvo costruirsi da sé il supporto a tali font, l'unica altra via è scegliere uno strumento capace di usare i font True Type (`.ttf`) e Open Type (`.otf`). XeTeX è proprio questo strumento. Una volta installati dei font di quel tipo sul proprio sistema, XeTeX è in grado di usarli senza alcun problema.

Non tutti i font hanno i glifi del greco, del cinese, del russo e degli altri alfabeti diversi dal latino. Capita spesso che un font anche commerciale non abbia neanche tutti i glifi necessari alle lingue basate sull'alfabeto latino. Quando possibile, bisogna scegliere il font adatto al lavoro da svolgere. Nel caso specifico serve un font che abbia anche i glifi del greco. GNU FreeFont della Free Software Foundation li ha; tra i font commerciali Palatino Linotype fornito con Windows li ha meno completi.

A differenza di pdfLaTeX, XeLaTeX non ha bisogno di un comando come `\textgreek` per capire quando inserire i glifi greci. I caratteri vengono composti correttamente da XeLaTeX stesso che lavora nativamente con Unicode. Usare e comporre correttamente i caratteri non vuol dire aver riconosciuto la lingua; anche XeLaTeX ha bisogno di un pacchetto per attivare le sillabazioni corrette (polyglossia).

L'uso di un font obbligatorio ma senza glifi per il greco richiede l'uso contemporaneo di un secondo font che li abbia, così da rendere il testo in greco. Ci sono tre tecniche per passare dall'uno all'altro font e saranno descritte nelle sezioni 2.2.1, 2.2.2 e 2.2.3. In tutti gli esempi di questa sezione il font obbligatorio è ITC Giovanni; gli viene affiancato il già citato Palatino Linotype che, pur non molto somigliante nei glifi latini, è abbastanza simile a ITC Giovanni nel disegno dei glifi greci. Per capire quest'affermazione bisogna osservare le figure 1 e 3. Il testo italiano della Figura 3 ha la 'o' (ITC Giovanni) a simmetria pressoché orizzontale e verticale; il testo greco della stessa figura ha la 'o' (omicron, Palatino Linotype) con la simmetria orizzontale e verticale, molto simile a ITC Giovanni. La 'o' latina di Palatino Linotype è invece simile alla 'omicron' del testo greco della Figura 1:



(a) 'O' minuscola latina di ITC Giovanni

(b) 'O' minuscola latina di Palatino Linotype

(c) 'Omicron' minuscola di Palatino Linotype

FIGURA 2: Dettaglio della lettera 'o' (e 'omicron') minuscola di due font diversi e dei relativi assi di simmetria

gli assi di simmetria sono evidentemente ruotati, sebbene in direzioni differenti. La Figura 2 mostra un ingrandimento dei caratteri menzionati. Gli assi ivi riportati sono indicativi e non tengono conto dei dettagli del disegno dei glifi, come ad esempio il fatto che i profili esterni delle 'o' sono disassati rispetto ai profili interni.

2.2.1 Cambio di font per selezione diretta

Il primo metodo utilizzabile per ottenere il risultato voluto consiste nello specificare il font da impiegare immediatamente prima del testo da comporre. Il pacchetto fontspec fornisce allo scopo il comando `\fontspec` come mostra il sorgente seguente. Purtroppo per passare dall'italiano al greco o viceversa bisogna ridefinire il font in uso, specificandone anche tutti gli attributi (nell'esempio solo le legature tipiche di TeX).

```
\documentclass{minimal}

\usepackage{fontspec}
\usepackage{polyglossia}
\setmainfont[Mapping=tex-text]{ITC Giovanni}
\setmainlanguage{italian}

\begin{document}
In principio era il Verbo, il Verbo era presso Dio
e il Verbo era Dio.

\fontspec[Mapping=tex-text]{Palatino Linotype}
Ἐν ἀρχῇ ἦν ὁ Λόγος, καὶ ὁ Λόγος ἦν πρὸς
τὸν θεόν, καὶ θεὸς ἦν ὁ Λόγος.
\end{document}
```

La Figura 3 mostra il risultato della compilazione.

2.2.2 Cambio di font tramite le famiglie di font

Questo secondo metodo, senz'altro il più intelligente tra i tre proposti, prevede di assegnare a

Tabella 5: Caratteri greci (politonici) ottenuti con LaTeX per traslitterazione

|.		"'									?;
								``	`'		¡ì
QX	WΩ	EE	RP	TT	YΨ	UΥ	II	OO	PΠ	éé	
qχ	wω	eε	rϱ	tτ	yψ	uυ	iι	oο	pπ	èè	
AA	SΣ	DΔ	FΦ	GΓ	HH	JΘ	KK	LΛ	ÒÒ	ÀÀ	ÙϒÌ
aα	sς	dδ	fφ	gγ	hη	jϑ	kϰ	lλ	òò	àα	ùὺ
ZZ	XΞ	C◌̓	V◌̌	BB	NN	MM	;·			>◌̔	
zζ	xξ	cϛ		bβ	nν	mμ				<◌̔	
<~h|ᾖ	>~h|ᾗ	<'h|ᾄ	>'h|ᾅ	<`h|ᾂ	>`h|ᾃ						

I caratteri dell'ultima riga sono stati riportati solo come esempio dell'ordine in cui spiriti, accenti e iota sottosegnato vanno scritti per il corretto posizionamento. Pur non avendo sempre bisogno di tutti i segni diacritici contemporaneamente, occorre ricordare quali devono essere scritti prima e quali dopo il carattere. In rosso sono evidenziate le differenze o le aggiunte rispetto alle corrispondenze tastocarattere di Emacs (Tabella 6).

In principio era il Verbo, il Verbo era presso Dio e il Verbo era Dio.
Ἐν ἀρχῇ ἦν ὁ Λόγος, καὶ ὁ Λόγος ἦν πρὸς τὸν Θεόν, καὶ Θεὸς ἦν ὁ Λόγος.

Figura 3: Testo compilato con XeLaTeX

uno *switch*, un comando-scorciatoia, la famiglia di font alternativa completa di attributi e di attivarlo per ottenere il testo successivo composto nel font alternativo.

Il problema sorge omettendo di associare uno switch anche al font principale. L'attivazione dello switch per il font alternativo lascia attivo quest'ultimo fino allo switch successivo. L'unico modo di tornare al font principale senza switch è ricorrere alla selezione diretta discussa nella sezione 2.2.1.

Il manuale di fontspec suggerisce la definizione di un comando che accetti come parametro il testo da comporre col font alternativo. Il listato seguente mostra un esempio di definizione di uno switch di portata (*scope*) limitata mediante la creazione di un comando (`\greco`) che sfrutta una precedente definizione `\newfontfamily`.

Il sorgente dello switch definito tramite `\newfontfamily` e del comando con scope limitato definito con `\newcommand` si trovano nel listato seguente:

```
\documentclass{minimal}

\usepackage{fontspec}
\usepackage{polyglossia}
\setmainfont[Mapping=tex-text]{ITC Giovanni}
\setmainlanguage{italian}
\newfontfamily\pal{Palatino Linotype}
\newcommand\greco[1]{{\pal #1}}

\begin{document}
In principio era il Verbo, il Verbo era presso Dio
e il Verbo era Dio.

\greco{Ἐν ἀρχῇ ἦν ὁ Λόγος, καὶ ὁ Λόγος ἦν πρὸς
τὸν Θεόν, καὶ Θεὸς ἦν ὁ Λόγος.}
\end{document}
```

Il risultato è quello visto nella Figura 3.

2.2.3 *Cambio di font tramite le famiglie di default o le relative forme*

L'ultima delle soluzioni proposte è assegnare il font per comporre il greco a una delle famiglie di default (serif, sans o mono) o a una delle loro forme (grassetto, corsivo, maiuscoletto). Questa soluzione non è molto elegante o ortodossa, quindi viene citata come mera possibilità tecnica.

L'esempio che segue ridefinisce sia la famiglia mono (*tt*) che il maiuscoletto della famiglia serif per fornire il Palatino. Il risultato è di nuovo esattamente lo stesso visto nella Figura 3.

```
\documentclass{minimal}

\usepackage{fontspec}
\usepackage{polyglossia}
\setmainfont[Mapping=tex-text,
SmallCapsFont={Palatino Linotype}]{ITC Giovanni}
\setmonofont[Mapping=tex-text]{Palatino Linotype}
\setmainlanguage{italian}

\begin{document}
In principio era il Verbo, il Verbo era presso Dio
e il Verbo era Dio.

\texttt{Ἐν ἀρχῇ ἦν ὁ Λόγος, καὶ ὁ Λόγος ἦν πρὸς}
\textsc{τὸν Θεόν, καὶ Θεὸς ἦν ὁ Λόγος.}
\end{document}
```

2.2.4 *Numerazione*

Anche il pacchetto polyglossia, al pari di babel, fornisce il meccanismo per la numerazione greca classica. Quando viene selezionata la lingua greca è possibile usare i comandi `\greeknumber` o `\greeknumeral` per ottenere dei numeri scritti con caratteri minuscoli oppure `\Greeknumber` o `\Greeknumeral` per ottenere dei numeri scritti con caratteri maiuscoli. Anche in questo caso bisogna

trovare un font adeguato visto che Palatino Linotype difetta dei varii sampi e qoppa. GNU FreeFont potrebbe essere un ottimo candidato, visto che ha al suo interno sia qoppa che Qoppa e gli altri glifi necessari. È anche un font libero, quindi non necessita di pagamenti e si può modificare in base alle proprie esigenze.

2.2.5 Traslitterazione

Non occorre ribadire che XeLaTeX è stato progettato per lavorare nativamente con Unicode. Dev'essere questa la ragione per cui in nessun modo si riesce a fargli comporre un testo greco traslitterato.

3 Gli editor

Per digitare dei testi in greco si può usare uno dei due editor proposti, entrambi in grado di rimappare la tastiera indipendentemente dal sistema operativo, dalle sue impostazioni linguistiche e dal driver della tastiera installato.

3.1 Emacs

Emacs[5] è probabilmente l'editor più complesso e completo al mondo. Offre un supporto multilingua invidiabile e salva i file in ASCII o UTF-8 automaticamente in base al loro contenuto. Avviato l'editor è sufficiente premere `Ctrl-\` per scegliere il metodo di input e la conseguente configurazione della tastiera aggiuntivi a quelli predefiniti. La stessa sequenza di tasti servirà poi per passare dall'una all'altra configurazione.

La scrittura in greco classico richiede l'impiego di spiriti e accenti. Il metodo di input da selezionare in Emacs è `greek-babel`.

La Tabella 6 mostra una parte delle corrispondenze $_{\text{tasto}}$carattere di Emacs. La lista completa può essere consultata direttamente in Emacs selezionando Options→Mule (Multilingual Environment)→Describe Input Method... o premendo `Ctrl-h I` (Figura 4).

3.2 Yudit

Yudit è un altro editor in grado di scrivere nativamente in tutte le lingue poiché è stato pensato come editor Unicode. Anche questo editor salva un file con la codifica ASCII o UTF-8 a seconda del contenuto. Le mappe della tastiera sono associabili ai tasti funzione così da rendere più rapida la selezione delle mappe linguistiche usate con più frequenza.

Per associare una mappa a un tasto funzione bisogna cliccare innanzitutto sul tasto delle mappe; nella Figura 5 è il tasto con la freccia e l'etichetta straight (che indica la tastiera effettivamente installata nel sistema). Si apre dunque la finestra di selezione mappe (Figura 6).

Nel box F-Key Current KMaps si può selezionare uno dei tasti funzione (già associato a un'altra

5. Versione 23-2.

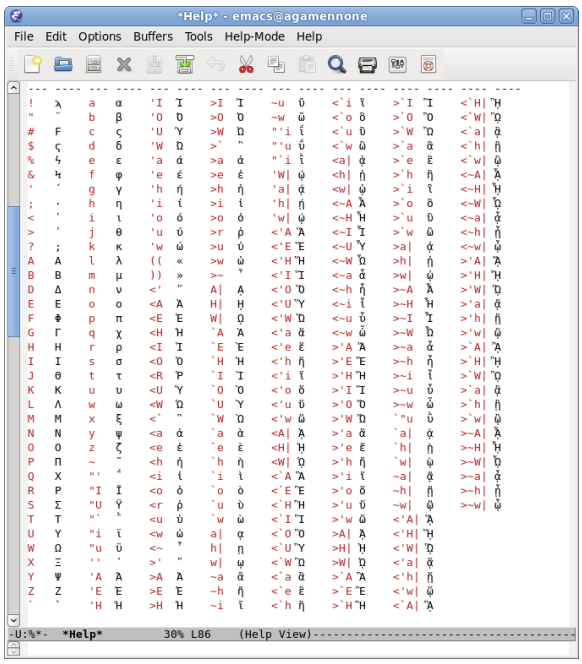


FIGURA 4: Sequenze di input in Emacs per il greco

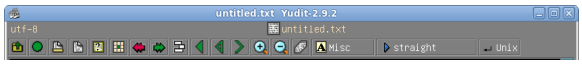


FIGURA 5: La barra dei comandi di Yudit

mappa) a cui associare la tastiera voluta tra quelle elencate nel box Available KMaps (Figura 7). A quel punto basta cliccare sul tasto freccia (quello con sopra il puntatore del mouse nella Figura 7).

Ora al tasto F6 (quello scelto nell'esempio) compare GreekPolytonic in luogo di GreekMonotonic e una serie di sequenze di tasti definiti per quel metodo (Figura 8); alla conferma con OK Yudit è pronto alla digitazione di testi greci politonici in base alle corrispondenze riportate nella Tabella 7.

Il numero e la varietà di caratteri permessi da Yudit è impressionante, e si può "vedere" scorrendo il contenuto della finestra Key Input Output del greco politonico.

L'uso del verbo vedere tra virgolette significa che in generale la maggior parte dei caratteri non

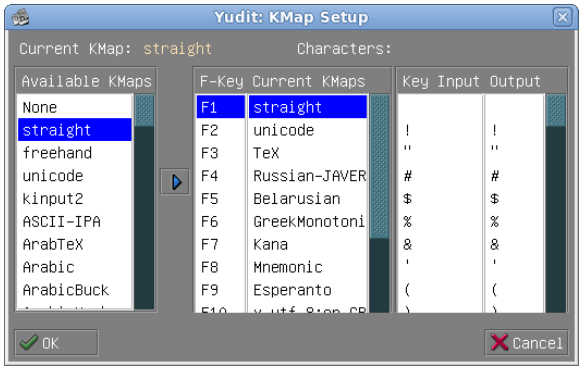


FIGURA 6: Finestra di associazione mappe–tasti funzione

Tabella 6: Caratteri greci (politonici) corrispondenti ai tasti della tastiera italiana in Emacs

|.	!᾽	".		$῾	%῍	&Ϙ*				?;
										`` ´
QX	WΩ	EE	RP	TT	YΨ	UϒΥ	II	OO	PΠ	
qχ	wω	eε	rρ	tτ	yψ	uυ	iι	oο	pπ	
AA	SΣ	DΔ	FΦ	GΓ	HH	JΘ	KK	LΛ		
aα	sς	dδ	fφ	gγ	hη	jϑ	kϰ	lλ		
ZZ	XΞ			BB	NN	MM	;·		>̓	
zζ	xξ	cϛ		bβ	nν	mμ			<̓	
<~h|῁	>~h|῁	<'h|῁	>'h|῁	<`h|῁	>`h|῁					

Per la spiegazione sui caratteri dell'ultima riga cfr. la Tabella 5.

*Il simbolo Koppa (o Qoppa o Coppa) qui riportato è una forma alternativa di quello mostrato da Emacs, più simile al simbolo koppa ottenuto col tasto %. Il pacchetto Teubner non definisce questo glifo.

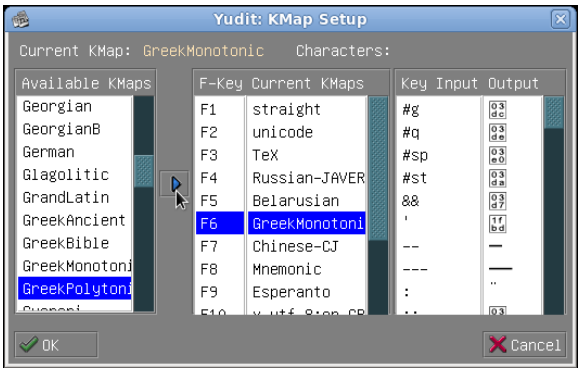


Figura 7: Dobbiamo associare una mappa tra quelle disponibili a un tasto funzione

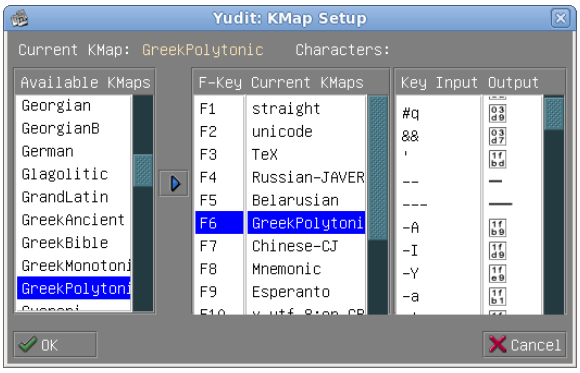


Figura 8: Il greco politonico è ora associato al tasto F6

è visibile nella finestra menzionata ma è sostituita dal codice Unicode corrispondente (Figura 8). Scegliendo il font adatto anche per questa sezione dell'editor quei caratteri altrimenti invisibili diventano visibili (Figura 9).

Yudit permette anche l'inserimento di caratteri tramite il loro codice Unicode. Ad esempio, per digitare '—' (codice Unicode 2014) bisogna selezionare la tastiera unicode e digitare la sequenza `u2014`; alla fine della digitazione comparirà il carattere voluto ('—', em-dash).

Maggiori informazioni su Yudit si possono ottenere leggendo le FAQ del programma (consultabili tramite il comando `help` dalla linea di comando di Yudit) o facendo riferimento a Sinai (2011).

3.3 Altri editor

Ovviamente, grazie alla natura testuale del documenti `.tex`, qualunque editor è valido, dal Blocco Note di Windows, allo spartano `vi` (e le accresciute caratteristiche di `vim`) a (per assurdo) MS Word™ e Open/Libre Office.

Tra i molti editor dedicati a TEX dobbiamo senz'altro citare TEXworks (Kew e Löffler, 2011), installato di default da TEX Live, TEXnicCenter (Weinkauf e Wiegand, 2011), WinEdt (Simonic (2011), una scelta "naturale", sebbene non libera, per gli utenti MikTEX), Texmaker (Brachet, 2011) e il suo diretto concorrente TexMakerX (van der Zander, 2011). Ognuno di essi ha pregi, difetti e caratteristiche adatte a uno stile di lavoro o a un compito specifico.

Tabella 7: Caratteri greci (politonici) corrispondenti ai tasti della tastiera italiana in Yudit

Q:	W·	EE	RP	TT	YϒΥ	UΘ	II	OO	PΠ
q;	wς	eε	rρ	tτ	yυ	uϑ	iι	oο	pπ
AA	SΣ	DΔ	FΦ	GΓ	HH	JΞ	KK	LΛ	
aα	sς	dδ	fφ	gγ	hη	jξ	kϰ	lλ	
ZZ	XX	CΨ	VΩ	BB	NN	MM			
zζ	xχ	cψ	vω	bβ	nν	mμ			
<~h|῁	>~h|῁	<;h|῁	>;h|῁	<`h|῁	>`h|῁				

Per la spiegazione sui caratteri dell'ultima riga cfr. la Tabella 5.

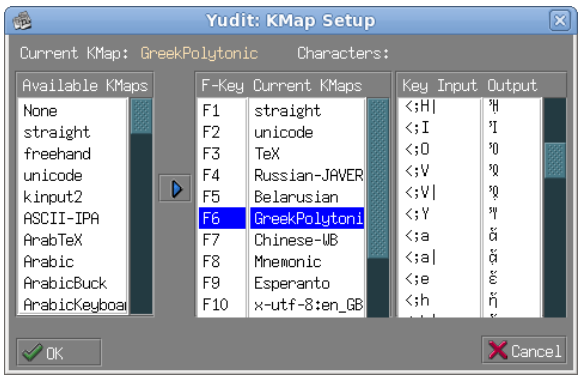


Figura 9: Col font giusto possiamo vedere quali caratteri vedremo anche nell'editor

I programmi appena menzionati non vengonoo approfonditi in questa sede perché, non essendo editor specifici per l'inserimento di testi multilingua, fanno affidamento sulle impostazioni del sistema operativo relative alla lingua e alla tastiera. L'editor non trasforma in alcun modo il testo ricevuto in input. Il fatto che riesca poi a mostrare i glifi selezionati non è poi così scontato: neanche Yudit ci riesce se non dopo aver impostato il font appropriato. Il sito Leal (2011) mostra alcune indicazioni su come rimappare la tastiera in base al sistema operativo e le tabelle di corrispondenza dei caratteri greci sulla tastiera italiana.

Scegliere uno o l'altro degli editor proposti non inficia comunque il lavoro di TEXe quanto descritto nella sezione 2 rimane, naturalmente, invariato e valido.

Detto ciò il lettore potrebbe domandarsi:«Qual è allora il vantaggio di usare Emacs o Yudit in luogo di uno degli altri editor elencati?»

Due sono i vantaggi immediati:

1. entrambi gli editor sono multipiattaforma. Ciò significa che se l'utente deve usarli su diversi sistemi operativi non deve preoccuparsi di sapere come questi ultimi mappano le tastiere, essendo i editor autosufficienti da questo punto di vista;

2. l'invariata produttività di cui al punto precedente aumenta considerando che entrambi hanno anche degli ausilii specifici per TEX.

Questo rende i due editor qui discussi molto più utili al lavoro dei grecisti che non gli editor specifici per TEX o qualunque altro editor o word processor esistente.

4 Edizioni con traduzione a fronte

Le edizioni con traduzione a fronte rappresentano un problema non ancora risolto completamente. I pacchetti realizzati allo scopo sono senz'altro buoni ma non scevri da imperfezioni che i loro stessi autori riconoscono.

4.1 Il pacchetto parallel

Il primo pacchetto utilizzabile per impaginare un testo con la sua traduzione a fronte è parallel. È possibile impaginare i due testi su due colonne nella stessa pagina o sue due pagine affiancate.

Il pacchetto parallel (Eckermann, 2003) fornisce l'ambiente Parallel; questo prende, nell'ordine, un parametro opzionale che indica lo stile di impaginazione (`c` su due colonne, `v` su colonne separate da una linea verticale, `p` su due pagine affiancate) e due parametri obbligatori relativi all'ampiezza della gabbia del testo (sinistra e destra).

I testi sinistro e destro vanno inclusi all'interno dell'ambiente mediante i comandi `\ParallelLText` e `\ParallelRText`. Sempre all'interno dell'ambiente, i paragrafi paralleli si possono separare tramite `\ParallelPar`.

L'autore del pacchetto avverte dell'esistenza di alcuni problemi, segnatamente la limitata capacità dei testi sinistro e destro[6] e le restrizioni nell'uso di tabelle, *minipage* e altri meccanismi simili all'interno dell'ambiente Parallel (nel caso di pagine parallele) o solo all'interno di `\ParallelLTeXt` e `\ParallelRText` (nel caso di colonne parallele). Un altro problema è rappresentato dalla stampa delle note a piè di pagina solo alla fine dell'ambiente.

Nella Figura 10 è visibile il primo versetto del Vangelo di Giovanni su due colonne parallele. Alla fine di ogni versione è stata posta una nota a piè di pagina di controllo. Purtroppo non è possibile in alcun modo distinguere le note in base alla loro posizione. Se la composizione del testo fosse avvenuta su due pagine parallele, le note a piè di pagina sarebbero andate in una pagina nuova, sempre una dopo l'altra.

Il pacchetto va senz'altro bene per soddisfare esigenze di composizione limitate. Se ci si accontenta di gestire "a mano" i casi più complicati, anche usando soluzioni miste non strettamente legate al pacchetto, si possono azzardare dei layout di pagina un po' più complessi.

4.2 Il pacchetto ledpar

Chi ha l'esigenza di comporre un'edizione critica con traduzione a fronte non troverà in parallel uno strumento di lavoro adeguato. A tale proposito Peter Wilson ha realizzato il pacchetto ledpar (Wilson, 2005). Questo compone il testo di un'edizione critica e i relativi apparati in maniera più coerente rispetto a parallel: ogni testo avrà il proprio apparato nella posizione corretta, e anche le

6. Un breve esperimento degli autori di questo articolo ha mostrato che, prima di avere problemi in compilazione, parallel ha digerito una sequenza di 900 (150 × 6) paragrafi del pacchetto lipsum, generando un documento di 419 pagine. Il primo errore in compilazione è stato ottenuto con 1050 (150 × 7) paragrafi e, siccome non sono state fatte prove con un numero di paragrafi inferiore, si può solo dire che un numero di paragrafi p : $900 < p \leqslant 1050$ decreta il superamento del limite del pacchetto.

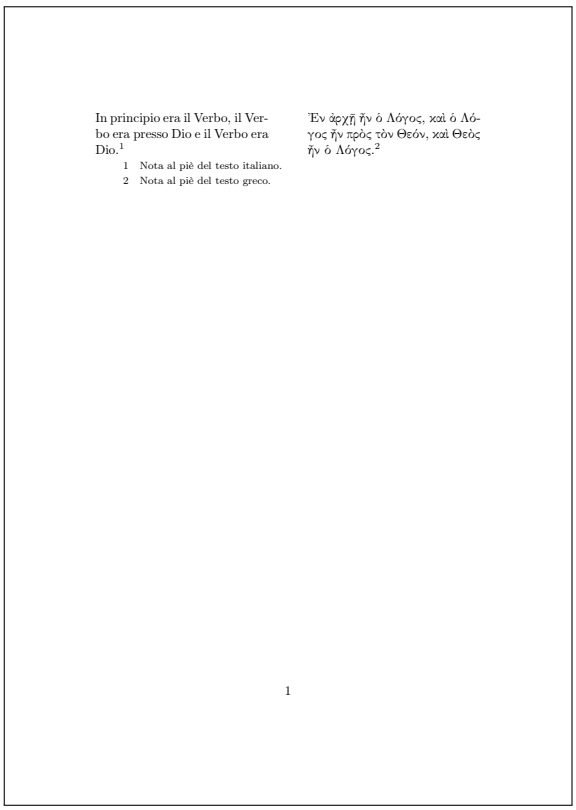


FIGURA 10: Testo comune con traduzione parallela su due colonne

FIGURA 11: Testo di un'edizione critica (senza apparati) con traduzione parallela su due colonne

note a piè di pagina saranno messe nella giusta posizione, a meno che il testo non venga composto in colonna; in quest'ultimo caso le note saranno poste sempre sotto la colonna sinistra (come nella Figura 11).

Sebbene i comandi e gli ambienti forniti abbiano una sintassi diversa dai comandi di parallel, lo stile d'uso è identico, con la definizione del testo che andrà sulle pagine pari (o nelle colonne sinistre) e dispari (o colonne destre).

Per comporre il documento della Figura 11 si può scrivere questo codice:

```
\documentclass{article}

\usepackage[T1]{fontenc}
\usepackage[utf8x]{inputenc}
\usepackage[polutonikogreek, italian]{babel}
\usepackage{ledmac}
\usepackage{ledpar}
\usepackage[a5paper]{geometry}

\begin{document}
\begin{pairs}
\setlength{\Lcolwidth}{0.45\textwidth}
\setlength{\Rcolwidth}{0.45\textwidth}

\begin{Leftside}
\beginnumbering
\pstart
\noindent
In principio era il Verbo, il Verbo era presso Dio e
il Verbo era Dio.\footnote{Nota al pi\`e del testo
italiano.}
\pend
\endnumbering
\end{Leftside}
\begin{Rightside}
\beginnumbering
\pstart
\noindent
\textgreek{Ἐν ἀρχῇ ἦν ὁ Λόγος, καὶ ὁ Λόγος ἦν πρὸς
τὸν Θεόν, καὶ Θεὸς ἦν ὁ Λόγος.}\footnote{Nota al
pi\`e del testo greco.}
\pend
\endnumbering
\end{Rightside}
\Columns
\end{pairs}
\end{document}
```

Il pacchetto ledpar è più complesso di parallel e mette a disposizione alcuni comandi e ambienti per comporre versi. L'impiego di ledmac è propedeutico all'uso di ledpar. In tal modo è possibile usare i meccanismi di numerazione delle righe del testo critico, di composizione e di numerazione automatica degli apparati, e quanto necessario per comporre un'edizione critica.

Attenzione: il testo non numerato non verrà composto in colonne parallele e le note al piè non riconosceranno le lettere accentate se non nel puro stile TEX.

4.3 Divisione manuale

Nei casi più ostici, e comunque quelli in cui i pacchetti menzionati nelle sezioni 4.1 e 4.2 non danno risultati soddisfacenti, bisogna ricorrere all'intervento manuale. Nel capitolo 5 di LEAL e PIGNALBERI (2011) viene proposto un metodo la cui efficacia

non fa rimpiangere il tempo speso nelle prove per trovare i punti di suddivisione ottimali.

L'esempio citato si riferisce a un caso di composizione di un'edizione critica per cui ledpar non riesce a gestire correttamente l'apparato e le relative note. Il modo più immediato di comporre il testo critico e la sua traduzione è creare due documenti separati, uno per il testo critico e uno per la traduzione, numerandone le pagine rispettivamente solo con numeri pari e dispari. I documenti creati in questo modo saranno poi fusi, intercalandone le pagine, nel documento finale.[7]

Sarà cura dell'autore o del compositore far corrispondere il testo critico di una pagina alla traduzione della pagina a fronte. Questo può comportare il ricorso all'interruzione manuale delle pagine. A seconda del punto e del metodo di interruzione, la creazione di un capoverso arbitrario o altri problemi possono inficiare la qualità del lavoro.

Il documento preso come esempio è il file `recomplex.tex`, tratto da Leal e Pignalberi (2011) ma leggermente modificato nelle dimensioni della pagina. Il suo codice è mostrato nella Figura 17 in coda all'articolo, mentre il documento finale è mostrato nella Figura 12.

Il caso in esame prevede di interrompere la prima pagina del documento nella Figura 12 immediatamente prima del punto '5' nell'ultima riga (nel sorgente è la riga 264 di pagina 30). Si può decidere di mandare a capo il testo dal punto indicato in poi lasciando una riga vuota tra le righe 264 e 265 del sorgente (*soluzione 1*). In questo caso si introdurrà un capoverso con tanto di rientro della riga successiva ma assente nella versione originale del testo, la riga interrotta non sarà giustificata e la pagina non sarà interrotta al punto giusto (Figura 13).

La *soluzione 2* prevede di mandare a capo il testo col comando `\\` quindi il codice della riga 264 diventa `{\Afootnote{\textit{Act.} 2,17; \textit{Joel} 2,28.}}.\\`. In questo caso la pagina sarà interrotta correttamente, non ci sarà il rientro nella riga successiva ma la riga interrotta non sarà giustificata (Figura 14).

L'uso del comando `\newpage` da porre nel punto in cui si vuole interrompere la pagina dà un risultato comunque negativo (`{\Afootnote{\textit{Act.} 2,17; \textit{Joel} 2,28.}}. \newpage`, *soluzione 3*): la pagina viene interrotta nel punto desiderato ma senza giustificare la riga e introducendo un capoverso arbitrario con rientro della prima riga (Figura 15).

La soluzione migliore in questo caso (*soluzione 4*) è il comando di interruzione di riga `\linebreak` posto all'inizio della riga 265 del sorgente (`\linebreak 5. Itaque et nos qui sicut prophetias ita et uisiones`) che, oltre ad andare a capo senza rientri o altro, giustifica la riga interrotta (Figura 16).

È evidente che il caso appena osservato sia un caso molto particolare, con il punto di interruzione posto all'ultima riga di una pagina e vicino alla fine della riga. Il problema dell'interruzione migliore non ha una soluzione univoca, pertanto saranno richieste diverse prove per trovare la soluzione migliore per ogni caso.

7. L'uso di pdfpages è fortemente consigliato.

5 Conclusioni

L'articolo ha introdotto alcuni strumenti e metodi utili ai grecisti classici per comporre le loro opere con TEX: programmi di composizione (pdfLATEX e XELATEX), editor (Emacs e Yudit), ausilii alla numerazione classica (teubner polyglossia), alla traslitterazione (babel) e al posizionamento dei testi tradotti con testo originale a fronte (parallel e ledpar, oltre a una soluzione manuale molto efficace). L'intento era fornire una minima serie di strumenti e di istruzioni (una specie di *starters' kit*) per rendere operativi fin da subito i grecisti in questione. Sebbene ogni utente LATEX abbia ampia scelta nel personalizzare il proprio ambiente di lavoro, le combinazioni presentate sono quelle che gli autori hanno trovato nel tempo le più versatili ed efficaci.

L'ideale sarebbe realizzare una distribuzione minimale per i filologi, che il GUIT sta iniziando a progettare, e un pacchetto per migliorare i risultati permessi da parallel e ledpar.

Riferimenti bibliografici

Beccari, C. (2010). *teubner.sty An extension to the greek option of the babel package*.

Braams, J. (2010). *Babel, a multilingual package for use with LATEX's standard document classes*.

Brachet, P. (2011). «Texmaker». URL `http://www.xm1math.net/texmaker/`.

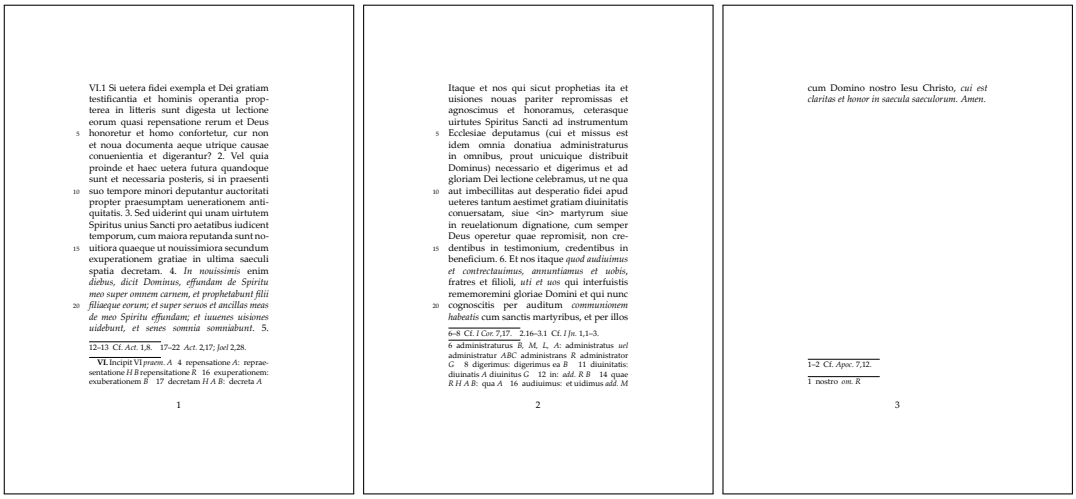
Eckermann, M. (2003). *The Parallel-Package*.

Kew, J. e Löffler, S. (2011). «TEXworks, lowering the entry barrier to the TEXworld». URL `http://www.tug.org/texworks/`.

Leal, J. (2011). «Configurazione della tastiera e dei caratteri greci unicode: come scrivere il greco politonico (greco classico)». URL `http://www.didaskalikos.org/CeTeX/greco/`.

Leal, J. e Pignalberi, G. (2011). *Edizioni Critiche. Guida alla composizione con il proprio computer*. In attesa di pubblicazione.

Pantieri, L. (2008). *L'ARTE DI SCRIVERE IN GRECO CON LATEX*. URL `http://www.lorenzopantieri.net/LaTeX_files/Greco.pdf`.



(a) `recomplex.pdf`, pagina 1 (b) `recomplex.pdf`, pagina 2 (c) `recomplex.pdf`, pagina 3

FIGURA 12: Documento ottenuto dalla compilazione di `recomplex.tex` originale

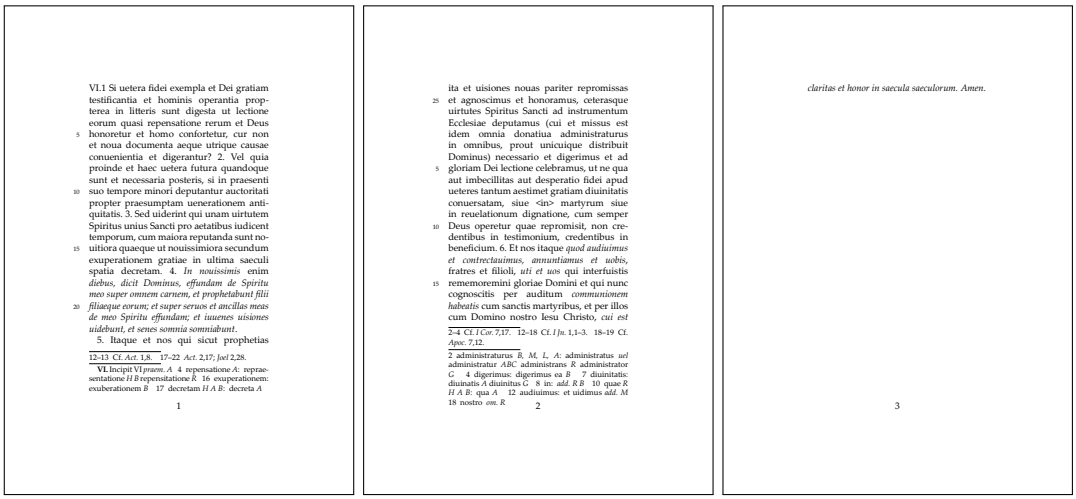


FIGURA 13: Risultato di `recomplex.tex` proposto nella *soluzione 1*: si introduce un capoverso arbitrario, la pagina viene interrotta scorrettamente e la riga interrotta non è giustificata

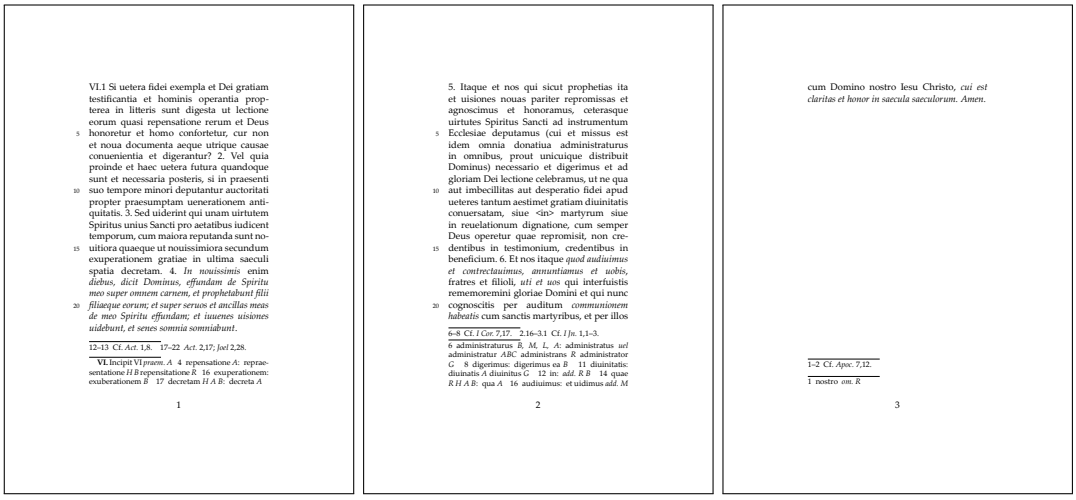


FIGURA 14: Risultato di `recomplex.tex` proposto nella *soluzione 2*: la pagina interrotta correttamente, non c'è il rientro nella riga successiva all'interruzione ma la riga interrotta non è giustificata

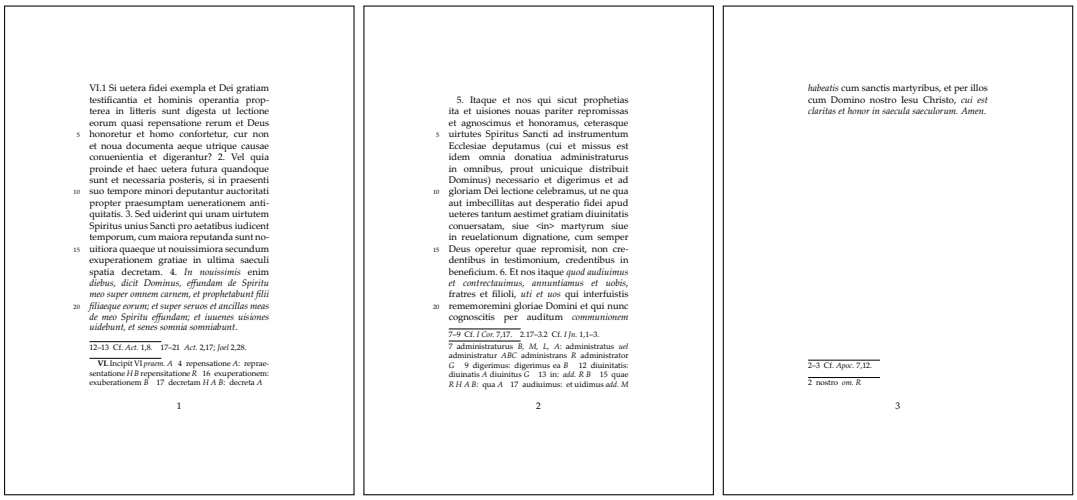


FIGURA 15: Risultato di `recomplex.tex` proposto nella *soluzione 3*: si introduce l'interruzione di pagina correta ma la riga interrotta non viene giustificata e compare un capoverso arbitrario con il rientro della riga successiva all'interruzione

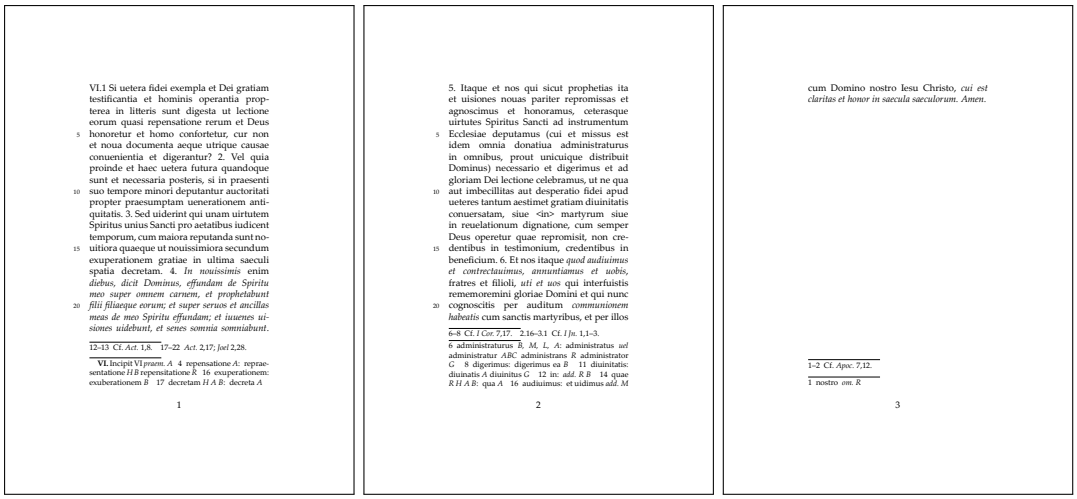


FIGURA 16: Possibile soluzione per suddividere `recomplex.tex` nel punto desiderato (*soluzione 4*)

SIMONIC, A. (2011). «Winedt». URL `http://www.winedt.com/`.

SINAI, G. (2011). «Yudit, the YUnicode eDITor». URL `www.yudit.org`.

VAN DER ZANDER, B. (2011). «TexMakerX: Free cross-platform LaTeX editor». URL `http://texmakerx.sourceforge.net/`.

WEINKAUF, T. e WIEGAND, S. (2011). «TEXnicCenter». URL `http://www.texniccenter.org/`.

WILSON, P. (2005). *Parallel typesetting for critical editions: the ledpar package*.

▷ Gianluca Pignalberi
`g dot pignalberi at alice dot it`

▷ Salvatore Schirone
`schirone at gmail dot com`

▷ Jerónimo Leal
Dipartimento di Storia della Chiesa
Pontificia Università
della Santa Croce
Piazza Sant'Apollinare 49
00186 Roma
`jleal at pusc dot it`

```
\documentclass[12pt, a5paper, oneside]{article}

\usepackage[a5paper,body={8cm,13.5cm}]{geometry}
\usepackage{fontspec}
\usepackage{xunicode}
\usepackage{xltxtra}

\setmainfont[Mapping=tex-text]{Palatino Linotype}

\usepackage{polyglossia}
\setdefaultlanguage{latin}


\usepackage{ledmac}

\renewcommand{\rbracket}{:}

\lefthyphenmin=3
\makeatletter


\newbox\lp@rbox

\newcommand{\ffootnote}[1]{%
  \ifnumberedpar@
    \xright@appenditem{\noexpand\vffootnote{f}{{\l@d@nums}{\@tag}{#1}}}%
      \to\inserts@list
      \global\advance\insert@count by 1
  \fi\ignorespaces}

\newcommand{\gfootnote}[1]{%
  \ifnumberedpar@
    \xright@appenditem{\noexpand\vgfootnote{g}{#1}}%
      \to\inserts@list
      \global\advance\insert@count by 1
  \fi\ignorespaces}

\newcommand{\setlp@rbox}[3]{%
  {\parindent\z@\hsize=2.5cm\raggedleft\scriptsize
   \baselineskip 9pt%
   \global\setbox\lp@rbox=\vbox to\z@{\vss#3}}}

\newcommand{\vffootnote}[2]{\setlp@rbox#2}

\newcommand{\vgfootnote}[2]{\def\rd@ta{#2}}

\renewcommand{\do@line}{%
  {\vbadness=10000 \splittopskip=0pt%
   \gdef\rd@ta{}%
   \global\setbox\one@line=\vsplit\raw@text to\baselineskip}%
   \unvbox\one@line \global\setbox\one@line=\lastbox
   \getline@num
   \hbox to\hsize{\affixline@num\add@inserts\hbox to\z@%
     {\hss\box\lp@rbox\kern\linenumsep}%
     \hfil\hbox to\wd\one@line{\new@line\unhbox\one@line%
     \hbox to\z@{\kern\linenumsep\notenumfont\rd@ta\hss}}}%
  \add@penalties}

\renewcommand{\affixline@num}{%
  \ifsublines@
    \@l@dtempcntb=\subline@num
      \ifnum\subline@num>\c@firstsublinenum
        \@l@dtempcnta=\subline@num
        \advance\@l@dtempcnta by-\c@firstsublinenum
        \divide\@l@dtempcnta by\c@sublinenumincrement
        \multiply\@l@dtempcnta by\c@sublinenumincrement
        \advance\@l@dtempcnta by\c@firstsublinenum
      \else
        \@l@dtempcnta=\c@firstsublinenum
      \fi
                %
      \ifcase\sub@lock
        \or
          \ifnum\sublock@disp=1
            \@l@dtempcntb=0 \@l@dtempcnta=1
```

FIGURA 17: Sorgente completo di `recomplex.tex`

```
            \fi
          \or
            \ifnum\sublock@disp=2 \else
              \@l@dtempcntb=0 \@l@dtempcnta=1
            \fi
          \or
            \ifnum\sublock@disp=0
              \@l@dtempcntb=0 \@l@dtempcnta=1
            \fi
          \fi
        \else
          \@l@dtempcntb=\line@num
          \ifnum\line@num>\c@firstlinenum
            \@l@dtempcnta=\line@num
            \advance\@l@dtempcnta by-\c@firstlinenum
            \divide\@l@dtempcnta by\c@linenumincrement
            \multiply\@l@dtempcnta by\c@linenumincrement
            \advance\@l@dtempcnta by\c@firstlinenum
          \else
            \@l@dtempcnta=\c@firstlinenum
           \fi
           \ifcase\@lock
             \or
               \ifnum\lock@disp=1
                 \@l@dtempcntb=0 \@l@dtempcnta=1
               \fi
             \or
               \ifnum\lock@disp=2 \else
                 \@l@dtempcntb=0 \@l@dtempcnta=1
               \fi
             \or
               \ifnum\lock@disp=0
                 \@l@dtempcntb=0 \@l@dtempcnta=1
               \fi
             \fi
          \fi
               %
          \ifnum\@l@dtempcnta=\@l@dtempcntb
            \@l@dtempcntb=\line@margin
            \ifnum\@l@dtempcntb>1
              \advance\@l@dtempcntb by\page@num
            \fi
            \ifodd\@l@dtempcntb
              \xdef\rd@ta{\the\line@num}%
            \else
              \llap{{\leftlinenum}}%#1%
            \fi
        \else
              %#1%
        \fi
        \ifcase\@lock
        \or
               \global\@lock=2
        \or \or
               \global\@lock=0
        \fi
        \ifcase\sub@lock
        \or
               \global\sub@lock=2
        \or \or
               \global\sub@lock=0
        \fi}

\lineation{page}
\linenummargin{left}
\renewcommand{\footfudgefiddle}{71}

\footparagraph{A}
\footparagraph{B}
\footparagraph{C}

\renewcommand{\notenumfont}{\footnotesize}
\newcommand{\notetextfont}{\footnotesize}

\let\Cfootnoterule=\relax
```

FIGURA 17: Sorgente completo di `recomplex.tex`

```
\addtolength{\skip\Afootins}{1.5mm}

\count\Afootins=825
\count\Bfootins=825
\count\Cfootins=825

\newcommand{\Aparafootfmt}[3]{%
  \ledsetnormalparstuff
  \scriptsize
  \notenumfont\printlines#1|\enspace
  \notetextfont
  #3\penalty-10\hskip 1em plus 4em minus.4em\relax}

\newcommand{\Bparafootfmt}[3]{%
  \ledsetnormalparstuff
  \scriptsize
  \notenumfont\printlines#1|%
  \ifledplinenum
   \enspace
  \else
   {\hskip 0em plus 0em minus .3em}
  \fi
  \select@lemmafont#1|#2\rbracket\enskip
  \notetextfont
  #3\penalty-10\hskip 1em plus 4em minus.4em\relax }

\newcommand{\Cparafootfmt}[3]{%
  \ledsetnormalparstuff
  \scriptsize
  \notenumfont\printlines#1|\enspace
  \notetextfont
  #3\penalty-10\hskip 1em plus 4em minus.4em\relax}

\newcommand*{\previous@A@number}{-1}
\newcommand*{\previous@B@number}{-1}
\newcommand*{\previous@C@number}{-1}
\newcommand*{\previous@page}{-1}

%% NO \rbracket IN FRONT OF om./inv./add. ETC.

\newcommand{\abb}[1]{#1%
        \let\rbracket\nobrak\relax}
\newcommand{\nobrak}{\textnormal{}}
\newcommand{\morenoexpands}{%
        \let\abb=0%
}

\renewcommand{\Bparafootfmt}[3]{%
  \normal@pars\scriptsize
  \parindent=0pt \parfillskip=0pt plus1fil
  \notenumfont\printlines#1|\enspace
  \select@lemmafont#1|#2\rbracket\enskip
  \notetextfont
  #3\penalty-10\hskip 1em plus 4em minus.4em\relax }

%% END DEFINITION OF \abb

\makeatother

\newcommand*{\killnumber}{\linenum{|-1|||-1||}}
\makeatletter
\def\printlines#1|#2|#3|#4|#5|#6|#7|{\begingroup
\ifnum#2=-1 \ledplinenumfalse\fi
 \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
\ifl@d@pnum #1\fullstop\fi
\ifledplinenum \linenumrep{#2}\else \symplinenum\fi
\ifl@d@ssub \fullstop \sublinenumrep{#3}\fi
\ifl@d@dash \endashchar\fi
\ifl@d@pnum #4\fullstop\fi
\ifl@d@elin \linenumrep{#5}\fi
\ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumrep{#6}\fi
\endgroup}

\makeatother
```

FIGURA 17: Sorgente completo di `recomplex.tex`

```
\let\Afootfmt=\Aparafootfmt
\let\Bfootfmt=\Bparafootfmt
\let\Cfootfmt=\Cparafootfmt
\def\lemmafont#1|#2|#3|#4|#5|#6|#7|{\scriptsize}
\parindent=1em

\newcommand{\lmarpar}[1]{\edtext{}{\ffootnote{#1}}}
\newcommand{\rmarpar}[1]{\edtext{}{\gfootnote{#1}}}
\emergencystretch40pt

\begin{document}

\beginnumbering
\pstart

\noindent
VI.1 Si uetera fidei exempla et Dei gratiam testificantia
et\edtext{\abb{}}{\killnumber\Bfootnote{\textbf{VI.} Incipit VI
\textit{praem. A}}} hominis operantia propterea in litteris sunt digesta ut
lectione eorum quasi \edtext{repensatione}
        {\lemma{repensatione \textit{A}}\Bfootnote{repraesentatione
\textit{H B} repensitatione \textit{R}}}
 rerum et Deus honoretur et homo confortetur, cur non et noua documenta
aeque utrique causae conuenientia et digerantur? 2. Vel quia proinde et haec
uetera futura quandoque sunt et necessaria posteris, si in praesenti suo
tempore minori deputantur auctoritati propter praesumptam uenerationem
antiquitatis. 3. Sed uiderint qui unam \edtext{uirtutem Spiritus unius Sancti}
        {\Afootnote{Cf. \textit{Act.} 1,8.}}
 pro aetatibus iudicent temporum, cum maiora reputanda sunt nouitiora quaeque
ut nouissimiora secundum \edtext{exuperationem}
        {\Bfootnote{exuberationem \textit{B}}}
 gratiae in ultima saeculi spatia \edtext{decretam}
        {\lemma{decretam \textit{H A B}}\Bfootnote{decreta \textit{A}}}.
4. \edtext{\textit{In nouissimis} enim \textit{diebus, dicit Dominus,
effundam de Spiritu meo super omnem carnem, et prophetabunt filii filiaeque
eorum; et super seruos et ancillas meas de meo Spiritu effundam; et iuuenes
uisiones uidebunt, et senes somnia somniabunt}}
        {\Afootnote{\textit{Act.} 2,17; \textit{Joel} 2,28.}}.
5. Itaque et nos qui sicut prophetias ita et uisiones
nouas pariter repromissas et agnoscimus et honoramus, ceterasque uirtutes
Spiritus Sancti ad instrumentum Ecclesiae deputamus (cui et missus est
\edtext{idem omnia donatiua \edtext{administraturus}
        {\lemma{administraturus \textit{B, M, L, A}}\Bfootnote{administratus
\textit{uel} administratur \textit{ABC} administrans \textit{R} administrator
\textit{G}}}
 in omnibus, prout unicuique distribuit Dominus)}
        {\Afootnote{Cf. \textit{I Cor.} 7,17.}}
 necessario et \edtext{digerimus}
        {\Bfootnote{digerimus ea \textit{B}}}
 et ad gloriam Dei lectione celebramus, ut ne qua aut imbecillitas aut
desperatio fidei apud ueteres tantum aestimet gratiam \edtext{diuinitatis}
        {\Bfootnote{diuinatis \textit{A} diuinitus \textit{G}}}
 conuersatam, siue <\edtext{in}
        {\Bfootnote{\textit{add. R B}}}>
martyrum siue in reuelationum dignatione, cum semper Deus operetur
\edtext{quae}
        {\lemma{quae \textit{R H A B}}\Bfootnote{qua \textit{A}}}
 repromisit, non credentibus in testimonium, credentibus in beneficium. 6.
Et nos itaque \edtext{\textit{quod} \edtext{\textit{audiuimus}}
        {\lemma{audiuimus}\Bfootnote{et uidimus \textit{add. M}}}}
 \textit{et contrectauimus, annuntiamus et uobis}, fratres et filioli,
\textit{uti et uos} qui interfuistis rememoremini gloriae Domini et qui nunc
cognoscitis per auditum \textit{communionem habeatis} cum sanctis martyribus,
et per illos cum Domino \edtext{\abb{nostro}}
        {\Bfootnote{\textit{om. R}}}
 Iesu Christo}
        {\Afootnote{Cf. \textit{I Jn.} 1,1--3.}},
\edtext{\textit{cui est claritas et honor in saecula saeculorum. Amen.}}
        {\Afootnote{Cf. \textit{Apoc.} 7,12.}}
\pend
\endnumbering
\end{document}
```

FIGURA 17: Sorgente completo di `recomplex.tex`

Un dialogo tra GNU-R e LaTeX: xtable e Sweave

Marco Crivellaro

Sommario

Lo scopo di questo articolo è di mostrare le modalità con cui R, un sofware per il calcolo statistico, si integra con LaTeX per la stesura di reports statistici di alta qualità. L'articolo è frutto della mia esperienza durante la stesura della mia tesi triennale in marketing presso l'Università Ca' Foscari di Venezia.

Abstract

The aim of this article is to show the ways by which R, a sofware for statistical computation, combines itself with LaTeX for the writing of high quality statistical reports. This article is from my experience during the writing of my thesis in marketing at University Ca' Foscari in Venice.

1 Breve introduzione a R

GNU-R (d'ora in avanti solo R) è un software *open-source* disponibile per Windows, Mac OS X e distribuzioni Linux, rilasciato con licenza GPL. Esso è liberamente scaricabile dal sito `http://www.r-project.org/index.html`.

Per le prime due maggiori piattaforme sono disponibili i rispettivi eseguibili, mentre per sistemi Linux è possibile installare tutti i pacchetti necessari attraverso i *repositories* della propria distribuzione preferita. Per gli amanti dell'installazione da terminale, sono disponibili i sorgenti per la compilazione.

R è un programma per la statistica e la grafica. Esso consiste principalmente di un linguaggio e un insieme di librerie che consentono di svolgere molte analisi statistiche, utilizzando fortemente gli strumenti grafici, per la descrizione di un determinato fenomeno come spiegato in Venables *et al.* (2010).

Oltre che un semplice sofware statistico, R può essere tranquillamente definito come un vero e proprio ambiente di programmazione orientato alla gestione e all'analisi dei dati. Nasce nel 1996 per iniziativa di due studiosi, colleghi al Dipartimento di Statistica dell'Università di Aukland: Ross Ihaka e Robert Gentelman. Al tempo, insoddisfatti delle soluzioni fornite dai sofware che utilizzavano, Ihaka e Gentelman elaborarono un linguaggio e un ambiente di programmazione statistico molto simile a S, un linguaggio sviluppato nel 1980 presso i *Bell Laboratories*.

Le caratteristiche principali di R possono essere così riassunte:

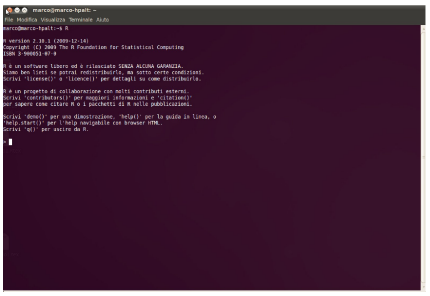


Figura 1: R Console in ambiente Linux

- si compone di un insieme di strumenti per l'analisi statistica dei dati;
- è un linguaggio *object-oriented* pensato per descrivere modelli statistici estremamente complessi;
- permette un'ottima rappresentazione grafica dei dati;
- lavora in generale con matrici e vettori;
- è gratuito.

R viene utilizzato tramite terminale in sistemi Linux, come in figura 1, o attraverso un'interfaccia grafica in Windows o Macintosh. Il comune denomitore tra le varie piattaforme è che per interloquire con le varie funzioni di R si deve *sempre ricorrere alla linea di comando*, contraddistinta dal *prompt* ">". La finestra, o *shell*, con cui l'utente interagisce è chiamata `R Console`.

Per prendere confidenza con R, uno dei comandi che si impara dall'inizio è quello per uscire. Basta digitare nella `Console` dopo il prompt

```
> q()
Save workspace image? [y/n/c]:
```

La successiva domanda ci chiede se salvare il lavoro oppure no.

Per richiamare il programma, basta cliccare due volte sulla relativa icona oppure digitare in un terminale semplicemente R. Prima di continuare con altri comandi, è bene ricordare che R è *case sensitive*: il maiuscolo e il minuscolo hanno significati completamente diversi tra loro. Un'altra cosa da tenere presente è l'uso della *virgola* come separatore tra le varie cifre che inseriamo mentre il *punto* come separatore della parte intera dai decimali. [1]

1. Vi sono casi in cui si può specificare la virgola come separatore dei decimali

Per testare le possibilità grafiche di questo programma, invito a digitare nella `Console` il seguente comando

```
> demo(graphics)
```

mentre se vogliamo spiegazioni su un determinato comando, basta digitare

```
> help(nome comando)
```

`R` svolge le più comuni operazioni matematiche ma lavora in generale con dati strutturati: i più semplici sono gli scalari e i vettori. Bisogna precisare che per "vettore" in `R` non ci riferiamo alla nozione usata nell'algebra lineare, ma ad un insieme indicizzato di oggetti come numeri, stringhe e funzioni.

Per assegnare ad una variabile un certo valore, basta utilizzare il comando "<-". Ad esempio, per assegnare ad `x` il valore 4 scriveremo

```
> x<-4
```

Per sapere cosa contiene `x` basterà digitare

```
> x
[1] 4
```

e `R` ci fornisce la risposta.

Se vogliamo invece creare un vettore `x` di valori, il comando da dare è il seguente

```
> x<-c(4,8,15,16,23,42)
```

Oltre che operare con le comuni matrici, `R` è in grado di gestire egregiamente delle matrici di dati più complesse, ovvero i *dataframe*: le righe rappresentano le unità statistiche oggetto dell'analisi mentre le colonne rappresentano le variabili quantitative e qualitative osservate su ciascuna unità. Per capire come può essere strutturato un *dataframe*, possiamo crearne uno molto semplice nel nostro *workspace*. Prima di tutto trasformiamo il vettore `x` creato precedentemente in una matrice a due colonne con il comando

```
> matrix(x,ncol=2)
```

e diamo `invio`. Ora creiamo il nostro dataframe, che chiameremo `datf`, con il comando `data.frame`

```
>  datf<-data.frame(matrix(x,ncol=2))
```

Per vedere il risultato, basta digitare il nome assegnato

```
> datf
  X1 X2
1  4 16
2  8 23
3 15 42
```

dove `X1` e `X2` rappresentano delle variabili osservate sulle unità statistiche `1`, `2` e `3`.

I *dataframes* possono essere prodotti da software diversi da `R`, su tutti *Microsoft Excel* o simili. `R` gradisce in particolar modo lavorare con file di testo in ASCII standard ma, se non disponiamo di questo tipo di files, è necessario che l'estensione del nostro *dataframe* sia di tipo *csv (Comma Separated Values)*. I comandi che permettono di caricare nel *workspace* queste matrici di dati sono `read.table` e `read.csv`. Per comprendere il loro funzionamento possiamo creare un file `.csv` dal *dataframe* precedente: per farlo utilizzeremo questo comando

```
> write.csv(datf,file="datf.csv",
  row.names=FALSE)
```

che salverà il *file* `datf.csv` nella *directory* di sistema scelta come da impostazioni. [2] `row.names` è impostato su `FALSE` poiché non abbiamo motivo in questa sede di specificare i nomi delle righe.

Per richiamare il nostro *dataframe*, utilizzeremo il comando

```
> read.csv("datf.csv",header=TRUE,
  sep=",")
```

Per un approfondimento dei vari metodi di importazione dei dati vedere Team (2010), Masarotto e Iacus (2007, pag. 313) e Pastore (2005).

Con `R` è possibile inoltre calcolare indici di posizione e dispersione come moda, media e varianza per una data distribuzione di frequenza e svolgere qualsiasi tipo di analisi statistica, su tutte analisi di regressione come spiegato in Masarotto e Iacus (2007).

Quello che ci viene richiesto è di fare un piccolo sforzo mentale per imparare la logica dei vari comandi e di interiorizzarli mediante la pratica: per questo scopo possiamo servirci di files di testo esterni dove copiare per promemoria i comandi che utilizzeremo. Si insiste in ambito accademico, anche se può sembrare un atteggiamento un po' "integralista", a preferire l'uso di questo software per un'analisi staristica rispetto ad altri programmi come, torno a citarlo, *Microsoft Excel*. Il motivo principale risiede nel fatto che il software della Microsoft non è concepito per calcoli di natura statistica mentre `R` ne fa la sua ragion d'essere.

Nel proseguio dell'articolo andremo a spiegare le varie modalità con cui `R` si integra alla perfezione con LaTeX: voglio far notare che opereremo sempre dalla `R Console`.

2 Esportare grafici

I grafici prodotti con `R` rappresentano sicuramente una delle prime cose che vorremmo esportare nel nostro report prodotto con LaTeX. Il comando in `R`,

2. Possiamo anche indicare una cartella precisa specificandone il percorso, facendo attenzione alla sintassi da usare in base al sistema operativo con cui si opera

che genera il grafico dei dati caricati nel workspace, è dato in linea generale da

```
> plot()
```

Questo comando ci consente inoltre di personalizzare il grafico che vogliamo ottenere. In Masarotto e Iacus (2007, pag. 318) troviamo altri comandi più specifici che ci permettono di creare grafici a barre, diagrammi a bastoncini, diagrammi a torta a seconda delle nostre esigenze.

Per capire come procedere dobbiamo ricordarci che R invia l'output dei vari comandi grafici ad una finestra, detta Device, che viene numerata progressivamente. Ricordiamo brevemente che R può restituire i grafici creati in formati di grafica vettoriale, come Postscript o PDF, e in formati *bitmap* come JPEG e GIF. Si consiglia, in funzione delle varie compilazioni che avverranno sul documento, di fare affidamento anche ai primi due formati.

Se si sta lavorando su una piattaforma MacOS o Windows, una volta aperto il device basterà fare un *copia e incolla* o il più blasonato *salva con nome* e il gioco è fatto.

In ambiente Linux invece l'operazione è diversa e richiede l'uso di alcuni comandi. Per attivare il device, al quale inviare l'output grafico, il comando da dare è il seguente

```
> dev.copy(device, file)
```

in cui per:

device si intende il formato che vogliamo ottenere, ossia PDF, postscript o JPEG;

file si intende il nome che vogliamo dare al nostro grafico e eventualmente la cartella in cui lo vogliamo salvare.

Una volta che abbiamo terminato il nostro lavoro e siamo soddisfatti del risultato ottenuto occore digitare

```
> dev.off()
```

per disattivare il device; in altre parole siamo pronti ad inserire il grafico nel report mediante l'ambiente figure. Possiamo creare un semplice grafico partendo dal *dataframe* precedente con il comando

```
> plot(datf,type="b",col=2)
```

Una volta dato invio, compare a schermo un'altra finestra che contiene il grafico prodotto da R. Per salvare il grafico nel formato .eps dobbiamo digitare

```
> dev.copy(postscript,
           file="datf1.eps")
> dev.off()
```

In alternativa al comando dev.copy, possiamo benissimo utilizzare

```
> dev.copy2eps(file="datf.eps")
```

che produce direttamente un file immagine di tipo postscript.

3 Il pacchetto xtable

Come abbiamo visto in precedenza, R lavora spesso con dati strutturati in tabelle come ad esempio i *dataframe*. Altro caso è quando stimiamo un modello di regressione per i dati oggetto della nostra analisi: qui R restituisce una tabella di sintesi del modello stimato.

Per esportare nel report che vogliamo produrre queste tabelle, R ci fornisce un ottimo pacchetto chiamato xtable, letteralmente *Export Tables*: una volta caricato, quando lo invochiamo esso ci restituisce nella R Console il codice LaTeX della tabella che vogliamo esportare.

Come spiegato in Dahl (2009), per prima cosa dobbiamo caricare il pacchetto con il comando

```
> library(xtable)
```

A titolo di esempio, possiamo stimare un modello di regressione lineare semplice partendo dal *dataframe* creato precedentemente. Il comando da dare è il seguente

```
> regmodel<-lm(X2~X1,data=datf)
```

La tabella di sintesi per il modello stimato viene generata semplicemente con

```
> summary(regmodel)
```

Con il comando

```
> xtable(regmodel)
```

ottengo il seguente codice LaTeX:

```
% latex table generated in R 2.10.1
% by xtable 1.5-6 package
\begin{table}[ht]
\begin{center}
\begin{tabular}{rrrrr}
  \hline
 & Estimate & Std. Error & t value
 & Pr($>$$|$t$|$) \\
  \hline
(Intercept) & 5.3710 & 2.5351 & 2.12
     & 0.2808 \\
  X1 & 2.4032 & 0.2514 & 9.56 & 0.0664 \\
   \hline
\end{tabular}
\end{center}
```

pronto da esportare nel nostro report. Ricordiamo brevemente che xtable riesce a gestire vari tipi di oggetti come data.frame, anova, prcomp, table da esportare, nel caso sia necessario, attraverso il pacchetto longtable.

4 Sweave

Il pacchetto Sweave per R è stato sviluppato dal Prof. Friedrich Leisch, attualmente ordinario di statistica presso l'Università Ludwig Maximilians di Monaco. Nelle ultime versioni di R questo pacchetto è già incluso nella distribuzione di base del programma.

Tutto comincia con la creazione di un file "ibrido", avente estensione del tipo `.Rtex`: significa che troviamo dei comandi R all'interno di un documento LaTeX. [3]

Voglio far notare che stiamo scrivendo un file R con all'interno LaTeX e non il contrario. Questo pacchetto si basa sull'idea del *literate programming*: invece di scrivere codice contenente documentazione, si scrive documentazione (in LaTeX) con all'interno del codice (in R). Pertanto per R tutto quello che non è contenuto all'interno di particolari delimitatori viene considerato come commento.

Al termine della stesura del documento, dovremmo precompilare il file sorgente "ibrido" prima nella `R Console` mediante il comando

```
> Sweave("report.Rtex")
```

che produrrà il file `report.tex` da compilare successivamente con `latex` o `pdflatex`.

Come spiegato prima in Himmelmann e Vavassori (2005) e più ampiamente in Leisch (2002), i delimitatori racchiudono porzioni di codice R che vengono chiamate *chunks*. Esse possono essere di due tipi:

noweb i cui file "ibridi" hanno solitamente estensione `.rnw` e comandi `<<>>` e `@` rispettivamente per l'apertura e la chiusura;

SweaveSyntaxLatex molto più personalizzabile e di solito `.Rtex` come estensione del file "ibrido".

Per questa trattazione prendiamo in esame solo il secondo tipo di *chunk*, ovvero `SweaveSyntaxLatex`. I delimitatori che danno forma al *chunk* sono dati da:

```
\begin{Scode}{<opzione1>,<opzione2>}
 ...
\end{Scode}
```

L'ambiente Scode permette l'uso di varie opzioni. Vediamone brevemente alcune:

echo=TRUE mostra il codice sorgente, ovvero i vari comandi R (impostazione di default);

eval=TRUE esegue i comandi R inseriti;

results= è il formato dell'output prodotto dal codice. Opzioni:

- `verbatim`: come in un terminale di R;
- `tex`: in codice LaTeX (da usare nel caso con `xtable`);
- `hide`: non mostra niente.

fig=FALSE mostra la figura che risulterebbe per ogni chunk (default);

fig=TRUE dobbiamo specificare che tipo di file vogliamo con le opzioni:

eps=TRUE immagine prodotta in formato `postscript`;

pdf=TRUE immagine in formato `pdf`.

Per ricapitolare, la procedura per comporre un report mediante Sweave è composta dai seguenti passi:

1. comporre il proprio documento su un file con estensione `.Rtex`;
2. compilare una prima volta dalla `R Console` questo file;
3. il risultato della prima compilazione restituisce il file con estensione `.tex`;
4. compilare con `latex` o `pdflatex` il file `.tex` per ottenere il documento finale.

Nel manuale del pacchetto troverete vari esempi di composizione dei vari tipi di files, così da capire il suo funzionamento. Di seguito riporto alcuni brevi esempi di *chunks* e la loro realizzazione in codice LaTeX dopo la prima compilazione dalla `R Console`.

Per prima cosa vediamo il preambolo:

```
\documentclass[a4paper,10pt]{article}
\usepackage[T1]{fontenc}
\usepackage[italian]{babel}
\usepackage[utf8]{inputenc}
\title{Esempio Sweave}
\author{Me medesimo}
```

Voglio far notare che dopo la prima compilazione, nel preambolo del documento con estensione `.tex` verrà caricato il pacchetto Sweave.

Se vogliamo inserire il vettore e la matrice utilizzeremo i seguenti *chunks*:

```
Creo un vettore x
\begin{Scode}{echo=TRUE,eval=TRUE,
       results=verbatim}
x<-c(4,8,15,16,23,42)
\end{Scode}

e la matrice
\begin{Scode}{echo=TRUE,eval=TRUE,
       results=verbatim}
matrix(x,ncol=2)
\end{Scode}
```

3. Vi sono altri tipi di estensioni.

Dopo la prima compilazione, questi *chunks* compariranno nel file `.tex` così:

```
Creo un vettore x
\begin{Schunk}
\begin{Sinput}
> x<-c(4, 8, 15, 16, 23, 42)
\end{Sinput}
\end{Schunk}

 e la matrice
\begin{Schunk}
\begin{Sinput}
> matrix(x, ncol = 2)
\end{Sinput}
\begin{Soutput}
     [,1] [,2]
[1,]    4   16
[2,]    8   23
[3,]   15   42
\end{Soutput}
\end{Schunk}
```

Vengono così creati degli ambienti Schunk dove viene inserito del codice R secondo le nostre impostazioni.

Per quanto riguarda i grafici, il codice che possiamo utilizzare è il seguente:

```
\begin{Scode}{echo=TRUE,eval=TRUE,
              fig=TRUE,eps=TRUE}
plot(datf,type="b",col=2)
\end{Scode}
```

La prima compilazione dalla `R Console` apre il `Device` grafico e produce il seguente codice LaTeX:

```
\begin{Schunk}
\begin{Sinput}
> plot(datf, type = "b", col = 2)
\end{Sinput}
\end{Schunk}

\begin{figure}[tb]
\centering
\includegraphics{report-004}
\caption{Primo grafico con Sweave}
\end{figure}
```

L'ambiente figure è stato inserito in un secondo momento in quanto, come ben sappiamo, consente una migliore gestione dell'impaginazione della figura e ci permette di inserire una didascalia.

Infine per inserire la tabella di sintesi per il modello di regressione semplice riportato negli esempi precedenti utilizzeremo:

```
\begin{Scode}{echo=TRUE,eval=TRUE,
              results=tex}
library(xtable)
xtable(regmodel)
\end{Scode}
```

La compilazione produrrà il seguente codice LaTeX:

```
\begin{Schunk}
\begin{Sinput}
> library(xtable)
> xtable(regmodel)
\end{Sinput}
% latex table generated in R 2.10.1
  by xtable 1.5-6 package
% Fri Aug 12 16:03:14 2011
\begin{table}[ht]
\begin{center}
\begin{tabular}{rrrrr}
  \hline
 & Estimate & Std. Error & t value
 & Pr($>$$|$t$|$) \\
  \hline
(Intercept) & 5.3710 & 2.5351 & 2.12
            & 0.2808 \\
  X1 & 2.4032 & 0.2514 & 9.56 & 0.0664 \\
   \hline
\end{tabular}
\end{center}
\end{table}
\end{Schunk}
```

5 Conclusioni

GNU-R, come LaTeX, è un programma che richiede una certa "dedizione" in una fase iniziale per comprenderne la logica. Mano a mano che lo si utilizza anche per svolgere semplici calcoli esso diventa sempre più familiare e intuitivo. Il *dialogo* che riesce ad instaurare con il miglior programma di composizione tipografica produce risultati di elevatà qualità. Tutto questo è sempre inquadrato in quell'ottica del *privilegiare il contenuto*, attraverso quella passione di chi vuole comunicare e spiegare qualcosa.

6 Ringraziamenti

Volevo ringraziare Gianluca Pignalberi per avermi dato l'oppurtunità di scrivere per questa prestigiosa rivista e soprattutto gli editor per i preziosi consigli e suggerimenti dati per questo mio primo articolo.

Riferimenti bibliografici

Dahl, D. B. (2009). *Export tables to LaTeX or HTML*. `http://cran.r-project.org/web/packages/xtable/index.html`.

Himmelmann, M. W. e Vavassori, E. G. (2005). «Generazione automatica di report con R e LaTeX». In *Convegno Nazionale su TEX, LATEX e Tipografia Digitale*. Pisa.

Leisch, F. (2002). *Sweave User Manual for R Version 2.7.1*. http://www.stat.uni-muenchen.de/~leisch/Sweave/.

Masarotto, G. e Iacus, S. M. (2007). *Laboratorio di Statistica con R*. McGraw-Hill, Milano.

Pastore, A. (2005). «Creazione di Dataframes in R». http://venus.unive.it/pastore/.

Team, T. R. D. C. (2010). *R Data Import/Export*. http://cran.r-project.org/manuals.html.

Venables, W. N., Smith, D. M. e the R Development Core Team (2010). *An introduction to R*. http://cran.r-project.org/manuals.html.

▷ Marco Crivellaro
Via Anacleto Rossati, 6
45011 Bottrighe - Adria
Rovigo
settimo99 at alice dot it

Uno strumento per gestire progetti LaTeX cooperativi

Giovanni Mascellani

Sommario

Utilizzando e modificando strumenti liberi e a sorgente aperto già esistenti, abbiamo realizzato un portale che permette a studenti di collaborare nella scrittura di appunti e testi di studio in LaTeX. In particolare, è disponibile uno strumento automatico che compila i sorgenti e rende facilmente accessibili i PDF direttamente dal sito Internet.

Abstract

Using and modifying already available free and open source tools, we made a portal that students can use to collaborate in writing notes and texts with LaTeX. It is also features an automatic tool that compiles the sources and makes the PDF files easily reachable directly from the Internet site.

1 Introduzione

L'importanza che oggi ha LaTeX nelle università (ed in particolare nelle facoltà scientifiche) è molto probabilmente fuori dubbio: alcuni corsi di laurea prevedono normalmente, nei primi anni di studio, appositi corsi o perlomeno qualche lezione di introduzione a questo splendido sistema. In ogni caso la maggioranza degli allievi è costretta al fatidico incontro verso la fine del triennio di studio, per la scrittura della tesi.

L'incontro, normalmente, non è eccessivamente traumatico, nonostante le difficoltà ed i brutti scherzi che LaTeX e TeX ogni tanto fanno: chiunque si trovi nel bisogno di scrivere un testo nel quale la matematica giochi un ruolo non marginale non ci mette molto ad apprezzare l'eccellente capacità di resa, in particolare proprio nelle formule matematiche. Ne segue che spesso gli studenti iniziano a scrivere in LaTeX ben prima di essere costretti dalla tesi: ne approfittano per sistemare e condividere appunti per i corsi che seguono; non è infrequente, addirittura, che, scrivendo ad un docente, la poca flessibilità di espressione che si ha nel testo di un'email spinga piuttosto a scrivere la domanda in un documento LaTeX ed inviare questo in allegato (per poi, magari, vedersi rispondere nello stesso modo).

Gli studenti di matematica dell'università di Pisa hanno già da qualche anno una tradizione piuttosto consolidata di appunti condivisi in LaTeX, che per un certo numero di corsi vengono ad avere un'importanza non minore dei libri indicati dal docente.

Inutile, poi, dire che scrivere o revisionare appunti è uno dei modi migliori per studiare un corso, poiché richiede di verificare la propria comprensione dei contenuti appresi e di riscriverli e sistemarli in una forma più chiara possibile. Si tratta di un'attività che può anche essere fatta in forma collaborativa: più persone che lavorano sullo stesso documento possono confrontarsi e chiarirsi le idee a vicenda e sperimentare modi diversi per esporre quanto si vuole mettere su carta.

Condividendo e volendo dare più spazio possibile a queste iniziative, io ed alcuni miei colleghi abbiamo pensato di realizzare un piccolo sistema che aiutasse questo lavoro di collaborazione e condivisione. Non c'era bisogno di inventare nulla, ma solamente mettere insieme alcuni strumenti già esistenti e collaudati ("incollandoli" con qualche riga di codice nostro). Fortunatamente tutti gli strumenti con i quali volevamo lavorare sono software libero, dunque non abbiamo avuto problemi a metterli insieme. Ed il risultato è a sua volta disponibile, altrettanto liberamente, per chi avesse voglia di installarlo su un suo server locale.

2 Tecnologie e programmi

2.1 Git: un sistema di gestione delle versioni

Il primo livello necessario è quello della gestione del codice: ognuno dei collaboratori che lavorano su un certo documento deve essere in grado di lavorare indipendentemente per poi condividere il proprio lavoro con gli altri automaticamente. In genere è anche molto importante salvare tutte le versioni intermedie del gruppo di file (in gergo, *repository*) su cui si sta lavorando, in modo che in ogni momento si possa consultare lo storico delle modifiche (detto *log*), rivedere i cambiamenti fatti e correggere gli errori introdotti. Spesso sistemi di questo tipo (detti VCS, ossia *version control system*) sono progettati per contenere il codice di un programma, ma sono perfettamente adatti anche per gestire codice LaTeX.

Di sistemi di questo tipo ne esistono tanti, con caratteristiche anche molto diverse gli uni degli altri. Un nome storico è CVS (*concurrent version system*); un altro sistema, ancora molto diffuso oggi, è Subversion. Già da parecchi anni, tuttavia, si sta affermando una nuova categoria di VCS, che

sono detti "distribuiti": infatti in questi sistemi non vi è una copia centrale del repository, più importante delle altre, alla quale tutte le altre fanno riferimento. Ogni copia è completamente indipendente, anche se le modifiche fatte possono essere facilmente trasferite dall'una all'altra. Questo ovviamente non significa che i vari collaboratori non possano accordarsi per dichiarare una di queste copie quella "canonica", ma semplicemente vuol dire che tale decisione è una convenzione di metodo, non tecnica. Tra i VCS distribuiti troviamo, per esempio, Bazaar, Mercurial, Darcs e Git, che è quello che abbiamo deciso di utilizzare nel nostro progetto.

Chiaramente non è questo il luogo per discutere le differenze tra i vari sistemi ed i motivi per scegliere uno oppure l'altro. Di motivi per orientare una tale scelta ne possono esistere tanti, tra i quali un ruolo per nulla trascurabile è dato dal gusto e dalle abitudini dell'utente. Basti dire, in favore della nostra scelta, che Git è uno dei sistemi più usati al mondo, in particolare è il sistema che gestisce il codice del kernel Linux (in effetti è stato sviluppato apposta per tale progetto).[1] Chi vuole approfondire l'argomento troverà su Internet tantissimi siti che offrono opinioni e confronti sull'argomento.

2.2 Gitorious: un'interfaccia Web a Git

Git integra già dentro di sé tutte le funzionalità necessarie per trasferire contenuti via rete da una copia ad un'altra di un repository; tuttavia per poter gestire un grande numero di repository (ciascuno dei quali associato ad un diverso progetto o documento) ed un grande numero di utenti che vi lavorano è molto utile avere un sistema che semplifichi le procedure di amministrazione. Un sistema di questo tipo deve:

- permettere agli utenti registrati di gestire i propri repository e le proprie preferenze; in particolare per ogni repository deve essere possibile assegnare privilegi di accesso diversi ad utenti diversi;
- permettere ai visitatori (anche non registrati sul sistema) di accedere al contenuto dei repository anche senza dover installare Git e clonare una copia del repository.

Gitorious è uno di questi sistemi: oltre ad essere disponibile come servizio Web gratuito all'indirizzo `http://www.gitorious.org/`, è possibile scaricarne i codici sorgenti ed installarlo su una macchina propria. L'interfaccia Web esposta permette la registrazione degli utenti e la creazione di repository, con i quali si può poi interagire con il programma Git (eventualmente tramite una delle tante interfacce grafiche disponibili).

1. Per i curiosi, i sorgenti di LaTeX sono conservati in un repository Subversion.

2.3 Compilatex: un sistema per compilare automaticamente i documenti

L'interfaccia Web di Gitorious è molto comoda per consultare i sorgenti contenuti all'interno dei repository ospitati. Tuttavia, per il visitatore che è solamente interessato ad ottenere una copia del documento, è necessario anche avere la possibilità di scaricare direttamente il file PDF, già compilato e pronto per la consultazione.

Per fare questo abbiamo scritto ex-novo un piccolo programma in Python (chiamato "Compilatex") che mantiene una copia di ciascuno dei repository indicati in un file di configurazione e, su richiesta, compila i relativi PDF. Ciascun PDF viene mantenuto in una memoria temporanea, in modo da non richiedere una nuova compilazione, perlomeno finché il relativo repository non venga modificato. La compilazione viene gestita tramite il programma `rubber`, che si preoccupa di eseguire autonomamente schemi di compilazione complessi (richiesti, per esempio, quando è presente un indice generale oppure una fonte bibliografica realizzata con BibTeX; in questi casi la compilazione richiede di eseguire vari comandi diversi in un certo ordine).

È possibile anche configurare uno o più indirizzi email per ogni repository: in caso di errore nella compilazione, Compilatex è in grado di inviare un'email all'indirizzo relativo in modo da notificare il problema. Nel frattempo dall'interfaccia Web viene servita l'ultima versione compilabile del documento richiesto, finché il problema non viene risolto.

3 Come utilizzare il portale

Naturalmente non è questa l'occasione per addentrarmi nei dettagli tecnici del funzionamento complessivo del sistema che abbiamo realizzato, che non riguardano tanto il mondo del TeX, quanto quello della programmazione.

Preferisco, piuttosto, spiegare brevemente il funzionamento del portale dal punto di vista dell'utente.

3.1 La struttura generale

Il portale è raggiungibile al sito `http://git.phc.unipi.it/`. La pagina introduttiva di Gitorious mostra immediatamente la lista delle ultime attività eseguite sul sito da parte dei vari autori. Tramite i pulsanti in alto a destra è possibile passare alla lista dei progetti disponibili, filtrare la lista tramite dei *tag* e fare ricerche con testo libero.

All'interno di ogni progetto possono esserci uno o più repository, i quali contengono effettivamente il codice LaTeX. La possibilità di avere più di un repository associato allo stesso progetto potrebbe permettere di raggruppare insieme documenti che trattano di argomenti simili, ma finora (soprattutto per mancanza di un chiaro schema da utilizzare) questa possibilità è stata utilizzata molto poco.

Accedendo alla pagina di un repository è possibile consultare il log delle ultime modifiche (a ciascuna modifica è anche associata una breve descrizione) e l'albero dei file contenuti all'interno del repository. Se il repository è configurato per l'utilizzo con Compilatex, sulla destra comparirà anche un pulsante per scaricare il file PDF compilato. La stessa pagina riporta, inoltre, l'URL tramite la quale è possibile clonare il repository sulla propria macchina. Non essendo possibile in questa sede introdurre l'uso di Git, rimandiamo gli interessati ai numerosi manuali e tutorial in rete (per esempio Lynn).

3.2 Interagire tramite Git

Al momento è possibile iscriversi liberamente e gratuitamente al portale tramite i link presenti in alto a destra (questo potrebbe cambiare se, in futuro, un numero eccessivo di utenti richiedesse al nostro server troppe risorse). Una volta registrato, un utente può creare nuovi progetti e repository ed essere aggiunto come collaboratore (dotato, quindi, di diritti di scrittura) ai repository già esistenti.

Quando un utente accede alla pagina di un repository per il quale ha diritti di scrittura, la barra che riporta l'URL tramite la quale clonare il repository permette due possibilità diverse: la prima, identificata con "GIT", è la stessa già descritta prima e permette l'accesso unicamente in lettura; la seconda, identificata con "SSH", permette l'accesso sia in lettura che in scrittura. Questa seconda possiblità sfrutta, appunto, il protocollo SSH (*secure shell*): si tratta di un sistema molto diffuso per accedere ed inviare comandi ad un computer remoto. Come il nome suggerisce, grazie a vari sistemi di crittografia a chiave pubblica, è in grado di gestire molti aspetti di sicurezza della connessione: l'autenticazione del client verso il server, il riconoscimento del server da parte del client e la crittografia del contenuto della connessione. Per non dover riscrivere da capo queste funzioni (piuttosto delicate da implementare correttamente), gli autori di Git hanno preferito affidarsi ad un prodotto già noto e consolidato.

Per utilizzare il protocollo SSH ed accedere al repository anche in scrittura è necessario utilizzare uno schema di autenticazione a chiave pubblica ed aggiungere le chiavi che si intende utilizzare nella sezione "Manage SSH keys" raggiungibile dalla propria pagina utente su Gitorious. Una buona introduzione si può trovare in Ubuntu SSH; è pensata per gli utenti Ubuntu, ma si adatterà senza grosse difficoltà a tutti i sistemi operativi basati su Linux ed in buona parte anche al sistema Macintosh. Per gli utenti Windows esistono varie implementazioni sia di Git che di SSH, per cui sarà necessario fare di volta in volta riferimento alla documentazione del programma che si usa.

Una volta che le chiavi pubbliche sono state configurate è possibile clonare il repository tramite SSH, modificarlo e poi rimandare sul server le modifiche, che compaiono immediatamente sul sito.

3.3 Utilizzare Compilatex

La possibilità di offrire PDF già compilati tramite Compilatex non è immediatamente disponibile per i nuovi repository creati, ma richiede l'intervento manuale degli amministratori del portale. Questo per due motivi: primo, la compilazione di un file LaTeX può richiedere molte risorse di sistema e può essere un vettore di attacco alla macchina che lo ospita; secondo, per poter utilizzare Compilatex sono necessarie alcune configurazioni da fare per forza manualmente.

Per attivare l'utilizzo di Compilatex su un repository è dunque necessario mandare un'email all'indirizzo degli amministratori (l'indirizzo è indicato sulle FAQ del portale), specificando il repository per il quale si sta facendo la richiesta. Normalmente l'attivazione viene fatta entro pochi giorni.

4 Stato attuale

Lo sviluppo di Compilatex e delle nostre modifiche a Gitorious si è svolto all'incirca nell'ultima settimana di ottobre. Qualche giorno più tardi il portale è stato reso pubblico e nelle settimane successive una buona parte degli appunti già disponibili sono stati migrati sul nuovo sito.

Nel frattempo è stato organizzato, a cura di alcuni studenti di matematica e di fisica, un breve corso rivolto ad altri studenti interessati ad imparare come utilizzate LaTeX ed il nuovo portale. Alcuni studenti o gruppi di studenti si sono poi organizzati ed hanno iniziato a copiare in LaTeX i loro appunti presi a lezione ed a tenerli sul nostro sistema: alcuni hanno iniziato a lavorare su nuovi documenti, altri hanno recuperato documenti già presenti (ma magari incompleti, da sistemare o da aggiornare) ed hanno continuato il lavoro di chi li aveva iniziati.

Dopo qualche mese il lavoro fatto non è stato certo poco: inutile negare che molti dei documenti che sono stati aggiunti sono poi rimasti incompleti, scritti di fretta oppure con poca conoscenza tecnica del LaTeX. Ritengo, tuttavia, che uno dei pregi di questo modello di scrittura di testi ed appunti sia quello di permettere ogni anno a coloro che seguono un corso di migliorare qualcosa: se l'anno precedente chi ha steso gli appunti non è riuscito a terminare il lavoro, chi ci lavorerà quest'anno avrà questa possibilità, beneficiando della parziale opera già compiuta dal predecessore. L'anno prossimo sarà possibile rivedere ulteriormente gli appunti, rifinirli nei dettagli e, con un po' di buona volontà, portarli ad uno stato molto vicino ad un libro da stampare.

Al momento gli utenti registrati ed attivi sul sistema sono quasi tutti studenti di matematica di

Pisa. Vi è poi qualche studente di fisica presso la stessa università e un paio di altre persone da altre università. Nell'opinione mia e di coloro che hanno collaborato alla realizzazione di questo servizio, un sistema di questo tipo può essere molto comodo per molti altri studenti: noi siamo disponibili ad ospitare repository di università o corsi di laurea diversi dal nostro, fermo restando che, comunque, chiunque può utilizzare il nostro codice per mettere in funzione un altro sistema analogo su un proprio server.[2]

Come già detto, infatti, tutti i programmi utilizzati sono software libero: sulla nostra istanza di Gitorious vi è un progetto apposta (PHC Gitorious) per distribuire il codice sorgente di Compilatex e delle modifiche che abbiamo fatto al software Gitorious.

5 Conclusioni e ringraziamenti

Desidero porgere le mie scuse a tutti i lettori rimasti delusi per il fatto che sono stato costretto a tralasciare le ampie parti di questo articolo, che ci avrebbero portati in tecnicismi che non riguardano TEX e sarebbero andati oltre lo scopo dichiarato della rivista. Oltre che rimandare ai r iferimenti dati in bibliografia, desidero invitare chiunque avesse voglia di approfondire le tematiche (sia tecniche che sociali e didattiche) trattate in questo articolo a contattarmi.

A conclusione di questo articolo, desidero ringraziare Leonardo Robol, con il quale ho collaborato per realizzare questo progetto, ed i tanti altri miei colleghi che hanno incoraggiato me e Leonardo durante il lavoro. Aggiungo anche un ringraziamento per Luigi Amedeo Bianchi per il lavoro di revisione dell'articolo.

Riferimenti bibliografici

Lynn, B. «Git magic». URL `http://www-cs-students.stanford.edu/~blynn/gitmagic/`.

PHC Gitorious. «Progetto "PHC Gitorious"». URL `http://git.phc.unipi.it/phc-gitorious`.

Ubuntu SSH. «SSH documentation for Ubuntu». URL `http://help.ubuntu.com/community/SSH`.

▷ Giovanni Mascellani
Studente di matematica
all'università di Pisa
`mascellani at poisson dot phc dot unipi dot it`

2. Va detto che l'operazione di installare lo stesso software su un altro server non è del tutto immediata. Purtroppo non abbiamo le energie per curare una distribuzione completa: pubblichiamo le modifiche che facciamo, ma chi le vuole utilizzare dovrà occuparsi di configurarle opportunamente in funzione delle proprie esigenze.

Passare da LaTeX a XSL-FO*

Jean-Michel Hufflen

Sommario

Questo articolo si propone di introdurre gli utenti di LaTeX a XSL-FO. Il suo scopo non è quello di dare una panoramica esauriente di XSL-FO, ma di consentire all'utente di muovere i primi passi. Sono infine mostrate somiglianze e differenze tra questi due diversi approcci alla composizione di testi.

Abstract

This article aims at introdicung LaTeX users to XSL-FO. It does not attempt to give an exhaustive view of XSL-FO, but allows a LaTeX user to get started. We show the common and different points between these two approaches of text typesetting.

1 Introduzione

Anche se il formato XML[1] è largamente usato per lo scambio e il trattamento di strutture assimilabili a basi di dati, questo meta-linguaggio ha le sue lontane origini nella gestione di documenti di testo. Non è perciò sorprendente che la "cassetta degli attrezzi" di XML fornisca un linguaggio — XSL-FO[2] — per descrivere composizioni tipografiche di alta qualità. XSL-FO e LaTeX hanno numerosi punti in comune: in particolare, né l'uno né l'altro sono interattivi.[3]

Esistono diversi buoni manuali per l'apprendimento di XSL-FO; per esempio Ammane Rigaux (2002, cap. 8) in francese, Pawson (2002) in inglese, Montero Pinedae Krüger (2004) in tedesco,[4] ma ci sembra che sia relativamente semplice per un utente di LaTeX imparare le basi di XSL-FO e questo è lo scopo del presente articolo. Come vedremo tra breve, questo formato è estremamente verboso, ma in realtà non è stato pensato per essere direttamente scritto dall'utente. Tuttavia è proprio ciò che faremo nel paragrafo 2 con un piccolo esempio[5] volto a illustrarne le basi. In seguito mostreremo brevemente nel paragrafo 3 in che modo è stata prevista la scrittura di documenti contenenti diverse lingue e perfino diversi sistemi di scrittura. Infine, in un'ottica di separazione di forma e contenuto, i testi in XSL-FO sono spesso costruiti a partire da un file XML che contiene solo testo "grezzo", senza indicazioni di formattazione e applicando a esso un foglio di stile XSLT[6] (W3C, 2010), il linguaggio utilizzato per la trasformazione di testi XML. Toccheremo questo punto al paragrafo 4. Naturalmente, questa breve introduzione non è in grado di sostituire né le più voluminose opere didattiche citate in precedenza, né le specifiche W3C (W3C, 2006b), alle quali invitiamo il lettore curioso a fare riferimento per maggiori dettagli. Quanto ai comandi LaTeX che verranno menzionati nel corso di questo articolo, essi si trovano documentati nel *LaTeX Companion*, nella versione originale (Mittelbach et al., 2004).

2 Primi passi

2.1 Nozioni di base

Gli utenti di LaTeX hanno familiarità con la nozione di "classe di documento", la nozione equivalente fornita da XSL-FO è quella di **modello di pagina**. Quello che proponiamo nella figura 1 è molto semplice: una sola costruzione di pagina, specificata dall'elemento `fo:simple-page-master`, i cui attributi forniscono il formato del foglio e le dimensioni dei margini, dove non può essere stampato niente. Seguono le definizioni delle *regioni*, la cui disposizione è mostrata nell'illustrazione della figura 2. È bene notare che la terminologia evita per disegno di ricorrere ai termini "alto", "basso", "sinistra" e "destra" per riferirsi alle regioni, perché la loro posizione dipende in realtà dal sistema di scrittura:[7] così, il senso della

*L'articolo originale è apparso in Hufflen (2007). In seguito questa versione è stata tradotta in francese e estesa, per essere pubblicata sul numero 51 di Cahiers GUTenberg. La traduzione in italiano, comprendente qualche aggiornamento ad opera dell'autore rispetto all'originale, è di Massimiliano Dominici, e compare su ArsTeXnica con il permesso dell'autore e di Tierry Bouche, direttore di Cahiers Gutenberg.

1. *Extensible markup language.* In questo articolo daremo per scontata la conoscenza delle basi di questo meta-linguaggio. Un buon testo di riferimento per una introduzione a XML è Ray (2001).

2. Extensible stylesheet language-Formatting objects.

3. Non sono WYSIWYG (What you see is what you get) come si direbbe oltremanica.

4. Tutti questi documenti sono relativi alla versione 1.0 di XSL-FO (W3C, 2001), mentre la raccomandazione del W3C (*World Wide Web consortium*) più recente è la versione 1.1 (W3C, 2006b). I cambiamenti, tuttavia, sono minimi. Si veda W3C (2006b, § E). Segnaliamo comunque che il W3C ha cominciato a lavorare a una revisione del linguaggio W3C (2010), le cui motivazioni sono illustrate in W3C (2008).

5. Tutti i testi e i programmi di questo articolo, insieme ad alcune versioni alternative e a esempi supplementari, sono disponibuli sul *Web*, all'indirizzo `http://lifc.univ-fcomte.fr/home/~jmhufflen/texts/xsl-fo/`.

6. Extensible stylesheet language transformations.

7. Dato dall'attributo `writing-mode`, di cui riparleremo nel paragrafo 3.

```
<fo:layout-master-set xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:simple-page-master master-name="simple-page"
                         page-height="297mm" page-width="210mm"
                         margin-top="10mm" margin-bottom="20mm"
                         margin-left="25mm" margin-right="25mm">
    <fo:region-body margin-top="20mm" margin-bottom="20mm"
                    margin-left="20mm" margin-right="20mm"/>
                              <!-- (... eventualmente l'attributo column-count per scrivere su più colonne.) -->
    <fo:region-before extent="20mm"/>
                                                    <!-- È data soltanto la larghezza delle altre regioni. -->
    <fo:region-after extent="20mm"/>
                                 <!-- Se non si definiscono dei margini per la regione «body», ciò crea degli -->
                                                    <!-- effetti di sovrapposizione con le regioni vicine. -->
    <fo:region-start extent="15mm"/>
    <fo:region-end extent="15mm"/>
  </fo:simple-page-master>
</fo:layout-master-set>
```

FIGURA 1: Esempio di modello di pagina in XSL-FO.

scrittura procede dalla regione "start" alla regione "end". A titolo di esempio, nella figura 2 abbiamo riprodotto il posizionamento delle regioni per le scritture latina e semitica (usata da lingue quali l'ebraico o l'arabo). L'ordine di dichiarazione per le regioni è fisso: `fo:region-body`, poi `fo:region-before`, `fo:region-after`, `fo:region-start`, `fo:region-end`, tutti elementi facoltativi, tranne il primo.[8] L'elemento `fo:simple-page-master` ammette inoltre un attributo `reference-orientation` che è un angolo misurato in gradi: esso permette in particolare di realizzare una presentazione in formato *landscape* se impostato al valore `90` invece che `0`; quest'ultimo è il valore per difetto.[9]

Finché ci si limita a definire un unico modello di pagina — come nella figura 1 —, ciò significa che tutte le pagine del documento, qualunque sia il loro numero, saranno formattate secondo questo modello. La definizione di *modelli di sequenze*, tramite l'elemento `fo:page-sequence-master`, permette la limitazione del numero di pagine del documento, così come la combinazione di diversi modelli di pagina; per esempio, è possibile combinare due modelli per le pagine pari e dispari o definire modelli separati per la prima e l'ultima pagina di un documento.

La figura 4 mostra ciò che si ottiene processando il sorgente della figura 3. Possiamo notare che i modelli di pagina non vi sono definiti tramite l'inclusione di un file, come avviene con LaTeX. Se si desidera che un modello di pagina sia riutilizzato da nmerosi documenti, la soluzione è quella di usare una *entità esterna* (Ray, 2001), cosa che implica l'introduzione di un elemento `DOCTYPE` artificiale, in quanto non ci si riferisce realmente a un tipo di documento.[10]

Come è mostrato dalla figura 3, la radice di un testo XSL-FO è l'elemento `fo:root`, i cui figli sono un modello di pagina e una sequenza di pagine.[11]

Una sequenza di pagine definisce *cosa* scrivere e *dove*. Nella figura 3 è mostrato come associare un *contenuto statico* — in questo caso un titolo, seguito dal numero della pagina — a un piè di pagina, e come definire un *flusso* che scorra su regioni che possono appartenere a pagine successive. Un flusso è legato a una regione per mezzo dell'attributo `flow-name`, che si riferisce a una regione. Spesso vengono usate convenzioni implicite; così la definizione della regione "body" della figura 1 equivale a:

```
<fo:region-body
  region-name=xsl-region-body/>
```

2.2 Formattare del testo

In prima approssimazione, un elemento `fo:block` di XSL-FO si utilizza per un capoverso, che in TeX è terminato dal comando `\par`. In effetti, questi "blocchi" assomigliano più agli ambienti `minipage` di LaTeX, in quanto possono essere annidati. Gli attributi `color` e `background-color` sono utilizzati per il colore del testo e dello sfondo. Altri attributi — `border-style`, `border-width`, `border-color`,

8. In contraddizione con quanto riportato in Hufflen (2007). In effetti, la figura 2 di quell'articolo fu correttamente trattata da alcuni processori XSL-FO che sono tolleranti su questo punto; essa è comunque non valida, mentre l'ordine corretto è quello indicato dalla figura 1 del presente articolo.

9. Una spiegazione dettagliata degli effetti dati dalle diverse combinazioni di `writing-mode` e `reference-orientation` figura in Pawson (2002, pp. 37–39).

10. Si tratta di un modo per aggirare il problema. Un metodo più proprio sarebbe quello di usare degli elementi appartenenti a XInclude (W3C, 2006a). Versioni recenti degli strumenti menzionati qui di seguito — FOP apachefop 2011 e Saxon (Kay, 2004, App. E) (cf. § 4 e 5) — possono usare questi elementi, ma solo per mezzo di complicate opzioni.

11. Tra questi elementi obbligatori possono essere intercalati due elementi facoltativi che servono per dichiarazioni globali, (W3C, 2006b, § 6.3.3) (si tratta, in effetti, di dichiarazioni di colore) e per la specificazione di *segnalibri* (W3C, 2006b, § 6.11.1).

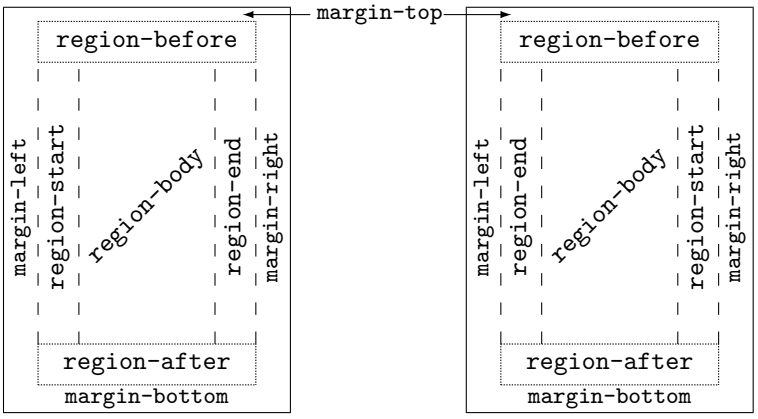


FIGURA 2: Regioni definite da XSL-FO per le scritture latine e semitiche.

```
<!DOCTYPE root [<!ENTITY en-dash "&#x2013;">
                <!ENTITY layout SYSTEM "layout.fo">]>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  &layout;
  <fo:page-sequence master-reference="simple-page" font-family="serif"
                    font-size="12pt" text-align="start">
    <fo:static-content flow-name="xsl-region-after">
      <fo:block text-align="center" line-height="14pt" color="green"
                font-size="10pt" font-family="serif" xml:lang="it">
        Ballo delle Ingrate (<fo:page-number/>)
      </fo:block>
    </fo:static-content>
    <fo:flow flow-name="xsl-region-body">
      <fo:block font-family="sans-serif" font-size="18pt"
                font-variant="small-caps" padding-top="3pt"
                text-align="center" color="white"
                background-color="blue" space-after="15pt" line-height="24pt">
        Ballo delle Ingrate
      </fo:block>
      <fo:block font-family="sans-serif" font-size="14pt" space-after="18pt"
                border-style="solid" border-width="0.5mm" border-color="blue"
                padding="4mm" start-indent="80mm" end-indent="4mm">
        <fo:block text-align="end">
          Ottavio Rinuccini
          <fo:inline font-style="italic">(1562&en-dash;1621)</fo:inline>
        </fo:block>
      </fo:block>
      <fo:block space-before.minimum="10pt" space-before.optimum="11pt"
                space-before.maximum="12pt">
        De l'implacabil Dio
      </fo:block>
      <fo:block>Eccone giunt'al Regno,</fo:block>
      <fo:block>Seconda, O bella Madre, il pregar mio.</fo:block>
      <fo:block (stessa spaziatura dell'elemento precedente) >Non tacerà mia voce</fo:block>
      <fo:block>Dolci lusinghe e prieghi</fo:block>
      <fo:block>Finche l'alma feroce</fo:block>
      <fo:block>Del Re severo al tuo voler non pieghi.</fo:block>
    </fo:flow>
  </fo:page-sequence>
</fo:root>
```

FIGURA 3: Esempio di un sorgente XSL-FO.

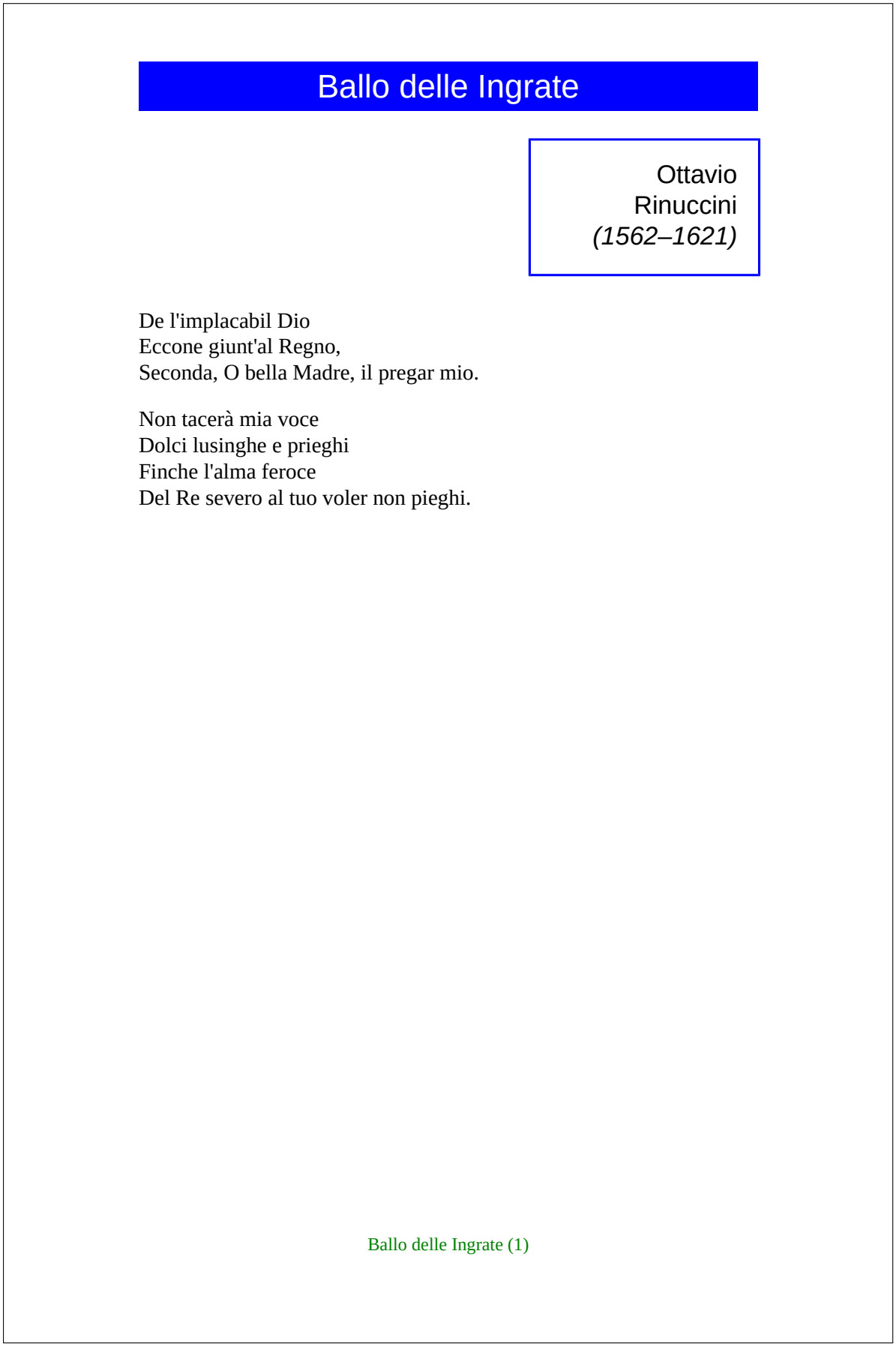


FIGURA 4: Il testo della figura 3, formattato.

... — sono collegati al tracciamento di una cornice intorno al blocco. Per default, `border-style` è impostato a una dimensione nulla, e non viene tracciata nessuna cornice. Una serie di attributi "`padding-...`" sono usati per gestire la spaziatura tra il testo e il bordo della scatola, tracciata o no che sia.

L'organizzazione delle diverse zone di un blocco è mostrata nella figura 5. L'attributo `start-indent` (rispettivamente: `end-indent`) determina la distanza tra il primo (rispettivamente: ultimo) carattere e il bordo iniziale (rispettivamente: finale) della scatola. Bisogna sempre tener presente che tutti questi concetti dipendono dal *senso* della scrittura; non è quindi sempre pertinente parlare di "bordo superiore sinistro" e di "bordo inferiore destro", come se ci trovassimo sempre nel caso delle scritture latine. Allo stesso modo, le convenzioni usate per i valori dell'attributo `text-align` — `justify` per un testo giustificato, `center` per un testo centrato, `start` e `end` per un testo composto a bandiera sulla regione in questione — evitano di riferirsi troppo scopertamente alla scrittura latina, benché i valori `left` e `right` siano accettati come rispettivamente equivalenti a `start` e `end`.[12] Un altro attributo, `text-align-last`, riguarda la formattazione dell'ultima riga del blocco; il suo valore iniziale è `relative`, che equivale a `start` quando `text-align` è impostato a `justify`, o al valore di quest'ultimo attributo in tutti gli altri casi.[13] I valori `left` e `right` sono anch'essi accettati dall'attributo `text-align-last` con le stesse convenzioni valide per l'attributo `text-align`. Gli altri due attributi mostrati nella figura 5 sono relativi alle dimensioni: `text-indent` determina il rientro tradizionale a inizio capoverso, `last-line-text-indent` è usato per un rientro a fine paragrafo. Va sottolineato che questi attributi non sono sempre presi in considerazione: `text-indent` influenza la composizione solo se il capoverso è giustificato o imbandierato a sinistra, mentre `last-line-text-indent` è pertinente solo a un testo imbandierato a destra.

In caso di annidamento di elementi XSL-FO, gli attributi non ridefiniti sono ereditati; per esempio, il blocco che introduce l'autore del poema nella figura 3 usa lo stesso carattere del blocco che lo ingloba. L'elemento `fo:inline` permette la ridefinizione temporanea di attributi senza aprire un nuovo blocco, ovvero all'interno dello stesso capoverso. Per quanto riguarda gli attributi che regolano l'uso dei caratteri, essi sono mostrati nella tabella 1. Si noterà che la maggior parte dei nomi di questi attributi coincidono con i nomi di proprietà usati nei fogli di stile in CSS.[14] In LATEX i cambiamenti che riguardano il *look* di un carattere sono espressi da combinazioni di comandi come `\textbf` o `\textit`, mentre gli attributi di XSL-FO sono maggiormente "tipizzati" (concetti di *peso*, di *stile*, ...). Ciò può sembrare artificioso a un utente abituato a LATEX, ma mostra chiaramente tutte le combinazioni possibili. Quanto all'espressione del *corpo*, la soluzione, largamente adottata in molti documenti XSL-FO, consiste nell'uso di dimensioni espresse in punti, cosa che complica evidentemente possibili trasformazioni di tali documenti a situazioni in cui i caratteri devono essere ingranditi o ridotti. Mentre l'uso di corpi relativi impone alcune restrizioni (si veda la tabella 1), sono i corpi assoluti che si avvicinano maggiormente alla "filosofia" di LATEX: quando un corpo è espresso in punti, esso corrisponde in tutti i suoi sottoelementi al corpo `medium` e tutti gli altri si deducono da esso, così come `\small`, `\footnotesize`, `\large`, ecc., sono adattati, in LATEX, in rapporto al corpo `\normalsize` della classe di documento, eventualmente precisato da un'opzione del comando `\documentclass`.

Tornando all'elemento `fo:block`, la spaziatura tra due blocchi consecutivi è controllata dagli attributi `space-before` e `space-after`. Le **componenti** permettono, in particolare, la realizzazione in XSL-FO delle *lunghezze elastiche*[15] di TEX. Ciò che è specificato per la prima strofa della figura 3 è che il valore ideale per la spaziatura verticale che precede questo blocco è di 11 punti — la componente `optimum` dell'attributo `space-before` — dovendo tale spaziatura situarsi all'interno dell'intervallo tra 10 e 12 punti — le componenti `minimum` e `maximum`. Specificare:

```
<... space-before=11pt
     space-before.minimum=10pt»
```

definisce la componente `minimum`, mentre alle altre componenti è assegnato il valore `11pt`. XSL-FO fornisce altre due componenti per la gestione della spaziatura: conditionality indica se la specifica ha effetto (valore `retain`) o no (valore di default, `discard`) all'inizio o alla fine di un'*area di riferimento*[16] — l'inizio (o la fine, rispettivamente)

12. Per compatibilità con CSS (*Cascading style sheets*, si veda Meyer (2004) per una buona introduzione a questo linguaggio), che impiega anch'esso una direttiva `text-align`.

13. Ciò corrisponde alla formattazione "tradizionale" di un capoverso. Si noti, comunque, che è l'insieme del capoverso che deve essere allineato in maniera più armoniosa possibile. Inoltre una modifica dell'attributo `text-align-last` può avere effetti sulla composizione dell'intero capoverso.

14. Si noterà anche che, come per i CSS, alcune proprietà di XSL-FO possono essere espresse in forma abbreviata: per esempio, gli attributi `font-size` e `font-family`, relativi al tipo di carattere usato con l'elemento `fo:page-sequence` della figura 3, possono essere rimpiazzati da: "`font=12pt serif`". Questo punto è qui segnalato — si veda W3C (2006b, §§ 5.2 e B.3) — ma non se ne farà uso nel corso dell'articolo perché la maggior parte dei processori XSL-FO non lo supportano.

15. Secondo la terminologia di Mittelbach et al. (2004), "*rubber lengths*" nella versione originale.

16. "*Reference area*" (W3C, 2006b, § 4), secondo la terminologia di XSL-FO.

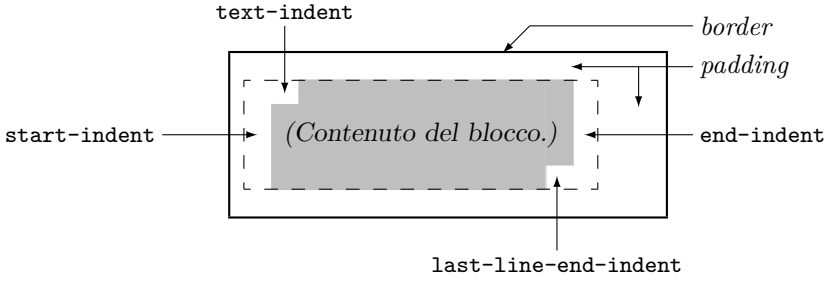


Figura 5: Visualizzazione delle parti di un blocco.

Tabella 1: Valori possibili per la maggior parte degli attributi legati ai caratteri.

Attributi	Default	Altri valori possibili
`font-family`	`serif`	`sans-serif`
`font-size`		Dimensioni assolute: `xx-small`, `x-small`, `medium`, `large`, `x-large`, `xx-large` Dimensioni relative: `smaller`, `larger` Dimensionsi esplicite, p. es.: `10pt`
`font-stretch`	`normal`	`wider`, `narrower`, `ultra-condensed`, `extra-condensed`, `condensed`, `semi-condensed`, `semi-expanded`, `expanded`, `extra-expanded`, `ultra-expanded`
`font-weight`	`normal`	`bold`, `bolder`, `lighter`
`font-style`	`normal`	`italic`, `reverse-normal`, `reverse-oblique`
`font-variant`	`normal`	`small-caps`

di una pagina per l'attributo `space-before` (o `space-after`) associato all'elemento `fo:block`, o l'inizio e la fine di una riga per gli stessi attributi, associati all'elemento `fo:inline`; `precedence` influenza il calcolo della spaziatura tra la fine di un'area di riferimento e l'inizio di quella seguente e può prendere un valore numerico qualsiasi (`0` per default), oppure il valore `force`, considerato più grande di ogni intero.[17]

Citiamo brevemente due attributi per i blocchi e per i testi "inline": `text-decoration`, usato per tracciare una linea sopra, sotto o attraverso il testo (W3C, 2006b, § 7.17.4), e `baseline-shift` per il posizionamento di indici ed esponenti. Per contro gli unici calcoli applicabili a un testo sono forniti dall'attributo `text-transform`, che può prendere i valori `uppercase` (conversione in lettere maiuscole[18]), `lowercase` (conversione in lettere minuscole), `capitalize` (la prima lettera di ciascuna parola è resa maiuscola, le altre sono minuscole), `none` (default). Per precisare cosa si intende qui con "calcolo", notiamo semplicemente che il frammento

```
\iflanguage{french}{Chanson italienne}{%
  Italian song}
```

— si veda Mittelbach et al. (2004, § 9.2.1) per ciò che riguarda il comando `\iflanguage` — non ha equivalenti in XSL-FO.

Altri attributi associati all'elemento `fo:block` riguardano la ripartizione del testo al momento della composizione dei blocchi: `keep-with-next` (o, rispettivamente, `keep-with-previous`), che indica che il contenuto di un capoverso deve essere posizionato sulla stessa pagina del precedente (rispettivamente: seguente); `keep-together`, che specifica che il contenuto di un capoverso deve essere posizionato sulla stessa pagina. Ciascuno di questi tre attributi possiede tre componenti — `within-line`, `within-column`, `within-page` — e i valori possibili sono degli interi che indicano l'importanza di tali vincoli, paragonabili all'argomento facoltativo dei comandi `\linebreak` e `\pagebreak` in LaTeX. Si possono usare due diverse notazioni: `auto` (default) e `always` per i livelli minimi e massimi. Forzare l'inizio di una nuova colonna o pagina è possibile per mezzo degli attributi `break-before` e `break-after`, i cui valori ammissibili sono `auto` (default), `column`, `page`, `even-page`, `odd-page`, dove gli ultimi due indicano un salto alla più vicina pagina pari o dispari che segue. Si veda Pawson (2002, pp. 70–72) per maggiori dettagli.

17. Le regole precise di questo calcolo sono fornite in W3C (2006b, §4.3.1).

18. Non va dimenticato che esistono delle scritture dove non esistono *maiuscole* e *minuscole* — per esempio le lingue dell'estremo oriente — e tale concetto, di conseguenza non ha senso, ecco perché l'attributo `text-transform` è usato sempre meno.

2.3 Altri elementi

La menzione di alcuni elementi qui di seguito serve a dare un'idea della potenza espressiva di XSL-FO. L'elemento radice per gli elenchi, assimilabile all'ambiente list di LaTeX, è `fo:list-block`. Per ciò che riguarda le tabelle, l'elemento radice è `fo:table` (Pawson, 2002, pp. 104–110) e la struttura è analoga a quella delle tabelle nel linguaggio HTML.[19] Le note a piè di pagina sono realizzate dall'elemento `fo:footnote`, ma la loro numerazione e l'eventuale linea orizzontale che separa la parte inferiore del testo da quella superiore delle note è a carico dell'utente (Hufflen, 2009b). Come in HTML, i riferimenti incrociati sono realizzati tramite collegamenti ipertestuali, verso altre parti dello stesso documento o verso documenti esterni, essendo `fo:basic-link` l'elemento usato. Come LaTeX, XSL-FO consente la manipolazione di *oggetti mobili*, — si veda W3C (2006b, § 6.12.2) riguardo l'elemento `fo:float` — e di *indici*, per mezzo di elementi e attributi (W3C, 2006b, § 7.24) analoghi all'ambiente `theindex`. Per terminare, si noti che non esiste un "modo matematico" in XSL-FO.

3 Gestione di testi in più lingue

XSL-FO fornisce degli attributi per controllare la sillabazione delle parole durante la formattazione di un blocco (W3C, 2006b, § 7.10). Innanzi tutto l'attributo `hyphenate`, per default impostato a `false`. In caso contrario, si specificheranno altri attributi,[20] tra i quali la specificazione di un carattere di cesura (`hyphenation-character`), di una lingua (`language`), di un paese (`country`). In pratica, l'attributo predefinito `xml:lang` — si vedano le sue due occorrenze nella figura 6 — può essere usato come abbreviazione per queste due ultime informazioni: si ricordi che si tratta di una lingua specificata da due lettere, eventualmente seguite da altre due che indicano il paese (Alvestrand, 1995). Ciò che manca, invece, alla presente versione di XSL-FO, è la possibilità di specificare dei punti di divisione *ad-hoc*,[21] come in LaTeX con i comandi `\-` e `\hypenhation`. Allo stesso modo non c'è un vero e proprio equivalente del comando `\discretionary`, che permette, in TeX, di specificare delle cesure più "esotiche", per esempio il fatto che in tedesco il gruppo "ck" si divideva come "k-k"[22] — `\discretionary{k-}{k}{ck}` — se non per mezzo di un attributo `script` che permette di specificare un programma di sillabazione (W3C, 2006b, § 7.10.3).

Come è stato accennato in precedenza nel paragrafo 2, il linguaggio XSL-FO non è limitato alle lingue che impiegano l'alfabeto latino. I vari elementi che definiscono i modelli di pagina — per esempio `fo:simple-page-master` — ammettono un attributo `writing-mode`, che fornisce delle *direzioni*: innanzi tutto la direzione delle righe, poi quella dei blocchi.[23] Per default questo attributo `writing-mode` vale `lr-tb`, che sta per "left-to-right, top-to-bottom" ovvero ciascuna riga di testo va letta da sinistra a destra, mentre più righe in successione vanno lette dall'alto verso il basso. Ecco tutti i valori che può assumere questo attributo (il processore Apache FOP[24] (apachefop 2011) riconosce alcuni di questi modi di scrittura,[25] come è stato illustrato in Hufflen (2009a)):

`rl-tb` da destra a sinistra e dall'alto in basso: è il modo impiegato dalle lingue semitiche;

`tb-rl` dall'alto in basso e da destra a sinistra: è il modo impiegato dalla scrittura giapponese tradizionale;

`tb-lr` dall'alto in basso e da sinistra a destra, usato in antichissimi testi in mongolico;

`lr-alternating-rl-tb` la prima riga si legge da sinistra a destra, la seconda da destra a sinistra e così via, mentre le righe si succedono dall'alto in basso; questo sistema, impiegato in alcune iscrizioni in greco arcaico, viene detto *bustrofedico*;[26]

`lr-alternating-rl-bt` analogo al precedente, ma le righe si succedono dal basso in alto; alcuni testi precolombiani nella lingua dell'Impero Inca usano questo sistema;

`lr-inverting-rl-bt` analogo al precedente, ma i caratteri che si leggono da destra a sinistra presentano una rotazione di 180°; questo "*bustrofedico inverso*" è stato osservato nella scrittura Rongo-rongo dell'Isola di Pasqua, non ancora decifrata;

`tb-lr-in-lr-pairs` il testo è frazionato in coppie di caratteri i cui elementi sono scritti da sinistra a destra, queste coppie sono poi disposte dall'alto in basso a formare una riga, e le righe successive scorrono da sinistra a destra; questo sistema è impiegato dalla lingua Maya.

Il *modus operandi* collegato agli altri valori possibili si deduce facilmente:

```
bt-lr bt-rl lr-bt rl-bt
  lr-inverting-rl-tb
```

19. *Hypertext markup language*. Una buona introduzione a questo linguaggio è Muscianoe Kennedy (2006).

20. Che per il momento molti processori XSL-FO non sono in grado di interpretare.

21. Questo punto sarà probabilmente corretto nella prossima versione di XSL-FO, per mezzo dell'attributo `hyphenation-exceptions (W3C, 2010, § 4.7.4)`.

22. "Divideva", perché dopo la riforma ortografica del 1996 si spezza la parola prima del gruppo "ck" (Duden, 2000, K 165).

23. "Inline-progression-direction" e "block-direction-progression", nella terminologia di XSL-FO.

24. *Formatting objects processor*.

25. Il lettore interessato alla storia dei sistemi di scrittura può consultare Février (1984).

26. Questa parola di origine greca (βουστροφηδόν) indica i movimenti di un bue che ara un campo.

e terminiamo notando che `lr`, `rl` e `tb` sono abbreviazioni per `lr-tb`, `rl-tb` e `tb-rl`. Quando un blocco viene processato, la direzione delle righe per una successione di caratteri può essere implicitamente determinata a partire dalla base di dati dei caratteri Unicode (The Unicode Consortium, 2006), ciò che permette di attivare l'algoritmo bidirezionale (Unicode Consortium, 2008). Si noti, però, che l'informazione fornita dall'attributo `writing-mode` ha precedenza sull'informazione implicita: in altri termini è possibile, per esempio, scrivere da destra a sinistra usando dei caratteri latini, se `writing-mode` assume un valore come `rl-tb`. Come è stato mostrato in Hufflen (2009a), la ridefinizione di questo attributo si effettua per mezzo degli elementi `fo:block-container` e `fo:inline-container`, i cui figli sono elementi `fo:block` e `fo:inline`.

4 Produzione di testi XSL-FO tramite XSLT

Ci prepariamo, adesso, a mostrare le idee di fondo che regolano l'uso di XSL-FO in un'ottica di separazione della forma dal contenuto. Una formulazione in XML del nostro poema italiano è riportata nella figura 6. Nelle figure da 7 a 9, invece, è raffigurata, *in extenso*, una trasformazione XSLT 2.0, il cui risultato è un testo in XSL-FO, di cui quello della figura 3 rappresenta una versione semplificata.[27] Non sono state aggiunte spiegazioni a queste figure — del resto l'argomento di questo articolo è XSL-FO e non XSLT; maggiori informazioni sulle tecniche impiegate possono essere trovate in Hufflen (2005, 2008), mentre il manuale didattico di riferimento per XSLT 2.0 è Kay (2004).

È stata usata la versione più recente di XSLT in quanto consente una maggiore espressività rispetto alla precedente (1.0, W3C (1999)) per questi tipi di compitii. XSLT 2.0 permette, in particolare, di specificare i tipi dei parametri e del risultato di un *template* e di validarli al momento dell'esecuzione. La specificazione avviene per mezzo dell'attributo `as`, impiegato con frequenza negli esempi in questione. L'uso di due *namespace*, definiti da prefissi `xmlns:xsl` e `xmlns:fo`,[28] distingue chiaramente le parti eseguite quando è in azione un programma XSLT (`<xsl:...>`) da quelle che costituiscono il risultato della trasformazione (`<fo:...>`). Si noti, a conclusione di questa sezione, che XSL-FO non fornisce degli strumenti per costruire automaticamente un sommario o un indice, ma questo compito non è molto difficile, dal momento che un testo XSL-FO è il risultato dell'applicazione di una trasformazione XSLT: alcuni esempi sono riportati in Pawson (2002, pp. 149–150).

5 Conclusioni

Eccoci giunti al termine della nostra iniziazione a XSL-FO. Questi primi passi ci hanno permesso già la realizzazione di effetti tipografici non banali. Per ciò che riguarda i processori disponibili, un elenco è presente sulla *home page* di XSL-FO (`http://www.w3.org/Style/XSL/`). Tra quelli elencati abbiamo sperimentato in particolar modo:

- PassiveTEX (Carlisle et al., 2000): è un adattamento di TEX per il trattamento di testi in XSL-FO, il risultato è un file DVI[29] (o PDF[30]) prodotto dal comando `xmltex` (o `pdfxmltex`);
- Apache FOP (apachefop 2011): scritto in Java, più completo, il risultato può essere un file PDF o Postscript (altri formati sono ugualmente possibili[31]);

usando Saxon (Kay, 2004, App. E) come processore XSLT 2.0.

Aggiungiamo a questo brevissimo elenco che la raccomandazione di W3C (W3C, 2006b) non impone un preciso formato di uscita, anche se la quasi totalità dei processori generano un file PDF, essendo divenuto questo formato uno standard di fatto. D'altra parte, per quanto è a nostra conoscenza, nessun processore attualmente esistente realizza per intero la raccomandazione del W3C, anche se in pratica riescono a trattare la maggior parte dei testi. Inoltre, alcuni — tra cui Apache FOP — permettono di sottoporre a una trasformazione XSLT il risultato della formattazione; si noti, però, che si tratta di trasformazioni che usano la versione 1.0 di XSLT.[32]

Riteniamo, tuttavia, che sia interessante seguire l'evoluzione di XSL-FO, perché questo linguaggio poggia su un concetto di base analogo a quello che hanno adottato i vari dialetti di TEX: il ricorso a una compilazione per ottenere il formato di uscita. Si tratta comunque di una versione alternativa — cioè differente per certi aspetti del funzionamento — di tali concetti; una versione, inoltre, più recente, anche se nel campo del multilinguismo e della gestione dei diversi sistemi di scrittura sono stati sviluppati strumenti come il pacchetto babel

27. In particolare, possiamo notare che il testo ottenuto mostra come specificare in XSL-FO una nota a piè di pagina (si vedano le figure 8 e 9).

28. Si ricordi che l'informazione che identifica un particolare *namespace* non è il prefisso in sé, in quanto identificatore, ma il valore ad esso associato, per esempio `http://www.w3.org/1999/XSL/Transform` in un programma XSLT. XInclude (si veda la nota 10 a pagina 42) introduce un altro *namespace* per realizzare inclusioni di file.

29. *DeVice Independent.*

30. *Portable Document Format*, il formato di Adobe.

31. In particolare, il formato RTF (*Rich Text Format*) usato per l'interscambio con Microsoft Word. Apache FOP è stato usato in questo articolo per produrre la figura 4 a partire dal sorgente riportato nella figura 3.

32. In realtà i due linguaggi appartenevano in origine allo stesso progetto. In seguito, un chiarimento degli obbiettivi ha portato a una separazione netta tra XSLT e XSL-FO.

```
<!DOCTYPE song-0 SYSTEM "song-0.dtd">
<song-0 xml:lang="it">
  <preamble>
    <title>Ballo delle Ingrate</title>
    <author>
      <firstname>Ottavio</firstname>
      <lastname>Rinuccini</lastname>
      <birth-year>1562</birth-year>
      <death-year>1621</death-year>
    </author>
    <note xml:lang="fr">Musique de Claudio Monteverdi (1567-1643).</note>
  </preamble>
  <body>
  <stanza role="Amore">
      <verse>De l'implacabil Dio</verse>
      <verse>Eccone giunt'al Regno,</verse>
      <verse>Seconda, O bella Madre, il pregar mio.</verse>
    </stanza>
    <stanza role="Venere">
      <verse>Non tacerà mia voce</verse>
      <verse>Dolci lusinghe e prieghi</verse>
      <verse>Finche l'alma feroce</verse>
      <verse>Del Re severo al tuo voler non pieghi.</verse>
    </stanza>
    <stanza role="Amore">
      <verse>Ferma, Madre, il bel piè, non por le piante</verse>
      <verse>Nel tenebroso impero,</verse>
      <verse>Che l'aer tutto nero</verse>
      <verse>Non macchiass'il candor del bel sembiante:</verse>
      <verse>Io sol n'andrò nella magion oscura,</verse>
      <verse>E pregand'il gran Re trarotti avante.</verse>
    </stanza>
    <stanza role="Venere">
      <verse>Va pur come t'agrada. Io qui t'aspetto,</verse>
      <verse is-followed-by="Sinfonia">Discreto pargoletto.</verse>
    </stanza>
  </body>
</song-0>
```

Figura 6: L'esempio di poema, *in extenso*.

```
<!DOCTYPE stylesheet [<!ENTITY layout SYSTEM "layout.fo">
                     <!ENTITY en-dash "&#x2013;">]>
<xsl:stylesheet version="2.0" id="song-0-2-fo"
                xmlns:fo="http://www.w3.org/1999/XSL/Format"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                exclude-result-prefixes="xsd">
  <xsl:output method="xml" indent="yes"/>
  <xsl:strip-space elements="*"/>     <!-- Elimina i nodi contenenti solo spazi. -->
  <xsl:template match="song-0" as="element(fo:root)">
    <fo:root>
      <xsl:apply-templates select="@xml:lang"/>
      &layout;
      <fo:page-sequence master-reference="simple-page" font-family="serif"
                        font-size="medium" text-align="start">
        <xsl:variable name="preamble" select="preamble"/>
        <xsl:apply-templates select="$preamble/title" mode="in-footer"/>
        <fo:flow flow-name="xsl-region-body">
          <xsl:apply-templates select="$preamble,body"/>
        </fo:flow>
      </fo:page-sequence>
    </fo:root>
  </xsl:template>
  <xsl:template match="preamble" as="element(fo:block)+">
    <fo:block font-family="sans-serif" font-size="x-large" padding-top="3pt"
              text-align="center" color="white" background-color="blue"
              space-after="15pt" line-height="24pt">
      <xsl:apply-templates select="title,note"/>
      <xsl:apply-templates select="collection">
        <xsl:with-param name="left" select="'('" as="xsd:string" tunnel="yes"/>
        <xsl:with-param name="right" select="')'" as="xsd:string" tunnel="yes"/>
      </xsl:apply-templates>
    </fo:block>
    <xsl:variable name="both" select="'4mm'" as="xsd:string"/>
    <fo:block font-family="sans-serif" font-size="large" space-after="18pt"
              border-style="solid" border-width="0.5mm" border-color="blue"
              padding="{$both}" start-indent="40mm" end-indent="{$both}">
      <xsl:apply-templates select="author"/>
    </fo:block>
  </xsl:template>
  <xsl:template match="title | collection" as="xsd:string">
    <xsl:call-template name="put-text"/>
  </xsl:template>
```

FIGURA 7: Trasformazione XSLT che genera un testo XSL-FO (inizio).

```
<xsl:template match="note" as="element(fo:footnote)">
  <fo:footnote>
    <xsl:call-template name="put-footnotemark">
      <xsl:with-param name="size" select="'large'" as="xsd:string"/>
    </xsl:call-template>
    <fo:footnote-body>
      <fo:block text-align-last="justify" color="black">
        <fo:leader leader-pattern="rule"/>
      </fo:block>
      <fo:block font-family="serif" font-size="small" text-align="justify"
                color="black">
        <xsl:apply-templates select="@xml:lang"/>
        <xsl:call-template name="put-footnotemark"/>
        <xsl:apply-templates/>
      </fo:block>
    </fo:footnote-body>
  </fo:footnote>
</xsl:template>

<xsl:template match="author" as="element(fo:block)">
  <fo:block text-align="end">
    <xsl:value-of select="data(firstname),data(lastname)"/>
    <fo:inline font-style="italic">
      <xsl:value-of
        select="' (',birth-year,
                if (data(//song-0/@xml:lang) eq 'fr') then '-' else '&en-dash;',
                death-year,')'"
        separator=""/>
    </fo:inline>
  </fo:block>
</xsl:template>

<xsl:template match="body" as="element(fo:block)+">
  <xsl:apply-templates/>
</xsl:template>

<xsl:template match="stanza" as="element(fo:block)">
  <fo:block space-before.minimum="10pt" space-before.optimum="11pt"
            space-before.maximum="12pt">
    <xsl:apply-templates select="@role,verse"/>
  </fo:block>
</xsl:template>

<xsl:template match="verse" as="element(fo:block)+">
  <fo:block><xsl:apply-templates/></fo:block>
  <xsl:apply-templates select="@is-followed-by"/>
</xsl:template>
```

Figura 8: Trasformazione xslt che genera un testo xsl-fo (seguito).

```
  <xsl:template match="@role" as="element(fo:block)+">
    <fo:block font-weight="bold" keep-with-next.within-page="always">
      <xsl:call-template name="put-text">
        <xsl:with-param name="left" select="'['" as="xsd:string" tunnel="yes"/>
        <xsl:with-param name="right" select="']'" as="xsd:string" tunnel="yes"/>
      </xsl:call-template>
    </fo:block>
  </xsl:template>
  <xsl:template match="@is-followed-by" as="element(fo:block)">
    <fo:block font-style="italic" text-align="center"
              keep-with-previous.within-page="always">
      <xsl:call-template name="put-text">
        <xsl:with-param name="left" select="'('" as="xsd:string" tunnel="yes"/>
        <xsl:with-param name="right" select="')'" as="xsd:string" tunnel="yes"/>
      </xsl:call-template>
    </fo:block>
  </xsl:template>
  <xsl:template match="@xml:lang" as="attribute(xml:lang)"><xsl:copy/></xsl:template>
  <xsl:template match="title" mode="in-footer" as="element(fo:static-content)">
    <fo:static-content flow-name="xsl-region-after">
      <fo:block text-align="center" line-height="14pt" color="green"
                font-size="small">
        <xsl:value-of select="data(.),'('"/>
        <fo:page-number/>
        <xsl:text>)</xsl:text>
      </fo:block>
    </fo:static-content>
  </xsl:template>
  <xsl:template name="put-text" as="xsd:string">
    <xsl:param name="left" select="''" as="xsd:string" tunnel="yes"/>
    <xsl:param name="right" select="''" as="xsd:string" tunnel="yes"/>
    <xsl:value-of select="$left,data(.),$right" separator=""/>
  </xsl:template>
  <xsl:template name="put-footnotemark" as="element(fo:inline)">
    <xsl:param name="size" select="'xx-small'" as="xsd:string"/>
    <fo:inline font-size="{$size}" vertical-align="super">*</fo:inline>
  </xsl:template>
</xsl:stylesheet>
```

FIGURA 9: Trasformazione XSLT che genera un testo XSL-FO (fine).

di LaTeX 2ε (Mittelbach et al., 2004, cap. 9) o il nuovo motore di composizione XeTeX (Kew, 2007). Riteniamo che una ripresa del progetto Passive TeX potrebbe sfociare in una interessante coordinazione degli sforzi tra le comunità di TeX e XSL-FO.

Ringraziamenti

Ringrazio in primo luogo gli spettatori della presentazione che ho tenuto in passato nell'ambito dei corsi dell'Université de Franche-Comté o in occasione delle giornate organizzate dai gruppi di utilizzatori di TeX. Il ricordo delle domande che mi hanno posto in tali occasioni mi ha permesso di migliorare sensibilmente alcuni punti di questo articolo.

Riferimenti bibliografici

Harald Tveit Alvestrand. (1995) *Request for Comments: 3066. Tags for the Identification of Languages*. http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc3066.html. UNINETT, Network Working Group.

Bernd Amman e Philippe Rigaux. (2002) *Comprendre* XSLT. Éditions O'Reilly France.

Apache FOP. (2011) http://xmlgraphics.apache.org/fop/.

David Carlisle, Michel Goossens, e Sebastian Rahtz. (2000) De XML à PDF avec xmltex, XSLT et

PassiveTEX. *Cahiers GUTenberg*, 35–36:79–114. In *Actes du congrès GUTenberg 2000*, Toulouse.

Duden. (2000) *Die deutsche Rechtschreibung. Band 1*. Bibliographisches Institut, Mannheim, 22 edition.

James G. Février. (1984) *Histoire de l'écriture*. Payot, 2 edition.

Jean-Michel Hufflen. (2005) Introduction to XSLT. *Biuletyn* GUST, 22:64. in *BachoTEX 2005 conference*.

Jean-Michel Hufflen. (2007) Introducing LATEX users to XSL-FO. TUG*boat*, 29(1):118–124. EuroBachoTEX 2007 proceedings.

Jean-Michel Hufflen. (2008) XSLT 2.0 vs XSLT 1.0. In *Proc. BachoTEX 2008 Conference*, pages 67–77.

Jean-Michel Hufflen. (2009a) Multidirectional typesetting in XSL-FO. In Tomasz Przechlewski, Karl Berry, e Jerzy B. Ludwichowski, editors, *TEX: at a Turning Point, or at the Crossroads? Proc. BachoTEX 2009 Conference*, pages 37–40.

Jean-Michel Hufflen. (2009b) Processing "computed" texts. *ArsTEXnica*, 8:102–109. In GUIT 2009 meeting.

Michael H. Kay. (2004) XSLT *2.0 Programmer's Reference*. Wiley Publishing, Inc., 3 edition.

Jonathan Kew. (2007) XeTEX in TEX live and beyond. TUG*boat*, 29(1):146–150. EuroBachoTEX 2007 proceedings.

Eric A. Meyer. (2004) CSS*: the Definitive Guide*. O'Reilly & Associates, Inc., 2 edition.

Frank Mittelbach e Michel Goossens, con Johannes Braams, David Carlisle, Chris A. Rowley, Christine Detig, e Joachim Schrod. (2004) *The LATEX Companion*. Addison-Wesley Publishing Company, Reading, Massachusetts, 2 edition.

Manuel Montero Pineda e Manfred Krüger. (2004) XSL-FO *in der Praxis.* XML*-Verarbeitung für* PDF *und Druck*. dpunkt.verlag.

Chuck Musciano e Bill Kennedy. (2006) HTML*: The Definitive Guide*. O'Reilly & Associates, Inc., 6 edition.

Dave Pawson. (2002) XSL-FO. O'Reilly & Associates, Inc.

Erik T. Ray. (2001) *Learning* XML. O'Reilly & Associates, Inc.

The Unicode Consortium. (2006) *The Unicode Standard Version 5.0*. Addison-Wesley.

Unicode Bidirectional Algorithm. (2008) The Unicode Consortium, `http://unicode.org/reports/tr9/`. Unicode Standard Annex #9.

W3C. (1999) XSL *Transformations (*XSLT*). Version 1.0*. `http://www.w3.org/TR/1999/REC-xslt-19991116`. W3C Recommendation. Edited by James Clark.

W3C. (2001) *Extensible Stylesheet Language (*XSL*). Version 1.0*. `http://www.w3.org/TR/2001/REC-xsl-20011015/`. W3C Recommendation. Edited by James Clark.

W3C. (2006a) XML *Inclusions (XInclude) Version 1.0*. `http://www.w3.org/TR/2006/REC-xinclude-20061115/`, 2 edition. W3C Recommendation. Edited by Jonathan Marsh, David Orchard, and Daniel Veillard.

W3C. (2006b) *Extensible Stylesheet Language (*XSL*). Version 1.1*. `http://www.w3.org/TR/2006/REC-xsl11-20061205/`. W3C Recommendation. Edited by Anders Berglund.

W3C. (2008) *Extensible Stylesheet Language (*XSL*) Requirements Version 2.0*. `http://www.w3.org/TR/2008/WD-xslfo20-req-20080326/`. W3C Working Draft. Edited by Klaas Bals.

W3C. (2010) *Design Notes for Extensible Stylesheet Language (*XSL*) 2.0*. `http://www.w3.org/TR/2009/WD-xslfo20-20101216/`. W3C Working Draft. Edited by Liam R E Quin.

▷ Jean-Michel Hufflen
Université de Franche-Comté
16, route de Gray
25030 Besançon Cedex – France
`jmhufflen at lifc dot univ-fcomte dot fr`

La virgola intelligente

Claudio Beccari

Sommario

La parte decimale di un numero deve essere separata dalla parte intera mediante un separatore decimale. Le norme ISO specificano un segno differente secondo la lingua usata; i codici interni di LaTeX per i caratteri matematici non agevolano una facile gestione del separatore decimale. Qui si descrivono alcune maniere per risolvere questo problema.

Abstract

The decimal fractional part of a number must be separated from the integer part by means of a decimal separator. The ISO regulations specify a different sign for different languages; the internal LaTeX mathematical character codes do not help treating this sign in a simple way. Here we describe a few ways to handle this problem.

1 Introduzione

La storia del separatore decimale fra la parte intera e quella fratta di un numero decimale è molto interessante, ma nel mondo d'oggi, dove le norme ISO standardizzano ogni cosa, le tradizioni storiche si riflettono solo nelle varianti di questo segno secondo la lingua. Infatti le norme ISO specificano l'uso del punto decimale nei testi scritti in inglese, e la virgola decimale negli altri casi. In Gran Bretagna è ancora comune l'uso del punto centrato a mezza riga, ma questa simpatica tradizione non è conforme alle norme ISO.

Perciò questa gestione differente del separatore decimale in accordo con la lingua, secondo me, dovrebbe essere affidata alla selezione della lingua mediante i pacchetti babel o polyglossia. Fino ad oggi i miei tentativi di inserire il giusto trattamento della virgola con l'opzione `italian` per babel non ha sortito effetti[1]. È ragionevole: perché solo per l'italiano e non anche per tutte le altre lingue?

L'altro aspetto della medaglia è costituito da come TeX vede le cose; ogni simbolo usato in matematica ha un *mathcode* che ne specifica la funzione. Quando il punto decimale viene usato come separatore decimale, il suo mathcode lo classifica come un simbolo 'normale', e questo significa che TeX non inserisce nessuno spazio né prima né dopo questo simbolo; quando il punto è usato come segno di punteggiatura, questo succede solo alla fine di una espressione matematica e quindi non separa nulla da qualcos'altro, per cui la mancanza di spazio dopo il punto è irrilevante: ci pensa la posizione dell'espressione nel suo contesto, sia essa fuori testo o in linea, a lasciare lo spazio necessario fuori da un ambiente matematico.

Al contrario, quando si usa la virgola come separatore decimale questa dovrebbe essere trattata diversamente da quando viene usata come segno di punteggiatura, per esempio come separatore degli elementi di una lista. Si confronti infatti la lista delle variabili indipendenti se si scrive $f(x,y)$ rispetto a quando si scrive $f(x, y)$; si confronti anche il separatore decimale nelle espressioni $e = 2{,}718$ e $e = 2,718$. Queste differenze avvengono perché la funzione di un segno di punteggiatura implica che ci sia un piccolo spazio alla sua destra. Naturalmente la spaziatura può essere aggiustata a mano mediante l'uso dei comandi di spaziatura matematica, ma questo è contrario al buon senso: LaTeX dovrebbe essere in grado di gestire il problema lasciano all'autore il compito di concentrarsi su quello che scrive, invece che sul suo aspetto.

2 Soluzioni pronte per l'uso

Esistono già diversi pacchetti che risolvono questo problema; da un lato ci sono quelli che gestiscono il segno corretto e la sostituzione del segno sbagliato. tra questi si possono citare numprint Harders (2008) e siunitx Wright (2010)

Il primo pacchetto non solo lavora sia in modo testo sia in modo matematico, ma provvede anche a cambiare il segno 'sbagliato' nel segno 'giusto', oltre a trasformare la notazione tipica dei calcolatori in una espressione matematica: per esempio può cambiare `12345.6e789` in $12345{,}6\cdot10^{789}$ mantenendo i font corretti sia nel testo che nelle espressioni matematiche.

Il secondo pacchetto è uno strumento completo per gestire le unità di misura (in particolare quelle del SI), i loro valori numerici sia in matematica sia nel testo, ma anche nella costruzione delle tabelle così da ottenere un documento composto alla perfezione con il segno decimale giusto, i font appropriati, gli incolonnamenti corretti nelle tabelle, le unità di misura scritte alla perfezione e distanziate mediante gli appositi spazi non separabili. Servizio completo, insomma!

Ci sono anche pacchetti più semplici che si occupano solo dei codici matematici per comporre

1. Solo la mia configurazione personale del file `italian.ld` contiene la virgola intelligente; troppo spesso mi dimentico che gli altri utenti italiani non ne sono dotati e purtroppo questo genera da parte mia dei malintesi ai quali dovrei stare più attento.

correttamente in modalità matematica. Si possono citare per esempio icomma Schmidt (2002) e ncccomma Rozhenko (2005).

Il primo pacchetto definisce una virgola costituita da un carattere attivo solo in matematica; esso controlla se è seguito da uno spazio e in questo caso inserisce una virgola di punteggiatura, altrimenti inserisce una virgola decimale.

Il secondo pacchetto definisce anch'esso un virgola costituita da un carattere attivo solo in matematica, ma se è seguito da una cifra inserisce una virgola decimale, altrimenti inserisce un virgola di punteggiatura. Il secondo pacchetto è marginalmente più comodo da usare, anche se più lento (qualche microsecondo!) perché deve eseguire fino a dieci confronti per riconoscere le cifre, invece di un solo confronto per riconoscere uno spazio; ma in questo modo il 'tastierista' non deve ricordarsi di nessuna particolare scrittura nel file sorgente perché pensa a tutto LaTeX, tranne in un solo caso: quando, cioè, la virgola serve per separare gli elementi di una lista di numeri come in $\forall i \in 0, 1, 2, \dots, n$, che deve essere scritta nella forma: `$\forall␣i␣\in␣0,␣1,␣2,\dots,n$`.

3 Una proposta per una virgola intelligente

Io propongo un'altra soluzione simile ma più semplice rispetto a quella di Alexander Rozhenko: la sua soluzione usa una sequenza di comandi `\expandafter`; io ho la presunzione di credere che la mia soluzione sia più facile da capire:

```
\makeatletter % non necessario quando
              % questo codice è in un
              % file .sty

% 1.a parte
\DeclareMathSymbol
  {\punctcomma}{\mathpunct}{letters}{"3B}
\DeclareMathSymbol
  {\decimalcomma}{\mathord}{letters}{"3B}

% 2.a parte
\AtBeginDocument{\mathcode`\,="8000}
{\catcode`\,=\active
  \gdef,{\futurelet\let@token\m@thcomma}}

% 3.a parte
\def\m@thcomma{%
  \let\@tempB\punctcomma
  \@tfor\@tempA:=0123456789\do{%
  \expandafter\ifx\@tempA\let@token
  \let\@tempB\decimalcomma
  \@break@tfor\fi}\@tempB}
```

Come si vede dal listato, il codice può essere diviso in tre parti:

1. Vengono dapprima definiti due simboli matematici: `\punctcomma` è la definizione della virgola da usare come segno di punteggiatura, mentre `\decimalcomma` rappresenta la virgola decimale. Il secondo parametro del comando dichiarativo specifica molto chiaramente quale sia il ruolo di questi due simboli in matematica.

2. Il carattere della virgola è reso attivo in matematica; il codice esadecimale `"8000` è lo speciale mathcode che TeX mette a disposizione per definire attivi i caratteri *solo* in modo matematico. Ma questa dichiarazione di 'attività' è dilazionata fino al momento di iniziare la composizione del documento in modo da non disturbare altri pacchetti nel caso facessero uso di caratteri attivi, virgola inclusa. L'azione della virgola attiva viene definita dentro un gruppo ma la definizione è globale così da superare la barriera del gruppo.

3. Alla fine si definisce la macro che fa davvero il lavoro che interessa. Questa macro prima assegna il comando per la virgola di punteggiatura ad un comando 'sosia'; poi scandisce una lista di token per mezzo di una macro ricorsiva `\@tfor` definita internamente al nucleo di LaTeX. Questa macro ricorsiva *definisce* un'altra macro temporanea mediante i successivi token della lista che è contenuta fra i segni di delimitazione `:=` e `\do`, costituita nel nostro caso dalla dieci cifre decimali, e poi esegue iterativamente quello che appare essere l'argomento della sequenza di controllo `\do` (che invece è solo un delimitatore[2]). Questa macro temporanea dapprima viene sviluppata (o espansa, come si è soliti dire) al fine di estrarre il token che ne costituisce la definizione e poi quest'ultimo viene confrontato con il token temporaneo `\let@token` che la macro principale aveva reso 'sosia' del primo token che seguiva la virgola nel file sorgente. Il primo 'sosia' della virgola di punteggiatura viene eventualmente reso 'sosia' della virgola decimale se questo confronto è positivo, cioè in definitiva se la virgola nel testo sorgente era direttamente seguita da un cifra. L'ultimo comando criptico `\@break@tfor` serve solo per uscire dalle iterazioni al primo confronto positivo. Come ultima cosa viene eseguito il 'sosia' definitivo della virgola che svolge la funzione giusta in base al contesto matematico in cui si trova.

Non c'è nulla di nuovo, tranne la semplicità e la versatilità. In effetti, aggiungendo qualche semplice variazione al tema svolto in quel breve listato, sarebbe possibile anche cambiare un punto decimale in una virgola decimale o in ogni altro

2. Per le definizioni di macro ad argomenti delimitati si può vedere l'articolo di Beccari (2008) oltre che il TeXbook, Knuth (1990).

segno in cui si voglia trasformare la virgola o il punto. Si potrebbe anche rendere un poco più veloce l'esecuzione dei comandi che costituiscono la definizione della virgola attiva aggiungendo un test che verifica preventivamente se nel file sorgente la virgola sia seguita da uno spazio, perché in tal caso sarebbe possibile evitare la macro ricorsiva:

```
% 3.a parte
\def\m@thcomma{%
  \let\@tempB\punctcomma
  \ifx\let@token\space\else
  \@tfor\@tempA:=0123456789\do{%
  \expandafter\ifx\@tempA\let@token
  \let\@tempB\decimalcomma
  \@break@tfor\fi}\fi\@tempB}
```

Peccato che nelle viscere del programma di composizione le cifre non abbiano un codice mathcode distinto da quello degli altri simboli che a tutt'oggi sono classificati come normali; la causa è dovuta al fatto che, prima dell'avvento di *xetex*, il programma di composizione messo a punto da Knuth disponesse solo di tre bit (corrispondenti a sette differenti combinazioni) per caratterizzare le funzioni di un simbolo matematico; l'ottavo bit serviva solamente per distinguere i caratteri attivi matematici[3].

Poiché il test `\iflanguage` è presente in entrambi i pacchetti babel e polyglossia, si potrebbe subordinare la dichiarazione del carattere matematico attivo alla lingua in uso in modo da adattare il separatore decimale al contesto testuale. Meglio ancora sarebbe quello di aggiungere l'assegnazione del codice matematico `"8000` ai comandi `\extras`*language* e ripristinare con `\noextras`*language* il mathcode di default quando si cambia lingua. Queste però sono cose che è bene siano fatte solo dai gruppi di lavoro che si occupano di babel e polyglossia.

4 Conclusioni

Le macro proposte nel paragrafo precedente lavorano bene sia con *pdflatex* sia con *xelatex*, in quest'ultimo caso anche quando si usa la dichiarazione `unicode-math`.

3. Senza scendere nei dettagli, il codice matematico è formato da quattro *nibble* (sequenze di quattro bit, ovvero mezzo byte); due di questi nibble sono destinati ad identificare la posizione del simbolo nella polizza dei caratteri matematici; uno serve a scegliere la polizza o 'alfabeto' matematico da usare; e l'ultimo serve a definire la funzione del simbolo: delimitatore, operatore di relazione, simbolo di punteggiatura, eccetera. Questo è anche uno dei motivi per i quali si possono usare solo 16 diverse polizze di caratteri e simboli matematici, sei delle quali sono già impegnate dal nucleo di LaTeX e dal pacchetto amsmath. Le restanti 10 polizze dovrebbero essere più che sufficienti per coprire qualunque esigenza, ma, sebbene accada molto raramente, a me è già capitato di tentare di eccedere quel limite! Questi sono gli inconvenienti residui che rimangono in TeX, dettati dalle necessità di economia di spazio connesse alla modestia degli elaboratori disponibili negli anni '70, quando TeX è stato concepito.

Tutto il codice ammonta a otto righe; queste possono venire copiate in un file di macro personali o nel preambolo di qualunque documento scritto che usa lingue diverse dalle varietà inglesi.

Io uso queste poche righe in tutti i miei documenti scritti in italiano; devo fare solamente attenzione ad inserire nel file sorgente degli spazi dopo le virgole quando queste separano gli elementi di una lista numerica. Altrimenti la virgola è sufficientemente intelligente da produrre la funzione desiderata come si suppone che faccia.

Mi piacerebbe che una soluzione così semplice, o una soluzione migliore, venisse alla fine presa in considerazione dai gruppi di lavoro che si occupano di babel e polyglossia, cosicché nel futuro ci possiamo scordare della virgola intelligente in quanto essa sarà già incorporata nel meccanismo di cambiamento di lingua.

Riferimenti bibliografici

Beccari, C. (2008). «Macroistruzioni con argomenti delimitati». *ArsTEXnica*, (5). URL `http://www.guit.sssup.it/arstexnica/download_ars/articoli_ars_05/Macroistruzioni%20con%20argomenti%20delimitati.pdf`.

Harders, H. (2008). «The numprint package». URL `$TEXMFDIST/doc/latex/numprint/numprint.pdf`.

Knuth, D. E. (1990). *The TEXbook*, volume A di *Computers and Typesetting*. Addison Wesley, 18ª edizione.

Rozhenko, A. I. (2005). «The ncccomma package». URL `$TEXMFDIST/doc/latex/ncctools/ncccomma.pdf`.

Schmidt, W. (2002). «The package icomma». URL `$TEXMFDIST/doc/latex/was/readme.1st`.

Wright, J. (2010). «siunitx — A comprehensive (SI) units package». URL `$TEXMFDIST/doc/latex/siunitx/siunitx.pdf`.

▷ Claudio Beccari
Villarbasse (TO)
`claudio dot beccari at gmail dot com`

The unknown *picture* environment

Claudio Beccari

Abstract

The old *picture* environment, introduced by Leslie Lamport into the LaTeX kernel almost 20 years ago, appears to be neglected in favor of more modern and powerful LaTeX packages that eliminate all drawbacks of the original environment. Nevertheless it is still being used behind the scenes by a number of other packages. Lamport announced an extension in 1994 that should have removed all the limitations of the original environment; in 2003 appeared the first version of this extension, in 2004 it was released the first stable version; in 2009 it was actually expanded to new functionalities. Nowadays the *picture* environment can perform as most simple drawing programs do, but it has special features that make it invaluable.

Sommario

L'ambiente *picture*, introdotto da Leslie Lamport nel nucleo di LaTeX quasi 20 anni fa, sembra essere trascurato a favore di pacchetti LaTeX più moderni e più potenti che eliminano tutti gli inconvenienti della versione originale. Tuttavia esso è ancora usato oggigiorno dietro le quinte da numerosi altri pacchetti. Nel 1994 Lamport annunciò un pacchetto di estensione che avrebbe dovuto rimuovere ogni inconveniente; solo nel 2003 apparve una prima versione; nel 2004 questa estensione divenne stabile; e nel 2009 essa è stata arricchita di nuove funzionalità. Oggi l'ambiente *picture* si comporta come qualsiasi altro semplice programma di disegno, ma presenta alcune particolarità uniche che lo rendono insostituibile.

1 Introduction

The plain TeX, as described in the TeXbook, (Knuth, 1990), contained a simple way to draw simple graphics with *tex*; when LaTeX was first published in 1984, it contained an environment suitable for relatively complex graphics; Lamport's handbook, (Lamport, 1994) described its workings and commands. But all this seems to be fallen in complete oblivion.

Most users of LaTeX related forums keep asking questions such as "How can I produce such and such symbol"; I keep answering "Use the *picture* environment". Apparently nobody follows my suggestions, which of course they are free not to; but they would save time if they spent no more than 15 minutes in reading the environment description on Lamport's handbook; the second edition of this handbook announces the extensions and discusses the eliminated drawbacks; these extensions were eventually realized only in 2003 by Gäßlein and Niepraschk, (Gässlein *et al.*, 2009). In 2004 the same authors released a stable version. In 2009, with the contribution of the third author Tkadlec, they released an enhanced versions that added quite a few new commands that substantially extend the *picture* environment functionality. But, except the latest additions of the year 2009, everything else was already documented by Lamport in his handbook second edition.

Not longer than a few days ago a forum participant asked how he could draw a square wave and a saw tooth wave of suitable size for setting them in line with his text: ⎍ and ⩘.

The whole thing required to produce two simple commands by resorting to the recent pict2e extension package new commands:

```
\newcommand*\sqwave[1][0.125ex]{{%
\unitlength=#1\relax
\picture(20,10)
\polyline(0,0)(6.5,0)(6.5,15)(13,15)%
         (13,0)(19.5,0)
\endpicture}}

\newcommand*\sawtooth[1][0.125ex]{{%
\unitlength=#1\relax
\picture(20,10)
\polyline(0,0)(6.5,0)(6.5,15)(13,0)%
         (19.5,0)
\endpicture}}
```

Most of the time suggestions are given by and to the forum participants to use external drawing programs; or to use the sophisticated PSTricks or TikZ packages. The former solution is generally to be avoided, because if some lettering is needed, it requires more work to use the same fonts as those used in the document. The latter two packages are really capable of doing marvellous and complicated drawings, but require a long documentation time and a steep learning curve.

But there is a unique feature to the *picture* environment: it can produce drawings of zero width and/or height. This special feature makes them precious for packages such as eso-pic, (Niepraschk, 2010), crop, (Franz, 2003), layout, (McPherson, 2000), layouts, (Wilson e Robertson, 2009), and others. These packages draw things on the page that do not require any external package, and therefore don't have any dependency; these drawings occupy no space, although they have a specific

position on the page; their contents reach whatever point on the page, as background images or marks that do not interfere with the positioning of other page elements.

In particular eso-pic (and similar packages for setting watermarks) and crop exploit this functionality exactly for setting a background picture or the crop marks in the correct positions without interfering with the other elements of the page.

At the same time within the *picture* environment it is possible to use the cubic Bézier curves and to draw polygons, arcs and sectors, oval frames whose corner curvature can be freely specified, white or filled circles of any dimension. The `\polyline` macro used in the above simple examples makes it very easy to draw polylines with any number of corners and any segment slope, to the point that if the nodes are sufficiently close, it is possible to draw "smooth" curves whose point may be calculated with any number-crunching personal or main frame computer.

2 In detail

Everybody can agree that the original *picture* environment, created by Lamport with its very first version of LaTeX, was pretty rudimentary but there was practically nothing else to use in its place. The straight lines could have slopes that were ratios of relatively prime integer numbers not exceeding 6 in absolute value. For vectors the limitation was even stricter; the slopes could not be specified with integer numbers larger than 4 in absolute value. Why there were such strange limitations? because straight lines were made up through the juxtaposition of small segments 10pt (≈ 3mm) long taken from a special font; this same special font contained also the vector arrow tips that occupied a large part of the available positions; and, remember, at that time the typesetting engine could deal only with 128-glyph fonts, so that the available short segments and arrow tips, plus a selection of closed and filled circles and/or quarter circles required strict limitations on the drawing performances.

Patient programmers made up extension packages such as curves, (Maclaine-cross, 2008), that could overcome such limitations by drawing lines of any slope and circles of any diameter by juxtaposing an "infinity" of small dots. For plain TeX there existed another package, PicTeX, (Wichura, 1987), that with a suitable interface could work also with LaTeX. That package performed very well on large mainframes with large memory capabilities, but worked very poorly on the desktops of that age, the eighties, when a 20MiB hard disk was a luxury and a 640KiB RAM was almost the maximum available. These packages mostly saturated the RAM and drawing was virtually impossible on personal computers.

Some progress was made when the PostScript format became available; some drawing packages (again curves) exploited the `\special` command in order to write on the output file raw PostScript drawing commands, so that the actual action of drawing was demanded on the screen or printer driver. Nevertheless powerful extensions in this directions, such as PSTricks appeared much later. Drawing with external programs and importing the resulting artwork was therefore a necessity, but not an easy task.

Things evolved in the right direction when personal computers, having become the universal complement of any person needing to write anything, started having a more user friendly interface, larger RAMs, larger hard disks, and better programs, the TeX system included. The nineties, besides the important passage from TeX2.x to TeX3.0, gave us LaTeX 2ε, PostScript fonts, and the drawing instrument METAPOST. This program used more or less the same philosophy that led Knuth to develop METAFONT, in order to draw the TeX system default fonts; METAPOST produced a simplified output PostScript code that was understood also by the new born typesetting program *pdftex*. METAPOST was, and still is, fully compatible with the rest of the TeX system typesetting engines, so that all the TeX and LaTeX features could be used for putting any lettering on the METAPOST output files.

Meanwhile LaTeX went on with its small drawing environment, without exploiting the new possibilities with the PostScript format and the PDF portable document format, until Gäßlein and Niepraschk wrote the *picture* extension announced by Lamport some ten years before.

Let see what are the enhancements introduced by the extension realized by Gäßlein and Niepraschk;

1. First of all the enhancements rely on the drivers that are being used to display or to print the document. More precisely when the *latex* program is used, the `\special` commands to the driver contain only PostScript language commands; this implies a transformation of the resulting DVI file into PostScript one by means of *dvips*, and possibly a second transformation to the PDF format by means of *ps2pdf*. On the opposite, if the document is processed with *pdflatex*, then the `\special` commands contain only the PDF language commands. Therefore the extension is fully compatible with the typical output formats provided by the most popular typesetting engines; but this performance is fully automatic and users need not bother about these details.

2. The output file size very often is smaller since the actual drawing computations are performed by the suitable drivers.

3. Here we describe the 2009 extension, therefore

there are also some commands that were not described by Lamport.

4. One of the limitations of the original environment was the slope of lines and vectors; in the first implementation of the extension the slope coefficients had to be integers non higher than 1000, in absolute value, therefore implementing Lamport's description of 1994; but the 2009 enhancements removes even this limitation, and the slope coefficients can be any fractional decimal number (well, yes, not too large, not higher than 16383.99998 which corresponds to the largest dimension in points that any TEX system typesetting engine can handle). Now line and vector slopes should not have any sensible limitation.

5. The above is valid also for vectors; even better, now it's possible to pass an option to the package so that it can draw the arrow tips in "LATEX style" or in "PostScript style". In LATEX style the joining sides to the arrow tip are slightly concave, and the arrow base is straight; in PostScript style they form a polygon that resembles a stealth aircraft.

6. Circles and quarter circles were available in a limited set; now they can be drawn in any size both filled and unfilled.

7. Line thicknesses could be specified only as `\thinlines` (default) and `\thicklines` (twice as thick compared to `\thinlines`); only vertical and horizontal lines could receive the thickness specified by means of the command `\linethickness`⟨*dimension*⟩. Now `\linethickness` can modify the thickness of all sorts of lines, Bézier splines included.

8. "Ovals", frames with rounded corners, could have the corner quarter circle with an automatic setting of its radius, in any case not larger that about 15pt – about 5mm –, and they could not receive a radius specification that might be chosen by the composer; of course this radius should not exceed the half length of the shorter frame sides (that is, half of the distance of the longer straight lines that form the longer sides of the frame) but the radius can now be specified as an optional argument to the `\oval` command so that the created frame can have a very different look when a smaller radius is chosen compared to the same sized frame with a larger corner radius.

9. Quadratic and cubic Bézier splines are now generated with the driver commands and they result smooth curves, not lines with a rough contour, due to the juxtaposition of many small dots. The possibility of specifying the number of dots is available even now, but it is mostly for backwards compatibility, although, even now, dotted splines might be used for special purposes. In any case they do not suffer any magnification when seen on the screen; they are scalable vector strokes. The previous command `\bezier` is maintained with its compulsory specification of the number of points to use, but two new commands with an optional specification of the number of points, are introduced, `\qbezier` for tracing quadratic Bézier splines, and `\cbezier` for tracing cubic Bézier splines; this last command was not described in Lamport, and is a completely new command to the package.

10. Up to this point the traditional commands are discussed and the differences with the past original environment are shown. The last extension of the pict2e, published in the second half of 2009, adds some other commands that draw other lines but in general don't require the use of `\put` to place these lines in a special position. Of course they may be shifted with `\put`, but this might come handy when fine tuning the picture, without being really necessary to use `\put`.

11. A first exception to the above statement is the new macro `\arc` that is a generalization of `\circle` (with or without asterisk; in both cases the command with asterisk produces a filled contour) and requires to put the arc center in a specific position, so that the whole command must be set as an argument to `\put`; the `\circle` command is used as:

    `\put(`⟨*x*⟩`,`⟨*y*⟩`){\circle`⟨***⟩`{`⟨*diameter*⟩`}}`

    and similarly the `\arc` command is used as:

    `\put(`⟨*x*⟩`,`⟨*y*⟩`){\arc`⟨***⟩`[`⟨*ang1*⟩`,`⟨*ang2*⟩`]{`⟨*radius*⟩`}}`

    The arc has its center at point `(`⟨*x*⟩`,`⟨*y*⟩`)`, and it will go from the angle ⟨*ang1*⟩ to the angle ⟨*ang2*⟩; angles are in sexagesimal degrees and are positive in the anticlockwise direction; if the optional angles are not specified, the full circle is drawn. The arc is actually drawn from the smaller angle to the larger one, so that the order in which ⟨*ang1*⟩ and ⟨*ang2*⟩ are given is not important.

12. The following lines do not require the `\put` command:

    `\Line(`⟨*x1*⟩`,`⟨*y1*⟩`)(`⟨*x2*⟩`,`⟨*y2*⟩`)`
    `\polyline(`⟨*x1*⟩`,`⟨*y1*⟩`)(`⟨*x2*⟩`,`⟨*y2*⟩`)...(`⟨*xN*⟩`,`⟨*yN*⟩`)`
    `\polygon(`⟨*x1*⟩`,`⟨*y1*⟩`)(`⟨*x2*⟩`,`⟨*y2*⟩`)...(`⟨*xN*⟩`,`⟨*yN*⟩`)`
    `\polygon*(`⟨*x1*⟩`,`⟨*y1*⟩`)(`⟨*x2*⟩`,`⟨*y2*⟩`)...(`⟨*xN*⟩`,`⟨*yN*⟩`)`

The first command is just the segment that joins the two points identified by the two pairs of coordinates. The second command is a sequence of segments that join with one another in the order of the N specified pairs of coordinates; we have seen it at work in the example shown in the Introduction. The third command is a closed polygon whose vertices are sequentially shown by the N pairs of coordinates. The fourth command is similar but it draws a filled polygon.

13. In order to draw the various lines and curves, the internal commands make use of the "turtle graphics" commands used within both the PostScript and PDF languages. Such elementary commands are available to the user also through package pict2e; they are:

    `\moveto(`⟨*x*⟩`,`⟨*y*⟩`)`
    `\lineto(`⟨*x*⟩`,`⟨*y*⟩`)`
    `\curveto(`⟨*x2*⟩`,`⟨*y2*⟩`)(`⟨*x3*⟩`,`⟨*y3*⟩`)(`⟨*x4*⟩`,`⟨*y4*⟩`)`

    and a few more that the reader may find in the documentation, (Gässlein *et al.*, 2009). These commands may be used in any order, except `\moveto` that must fix the first positioning of the drawing pen; but in order to finish the path it is optional to use `\closepath` in order to draw a line from the last point to the initial one, and then it is necessary to use `\strokepath` in order to actually draw the path or `\fillpath` in order to fill the path with the default color.

14. The initial and final points of an open path may be controlled with the commands `\buttcap` (just cut the path at the end points), or `\roundcap` (adjust the end points with a filled semicircle), or `\squarecap` (adjust the end points with a filled half square); in general with line art the `\roundcap` should be preferable, but sometimes it's better to use one of the other two kinds of end point finishing, figure 1.

15. Similarly the joins between adjacent segments of a polyline or a polygon may be adjusted with the three commands `\miterjoin`[1], `\roundjoin`, and `\beveljoin`, as shown in figure 2.

3 Examples

There are dozens of examples in the GUIT documentation, (Beccari, 2011), where every line art picture has a small legend containing the author name and the program used for producing it.

1. In the documentation, (Gässlein *et al.*, 2009), this command is erroneously spelled `\mitterjoin`.

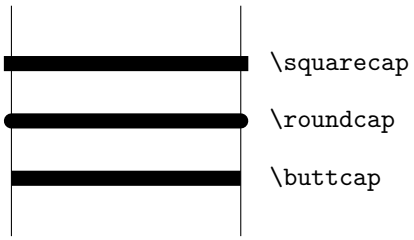


Figure 1: Various end point for segments of equal length

Figure 2: Various joins

Since this book, as a collective effort of the Italian Group of TEX Users, is completely downloadable from the GUIT site `http://www.guit.sasup.it/downloads/GuidaGuIT.pdf`[2], including the source files, where the interested reader can find plenty of ideas and useful "tricks"; well, yes, any software user develops his/her own shortcuts and tricks in order to simplify recurrent operations; he or she will write himself or herself suitable macros which are the ordinary tricks any LATEX user eventually makes use of. But this would be done also with any other more sophisticated drawing package, such as Ti*k*Z Tantau (2010), or PSTricks, van Zandt (2003).

Here we present few examples, possibly with their source code, in order to see the modern *picture* environment at work.

A heptagon

We compute the vertices of a heptagon inscribed into a circle with a diameter of 6cm by means of a pocket calculator:

$$v_1 = (1.3017, -2.7029)$$
$$v_2 = (2.9248, -0.6676)$$
$$v_3 = (2.3455, 1.8705)$$
$$v_4 = (0, 3)$$
$$v_5 = (-2.3455, 1.8705)$$
$$v_6 = (-2.9248, -0.6676)$$
$$v_7 = (-1.3017, -2.7029)$$

Then we set up the picture environment (within a *figure* environment, so we don't need to do anything to limit the scope of the `\unitlength` assignment) with the following code:

2. The GUIT home page is going to change to `http://www.guitex.org`. At the time of writing this transfer has not been done yet.

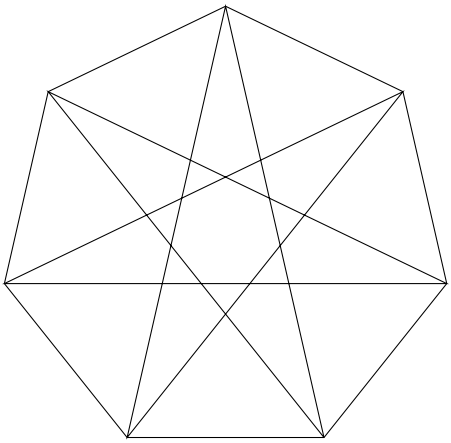


FIGURE 3: A heptagon and a star with seven vertices

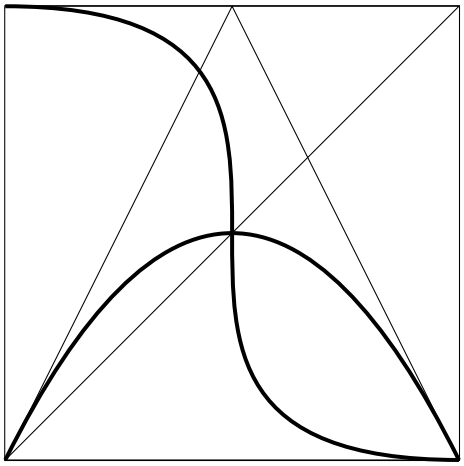


FIGURE 4: A quadratic and a cubic spline

```
\unitlength=1cm
\begin{picture}(6,6)(-3,-3)
\polygon(1.3017,-2.7029)(2.9248,-0.6676)%
    (2.3455,1.8705)(0,3)(-2.3455,1.8705)%
    (-2.9248,-0.6676)(-1.3017,-2.7029)
\polyline(1.3017,-2.7029)(0,3)%
    (-1.3017,-2.7029)(2.3455,1.8705)%
    (-2.9248,-0.6676)(2.9248,-0.6676)%
    (-2.3455,1.8705)(1.3017,-2.7029)
\end{picture}
```

Figure 3 contains also the seven pointed star inscribed in the heptagon.

Some splines

We draw some splines inside a square with sides 6cm long; a quadratic spline has its two nodes at the square base vertices, and the control node at the center of the upper side. A cubic spline uses the four square vertices as end and control nodes:

```
\unitlength=1cm
\begin{picture}(6,6)(-3,-3)
\put(-3,-3){\framebox(6,6){}}
\polyline(-3,-3)(0,3)(3,-3)
\polyline(-3,-3)(-3,3)(3,-3)(3,3)
\linethickness{1.5pt}
\qbezier(-3,-3)(0,3)(3,-3)
\cbezier(-3,-3)(-3,3)(3,-3)(3,3)
\end{picture}
```

Figure 4 displays the result; if you can read this document on the screen, you can magnify the picture and you can check the vector nature of such splines. You may observe the effect of the `\linethickness` assignment on the splines themselves. Figure 4 contains also the polylines that join the nodes and control points, so that it's easier to see the effect of these "control segments".

An electric circuit

Many years ago, at the end of the eighties, when only the *picture* environment was available, I needed to draw circuit diagrams; I had so many circuit diagrams to insert in my book that I needed to create suitable macros for drawing the circuit components and their connections to the various circuit nodes; of course every component had to be identified with a symbol and optionally should be assigned a value with the proper units.

Nowadays there are modular packages to work with Ti*k*Z, circuitikz, REDAELLI (2009), or with PSTricks, pst-circ, VOSS (2011); at that time there was nothing, or at least nothing I was aware of.

In my department there was a very good expert of technical drawing, and for my previous books I had asked him to draw my circuits; these drawings had to be glued to the camera ready copy, because at that time it was very difficult to insert graphical files into a document; not impossible, but difficult. The publisher, in any case, did not want any kind of file; he wanted only camera ready copies. This procedure was pretty lengthy; draw my circuits by hand, pass them to the technician with suitable descriptions about dimensions, lettering, line thicknesses, and the like; after the drawings were done, careful control of the correctness, the proper position of the labels, the right drop of indices, and any possible typo; start again with the second drafts, and so on.

Therefore I decided to write a personal package containing all the circuit macros; that had to work as an interface between the user and the *picture* environment with its internal macros. It took about two weeks; afterwords I had an almost complete circuit drawing TEX program. At that time, of course, arbitrary sloped lines were done by juxtaposition of a multitude of little dots, as well as quarter, half and full circles of any diameter. One port components were automatically drawn as vertical or horizontal elements; connections would make automatically the necessary bends in order to reach the destination nodes; two ports and four-pole devices were set in the proper orientation in order to avoid crossing their connections; opera-

tional amplifiers, nullators, norators and nullors were correctly designed; block diagrams with their signal flows, their branching nodes, their summing points, and so on, were at hand. The unit length was parametrized to the current font 'ex' unit, so that the circuit diagram would scale together with the size of the surrounding text font.

I saved a lot of time using my macros and the technical expert eventually congratulated with me admitting that my drawings were more consistent than his own.

When the important pict2e package became available in 2003, I started to eliminate all references to the old tiny-point-overlay technique, and promptly switched to the new technology.

I eventually added the logical components, so that this private package is almost complete. What is provided by circuitikz is much more complete, and this is the main reason why my package remains private. I keep using it for no other reason than compatibility with the past book files I wrote long ago, from which I often pick up some parts in order to assemble short tutorials for students that ask me some explanations.

Actually the package is too long to publish; therefore I will not show how the user commands are realized with the internal modern *picture* environment ones. I show just the source code for drawing the circut diagram of a band elimination filter:

```
\begin{circuito}(75,35)
\hconnect(0,0)(19,0)\HPolo(20,0)(65,0)
\polo(20,25)[30,25]\polo(65,25)[75,25]
\hconnect(66,0)(75,0)
\R(75,0)(75,25){R\ped{L}}
\E(0,0)(0,25){E}
\R(0,25)(19,25){R\ped{G}}
\serie*(30,0)(21,25)C{C_1}-L{L_1}
\parallelo(30,25)(55,25)L{L_2}-C{C_2}
\serie(55,0)(64,25)C{C_3}-L{L_3}
\nodi(55,0)(55,25)(30,25)(30,0)
\end{circuito}
```

and you can see the result in figure 5. As you can see the component and connection macros are really user friendly and the total amount of code for drawing a complicated circuit diagram[3] is very limited; if you are reading this file on screen you can magnify the image of figure 5 and verify that the whole drawing is a scalable vector one. You can also recognise that the resistors are drawn by means of the \polyline command with the \miterjoin specification for the connection of the various segments.

3. Actually the circuit diagram is not complicated at all; the complication is hidden in the user macros, especially those for inductors, where the various Bézier cubic splines are properly described and connected to one another.

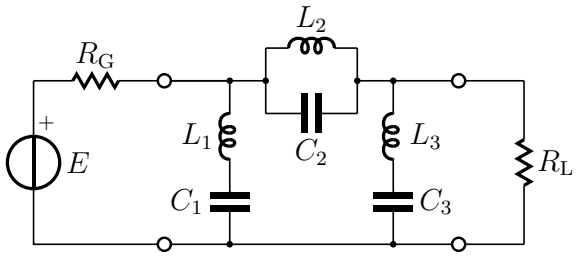


Figure 5: A band elimination filter

A Cartesian diagram

While teaching the synthesis of electrical circuits I often needed Cartesian diagrams of their performances; in figure 6 the squared magnitude of a fifth order elliptical filter characteristic function is plotted. The name "elliptical" derives from the use of first and second kind of elliptical integrals and functions. The diagram is just qualitative, although it would not have been a problem to compute the actual points by means of a suitable program, for a qualitative diagram the extreme points and the peaks and zeros should be sufficient. The whole diagram had to be also shown with a beamer, therefore a beamer presentation was made containing the same code:

```
\unitlength=0.9mm
\begin{picture}(80,60)(-40,-5)
  \VECTOR(-40,0)(40,0)
  \Zbox(40,-2)[tr]{\omega}
  \VECTOR(0,-1)(0,55)
  \Zbox(-1,55)[tr]{|F|^2}
  \multiput(-35,5)(4,0){18}%
    {\line(1,0){2}}
  \Zbox(2,7)[bl]{1}
  \multiput(-35,45)(4,0){18}%
    {\line(1,0){2}}
  \Zbox(1,46)[bl]{H^2}
  \multiput(-2,15)(4,0){4}%
    {\line(1,0){2}}
  \Zbox(1,16)[bl]{H}
  \LINE(-10,0)(-10,-1)\Zbox(0,-2)[t]{0}
  \Zbox(-10,-2)[t]{-1}
  \LINE(10,0)(10,-1)\Zbox(10,-2)[t]{1}
{\linethickness{1.5pt}%
  \cbezier(-12.5,55)(-12,40)(-12,40)%
        (-10,5)
  \cbezier(-10,5)(-10.1,0)(-9.75,0)%
        (-9.5,0)
  \cbezier(-9.5,0)(-9,0)(-9,5)(-8.5,5)
  \cbezier(-8.5,5)(-7.75,5)(-7.75,0)%
        (-7,0)
  \cbezier(-7,0)(-5.5,0)(-5.5,5)(-4,5)
  \cbezier(-4,5)(-2,5)(-2,0)(0,0)
  \cbezier(-13,55)(-13.75,45)(-13.75,45)%
        (-14.5,45)
  \cbezier(-14.5,45)(-15.75,45)%
        (-15.75,45)(-17,55)
```

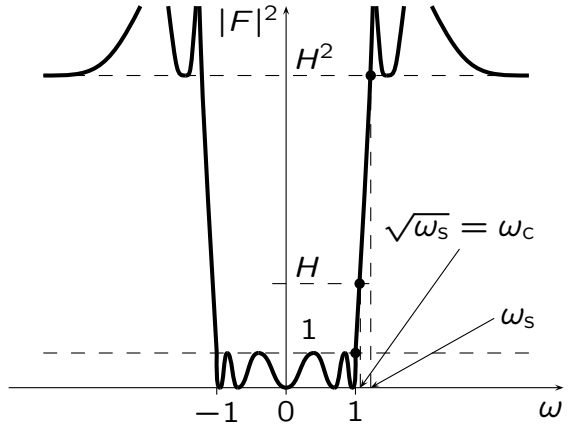


FIGURE 6: The squared magnitude of the characteristic function of an elliptical filter

```
  \cbezier(-21,55)(-26,45)(-28,45)%
          (-35,45)
%
  \cbezier(12.5,55)(12,40)(12,40)(10,5)
  \cbezier(10,5)(10.1,0)(9.75,0)(9.5,0)
  \cbezier(9.5,0)(9,0)(9,5)(8.5,5)
  \cbezier(8.5,5)(7.75,5)(7.75,0)(7,0)
  \cbezier(7,0)(5.5,0)(5.5,5)(4,5)
  \cbezier(4,5)(2,5)(2,0)(0,0)
  \cbezier(13,55)(13.75,45)(13.75,45)%
          (14.5,45)
  \cbezier(14.5,45)(15.75,45)(15.75,45)%
          (17,55)
  \cbezier(21,55)(26,45)(28,45)(35,45)
}%
  \put(10.6,15){\circle*{1.5}}
  \put(12.2,45){\circle*{1.5}}
  \put(10,5){\circle*{1.5}}
  \multiput(12.2,0)(0,4){11}%
     {\line(0,1){2}}
  \multiput(10.7,0)(0,4){4}%
     {\line(0,1){2}}
  \VECTOR(25,20)(10.7,0)
  \VECTOR(30,10)(12.2,0)
  \Zbox(25,21)[b]%
  {\sqrt{\omega\ped{s}}=\omega\ped{c}}
  \Zbox(31,10)[lc]{\omega\ped{s}}
\end{picture}
```

There are just two custom commands, `\Zbox` and `\VECTOR`, that were defined on purpose in order to speed up the data input; the former is short for inserting a zero dimension box at the proper coordinate; the latter is similar to the standard `\Line` command but applied to vectors. The other unusual macro `\LINE` is totally equivalent to `\Line`; I just had defined it before the 2009 enhancement of the pict2e package.

Figure 6 displays the whole diagram as scalable vector graphics, and as you can see the important messages about the filter characteristic function properties are fully and clearly expressed. This is just an example among the many such diagrams I used in my books and in my presentations. It was really worth the little time spent in defining a couple of service macros. Of course I could have used the plothandlers module of the Ti*k*Z library, or the tikz-3dplot Ti*k*Z extension package, not to mention the modules associated with PSTricks. But I saved myself the study of the 700 plus pages of the Ti*k*Z documentation or a similar amount for the PSTricks bundle. The fonts in figure 6 are just the ones I designed myself for presentations; they have been fully described in BECCARI (2007), and are available on any complete recent distribution of the TEX system.

4 Conclusion

As I said in the Introduction the *picture* environment is a very simple one, with few and specific drawing commands; the documentation is so simple that less than 10 pages suffice. At the same time these simple commands may be used to create more complex macros and eventually produce professional drawings. Certainly this environment cannot compete with more elaborate ones, such as those provided by the packages Ti*k*Z and PSTricks; but the latter ones require a steep learning curve, while the former one can be mastered in a few minutes. Very often the results obtained with the *picture* environment, completed by the recent enhancements provided by the pict2e package, are fully acceptable: it's possible to create complicated diagrams as well as simple symbols; it is possible to use this environment to place background or foreground images or symbols "wherever" on the page, even outside the margins; in BECCARI (2011) it is shown also how to make strange and unusual tables, Cartesian diagrams, and any sort of mix between line art and included pictures. In any case if any lettering is placed in the drawing, it surely uses the same fonts as those used for the text, thus eliminating the usual risk that may occur when using external drawing software.

I believe that beginners would find this enhanced environment the right first step for programmed drawings; with minimum effort they can reach really good results.

References

BECCARI, C. (2007). «I font per le slide LATEX resuscitati». *ArsTEXnica*, (4).

BECCARI, C. (a cura di) (2011). *Introduzione all'arte della composizione tipografica con LATEX*. GUIT.

FRANZ, M. (2003). «The crop package». TEX documentation attached to the crop.sty package.

GÄSSLEIN, H., NIEPRASCHK, R. e TKADLEC, J.

(2009). «The pict2e package». TEX documentation attached to the pict2e.sty package.

Knuth, D. E. (1990). *The TEXbook*, volume A di *Computers and Typesetting*. Addison Wesley, 18ª edizione.

Lamport, L. (1994). *A document preparation system – LATEX – User's guide and reference manual*. Addison Wesley Publishing Company, Reading, Massachusets, 2ª edizione.

Maclaine-cross, I. (2008). «The curves package». TEX documentation attached to the curves.sty package.

McPherson, K. (2000). «Displaying page layout variables». TEX documentation attached to the layout.sty package.

Niepraschk, R. (2010). «The eso-pic package». TEX documentation attached to the eso-pic.sty package.

Redaelli, M. A. (2009). «CircuiTikZ». TEX documentation attached to the circuitikz.sty package.

Tantau, T. (2010). «The TikZ and pgf packages». TEX documentation attached to the tikz.sty package.

van Zandt, T. (2003). «Pstricks – PostScript macros for generic TEX». TEX documentation attached to the pstriks.sty package.

Voss, H. (2011). «pst-circ a PSTricks package for drawing electric circuits». TEX documentation attached to the pst-circ.sty package.

Wichura, M. (1987). *PiCTEX*. Department of Statistics, University of Chicago.

Wilson, P. e Robertson, W. (2009). «The layouts package». TEX documentation attached to the layouts.sty package.

▷ Claudio Beccari
Villarbasse (TO)
claudio dot beccari at gmail dot com

Extending ConTEXt MkIV with PARI/GP

Luigi Scarso

Abstract

This paper shows how to build a binding to PARI/GP, a well known computer algebra system, for ConTEXt MkIV, showing also some examples on how to solve some common basic algebraic problems.

Sommario

Questo articolo mostra come costruire un *binding* a PARI/GP, un noto sistema di computer algebra, per ConTEXt MkIV; mostra inoltre alcuni esempi di come risolvere alcuni problemi algebrici basilari.

1 Introduction

PARI/GPPARI is a relatively small computer algebra system that comes as C library (`libpari`) and an interpreter (gp) for its own language (GP) built on upon the same library. Although it has respectable capabilities on symbolic manipulations, it has also an extensive algebraic number theory module, hence it can perform, due to the highly optimised C library, complex numeric calculations very quickly and accurately. PARI stands for "Pascal ARIthmétique" (the very first choice was the Pascal language, dropped soon for C), while GP originally was GPC for "Great Programmable Calculator, but also the C was dropped for unknown reasons. The current stable version is `2.3.4`.

ConTEXt MkIV is today the most advanced macro format that uses LuaTEX. It's still under active development: a milestone release is planned for spring 2012, when LuaTEX will reach the `1.0` version (currently is `0.70`). ConTEXt MkIV has an integrated support for Lua and MetaPost and offers a superb (as of today) management of OpenType fonts. Its philosophy is different from LATEX: ConTEXt MkIV is monolithic, while LATEX is more modular.

In this paper we will show a way to "extend" ConTEXt MkIV with PARI/GP and some examples on how to use this powerful library — hopefully it should be simple to adapt them for luaLATEX. We will see how to evaluate summations like $\sum_{k=0}^{30} \frac{4(-1)^k}{2k+1}$, calculating the exact value or an approximation; but also we will able to evaluate series like $\sum_{k=0}^{\infty} \frac{4(-1)^k}{2k+1}$ or symbolic sums as $\sum_{k=0}^{3} \frac{1}{x^2+k}$. PARI/GP is also able to solve equations like $x^5 + \arctan(x)x^3 + 2x^2 + x + 1 = 0$ and we will see how to use it together MetaPost to plot the roots of $P[X, Y] = X^3 - X - Y^2$. Finally, the last section is dedicated to show how to solve a simple problem of algebraic geometry using a few PARI/GP functions: the reader is encouraged to implement by himself the code.

2 Build the Lua binding

2.1 What is a binding ?

It's well known that ConTEXt MkIV uses LuaTEX as a typesetting engine, but maybe it's little known to the TEX-user that Lua by itself is a complete high level programming language. It's a scripting language, with a syntax quite similar to Pascal, and it's used either as *embedded language* to enhance an application (e.g. to build plug-ins) or as *glue language* to "connect" together several object libraries — exactly the same as in SageSAGE, where the glue language is Python. In the latter case Lua is *extended* with the new libraries that become Lua modules: they can be built in into the `lua` interpreter at compile time (as in the GSL ShellGShell program) or loaded at runtime — which is the case of the extension shown in this paper.

Nowadays it's quite common that a scripting or dynamic language can load at runtime an object library, written in C or C++ and then compiled. Usually it's necessary to write some C code that act as an interface between the library and the language: this process is called "build the binding for the library". Here, the developer decides which symbols of the library (i.e. functions, classes, variables and constants) export and how they are seen from the language (under which name, for example). This is a delicate phase, because one must know the conventions of the language, its interface code (the "Application Program Interface" or API of the language), the API of the target library and write the appropriate C code for each symbol to export: for libraries written in C these APIs are usually organised in the *header files* (with suffix `".h"`) that contain the declarations of each function, variable or constant defined in the library — but not always all of them can be exported: the developer must also know the set of useful symbols to export the set of admissible.

It's clear that each language has its own peculiarities, but many high level languages are implemented in C of C++ and hence shared a common

substrate; also, an object library must rigorously follows the conventions of the host Operating System, hence its structure cannot depend to the high level language.

Here is where SWIG enters to play. SWIG (*Simplified Wrapper and Interface Generator*, seeSWIG) is a program that helps the developer to build bindings and, for some libraries, practically can build automatically a binding only by reading all the headers files. SWIG reads a driver file, the so-called *interface file* `".i"`, executes its instructions and ends with the binding: most of the time it's C code that must be compiled into an object module. The power of SWIG is that the user can select the target high level language so that the same interface file can be used for different languages (i.e. Lua, Python, Ruby and so on); on the other side, the code of the binding is more complicated than an binding manually built by a developer.

2.2 Build the Lua binding for PARI/GP

The Lua API are completely listed in the Lua bookPIL and they describe a simple and robust mechanism: basically every C function that wants to interact with the Lua interpreter uses a stack to exchange data. The stack is modified by a relatively small set of functions that act on the *Lua state*, a global data structure that also keeps track of unused objects and calls the garbage collector when necessary. Hence every C function of the binding must only take care of calling the function of the target library with the right arguments and use the stack to exchange the *in* (input to the target function) and/or *out* (output to the Lua interpreter) values.

The interface file `pari.i` for the binding of `libpari`, the object library of PARI/GP, is shown below and its instructions are quite simples: they basically say "read all the headers and produce the binding". This is for example the role of the `%include "pari/paritype.h";` instruction, that just says "read the header `paritype.h` which is in the `pari` folder and write the binding code"; but we can also map some `libpari` functions into something else, as in

```
GEN uti_mael2(GEN m,long x1,long x2)
{return mael2(m,x1,x2);}
```

where the `liblua` macro `mael2` is wrapped into the C function `uti_meal2` for sake of simplicity.

The binding is then built with

```
swig  -lua pari.i
```

This is the complete interface file `pari.i` used under Linux 32 bit: the header files are in the subfolder `pari` of the folder that contains the build script.

```
%module pari
%{
#include "pari.h"
ulong overflow;
%}

%ignore gp_variable(char *s);
%ignore setseriesprecision(long n);
%ignore  killfile(pariFILE *f);

%include "pari/paritype.h";
%include "pari/parisys.h";
%include "pari/parigen.h";
%include "pari/paricast.h";
%include "pari/paristio.h";
%include "pari/paricom.h";
%include "pari/parierr.h";
%include "pari/paridecl.h";
%include "pari/paritune.h";
%include "pari/pariinl.h";

%inline %{
GEN uti_mael2(GEN m,long x1,long x2)
{return mael2(m,x1,x2);}
GEN uti_mael3(GEN m,long x1,long x2,long x3)
{return mael3(m,x1,x2,x3);}
GEN uti_mael4(GEN m,long x1,long x2,long x3,
long x4)
{return mael4(m,x1,x2,x3,x4);}
GEN uti_mael5(GEN m,long x1,long x2,long x3,
long x4,long x5)
{return mael5(m,x1,x2,x3,x4,x5);}
GEN uti_mael(GEN m,long x1,long x2)
{return mael2(m,x1,x2);}
GEN uti_gmael1(GEN m,long x1)
{return gmael1(m,x1);}
GEN uti_gmael2(GEN m,long x1,long x2)
{return gmael2(m,x1,x2);}
GEN uti_gmael3(GEN m,long x1,long x2,long x3)
{return gmael3(m,x1,x2,x3);}
GEN uti_gmael4(GEN m,long x1,long x2,long x3,
long x4)
{return gmael4(m,x1,x2,x3,x4);}
GEN uti_gmael5(GEN m,long x1,long x2,long x3,
long x4,long x5)
{return gmael5(m,x1,x2,x3,x4,x5);}
GEN uti_gmael(GEN m,long x1,long x2)
{return gmael2(m,x1,x2);}
GEN uti_gel(GEN m,long x1)
{return gmael1(m,x1);}
GEN uti_gcoeff(GEN a,long i,long j)
{return gcoeff(a,i,j);}
GEN uti_coeff(GEN a,long i,long j)
{return coeff(a,i,j);}
%};
```

The binding is quite straight: almost every symbol of `libpari` has a counterpart in Lua with the same name; the symbols "private" are stated in `paripriv.h`, which is not listed in `pari.i` — they aren't exported and hence they are not reachable from Lua.

The build script (for Linux) assumes SWIG and PARI/GP installed under `/opt/swig-2.0.2`:

```
/opt/swig-2.0.2/bin/swig  -lua pari.i
gcc -ansi \
    -I./pari -I/opt/swig-2.0.2/include \
```

```
    -c pari_wrap.c  -o pari_wrap.o
gcc -Wall -ansi -shared -I./pari \
    -I/opt/swig-2.0.2/include -L./ \
    -L/opt/swig-2.0.2/lib pari_wrap.o \
    -lpari -lm -o pari.so
```

Once compiled, the `pari.so` is suitable to be loaded as a Lua module with `require("pari")`.

As a final note for this section, the same steps can be followed under Windows using MinGWMINGW or with GUBGUB to cross-compile the library in a host system (Linux) for a target system (Windows) — as is the case of this paper, where the examples use a cross-compiled dll `libpari`.

3 Examples

3.1 Summations

As we said briefly in the introduction, PARI/GP has its own language GP, with more than 450 functions, and its interpreter, the gp program. Most of the time these functions are in a one-to-one correspondence with the functions exported by the library `libpari`, but sometimes there are some "sugar syntactic" constructs for the sake of simplicity. In any case, `libpari` has the `gp_read_str(char *)` function that evaluates a GP sentence and returns the result, so that on the Lua side it's possible to use both the library and the GP language. The library is usually faster than GP and it has a finer control — which usually also means that it's necessary to write more code.

In this first example, we will see how to calculate exactly a summation. The GP function is `sum(X,a,b,expr,start)` that stands for $\sum_{X=a}^{b} expr(X,\cdot)$, where `start` is the initial value of $expr(X,\cdot)$:

```
\startluacode
require("pari")
pari.pari_init(4000000,500000)
document = document or {}
document.lscarso= document.lscarso or {}
local function sum(X,a,b,expr,start)
   local avma = pari.avma
   local start = start or '0.'
   local res =
    pari.gp_read_str(string.format(
     "sum(%s=%s,%s,%s,%s)",X,a,b,expr,start))
   res = pari.GENtoTeXstr(res)
   pari.avma = avma
   return res
end
document.lscarso.sum = sum
\stopluacode

\starttext
\startTEXpage
\startformula
\sum_{k=0}^{30}\frac{4(-1)^k}{2k+1}=
  \ctxlua{context(document.lscarso.sum(
   "k",0,30,"4*(-1)^k/(2*k+1)","0"))}
\stopformula
\stopTEXpage
\stoptext
```

that gives

$$\sum_{k=0}^{30}\frac{4(-1)^k}{2k+1}=\frac{5863013579100197316985228 4}{1847292006410659792986502 5}$$

Let's explain step by step the code. First we need to load the module with `require("pari")` — assuming that the library is in the standard path or in the current folder (see `CLUAINPUTS` in LUATEX for more details).

Next, we must avoid conflicts with others Lua functions. A common solution is to define a namespace (`document.lscarso` in this case), and a local function (`sum(X,a,b,expr,start)`) to be stated within the namespace (`document.lscarso.sum = sum`). This is a general issue when one defines its own module, not only for PARI/GP — it's the same problem of redefining TEX macros.

There is another issue with PARI/GP. Like Lua, PARI/GP also uses a stack but it has not a garbage collector, and every time it makes a calculation the result is not deleted; after a while the stack is full and the process aborts. Fortunately, it's easy to clear the stack: at the beginning of every function it's sufficient to record the initial position on the stack with `local avma=pari.avma` and then reset the stack with `pari.avma=avma` just before the return statement of the function. This is an issue with `libpari`, because most of GP functions manage the stack correctly.

After these notes, calling the GP `sum` function is a matter of calling `gp_read_str(char *)` with the right formatted string, which is trivial thanks to `string.format`, a standard LuaTEX function. Last but not least is `pari.GENtoTeXstr(GEN)`, a `libpari` function that translates a pari object (e.g. a fraction) into a TEX expression. It's important to note that the result is exact because we have imposed with `start=0` that all the values are in $\mathbb{Q}$: if we want an approximated value we just use `start=0.` (`0.` means "zero" as floating point value) and the result is

$$\sum_{k=0}^{30}\frac{4(-1)^k}{2k+1}=3.1738423371907494086902241 40$$

But we can do things a bit better. First, we want to control the *precision* of the result, i.e. how many digits to show. This is quite simple: the GP `default(.,.)` function can be used to get/set some internal constants and `realprecision` is what we need:

```
local function set_precision(prec)
```

```
   local avma = pari.avma
   local prec = math.floor(prec+0.5) or 28
   local res = pari.gp_read_str(
    string.format("default(realprecision,%s)",
      prec))
   res = pari.GENtostr(pari.gp_read_str(
    "default(realprecision)"))
   pari.avma = avma
   return res
end

local function get_precision(prec)
   local avma = pari.avma
   local res = pari.GENtostr(
    pari.gp_read_str(
     "default(realprecision)"))
   pari.avma = avma
   return res
end
```

Once we define the precision, we can extend the summation to "infinity", i.e. until the partial sums are stable within the precision. Of course this depends on the character of the series — in our case it's an alternating series. For this kind of series GP has the `sumalt(X=a,expr)` function that does the job:

```
local function sum_alternate(X,a,expr,prec)
   local avma = pari.avma
   local gp = document.lscarso.get_precision
   local oldprec = gp(prec)
   document.lscarso.set_precision(prec)
   local res=pari.GENtostr(pari.gp_read_str(
    string.format("sumalt(%s=%s,%s)",
                          X,a,expr)))
   document.lscarso.set_precision(oldprec)
   pari.avma = avma
   res=string.gsub(res,"(%d)","%1\\hskip0sp")
   return res
end
```

We can hence try to calculate the series with a precision of 800 digits:

```
\startformula
\sum_{k=0}^{\infty}\frac{4(-1)^k}{2k+1}=
\stopformula
\ctxlua{context(
  document.lscarso.sum_alternate(
  "k",0,"4*(-1)^k/(2*k+1)",800))}
```

Given that the result is quite long (see fig.1) with `string.gsub(res,"(%d)","%1\\hskip0sp")` we insert an invisible skip to help TEX to break the expression.

Of course this is a well known series: from $\arctan(1) = \pi/4$ one can calculate the Taylor expansion of $\arctan(x)$ around $x = 0$ with `taylor(expr,x)`:

```
local function taylor(expr,x)
   local avma = pari.avma
   local res = pari.gp_read_str(
    string.format("taylor(%s,%s)",expr,x))
   res = pari.GENtoTeXstr(res)
   pari.avma = avma
```

$$\sum_{k=0}^{\infty} \frac{4(-1)^k}{2k+1} \approx$$

3.14159265358979323846264338327950288419
71693993751058209749445923078164062862089
98628034825342117067982148086513282306647
09384460955058223172535940812848111745028
41027019385211055596446229489549303819
64428810975665933446128475648233786783165
27120190914564856692346034861045432664821
33936072602491412737245870066063155881
74881520920962829254091715364367892590360
01133053054882046652138414695194151160943
30572703657595919530921861173819326117931
05118548074462379962749567351885752724891
22793818301194912983367336244065664308602
13949463952247371907021798609437027705392
17176293176752384674818467669405132000568
12714526356082778577134275778960917363717
87214684409012249534301465495853710507922
79689258923542019956112129021960864034418
15981362977477130996051870721134999999837
29780499510597317328160963186

FIGURE 1: Evaluation of an alternating series with 800 digit precision.

```
   return res
end

$\mathrm{arctan}(x)=$
\startformula
\ctxlua{context(document.lscarso.taylor(
        "atan(x)","x"))}
\stopformula
```

i.e.

$$\arctan(x) = x - \frac{1}{3}x^3 + \frac{1}{5}x^5 - \frac{1}{7}x^7 + \frac{1}{9}x^9 - \frac{1}{11}x^{11} + \frac{1}{13}x^{13} - \frac{1}{15}x^{15} + O(x^{16})$$

The series is convergent in $x = 1$ (there are several proofs about this, e.g. see Leibniz), and the result is exactly $\arctan(1)$, hence

$$4\sum_{k=0}^{\infty} \frac{(-1)^k}{2k+1} = \sum_{k=0}^{\infty} \frac{4(-1)^k}{2k+1} = 4\frac{\pi}{4} = \pi \quad .$$

It's important to note that theoretically this series has a slow convergence to π (it's hence a bad choice to calculate π) but *in practice* it can be used with PARI/GP to give quickly an high precision result — this is the power of the library.

Before going further, let's consider again this summation:

```
\startformula
\sum_{k=0}^{3}\frac{1}{x^2+k}=
\ctxlua{context(document.lscarso.sum(
  "k",0,3,"1/(x^2+k)","0"))}
\stopformula
```

that gives

$$\sum_{k=0}^{3} \frac{1}{x^2+k} = \frac{4x^6+18x^4+22x^2+6}{x^8+6x^6+11x^4+6x^2}$$

PARI/GP is also capable of some symbolic calculations — it's not only a numeric library.

3.2 Continued fractions

A simple *finite* (canonical) continued fraction is a rational number q given by

$$q = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots + \cfrac{1}{a_n}}}}$$

where a_0 is an integer and $a_j, j > 0$ are strictly positive integers. Every rational number can be expressed with a finite continued fraction; if we consider a succession of finite continued fractions, for $n \to \infty$ we have an *infinite* (canonical) continued fraction, and every irrational number has an unique infinite continued fraction. For a finite c.f. $[a_0, a_1, a_2, \ldots, a_n]$ the rational number given by calculating all the intermediate fractions is usually termed as p_n/q_n, i.e.

$$[a_0, a_1, a_2, \ldots, a_n] = \frac{p_n}{q_n}$$

For example [0,3] = 1/3 and it's possible to show that $p_n/q_n = [a_0, a_1, a_2, \ldots, a_n]$ is the fraction in lowest terms. The GF `contfrac` function calculate (the vector of) the continued fraction of a rational number, while `contfracpnqn`, given a (finite vector of) continued fraction, return p_n, q_n. But the interesting point here is to use, given a *real* number with a fixed precision, the continued fraction to find its best rational approximation. The `libpari bestappr(x,A)` function calculates exactly what we need:

```
local function bestappr(x,A)
   local avma = pari.avma
   local x = tostring(x) or nil
   local A = math.floor(A+0.5)
   local res, bestx
   if x == nil then return nil,nil end

   bestx = pari.bestappr(pari.geval(
     pari.strtoGENstr(x)),
     pari.geval(
      pari.strtoGENstr(tostring(A))))
   res = {}
   res[1] = pari.GENtostr(bestx)
   res[2] = pari.GENtostr(
            pari.uti_gel(bestx,1))
   res[3] = pari.GENtostr(
            pari.uti_gel(bestx,2))
   pari.avma = avma
   return res[1],res[2],res[3]
end
```

Note that the return value is an array with 3 components, namely p_n/q_n, p_n, q_n. We also use `pari.uti_gel`, the *wrapped* version of `libpari gel` function, to access an array by components.

Instead of an arbitrary real number, we choose π because the libpari `mppi(long)` function gives π with the required precision.

```
\startluacode
local collect = {}
local avma = pari.avma
local prec = 800
document.lscarso.set_precision(prec)
avma = pari.avma
local pi = pari.mppi(prec)
local pi_str = pari.GENtostr(pi)
pari.avma = avma
--print("=====>pi:",pi_str)
for d = 4,50000,1 do
   res,num,den =
     document.lscarso.bestappr(pi_str,d)
   collect[res] = {num,den,d}
end
context("\\starttabulate[|l|l|]")
context("\\HL")
context(string.format(
         "\\NC %s \\NC %s \\NC\\NR",
        "fraction","approx. value"))
context("\\HL")
for k,v in pairs(collect) do
   print( k, v[1]/v[2],v[3])
   -- context(k, v[1]/v[2],v[3])
   context(string.format(
    "\\NC %s \\NC %s \\NC\\NR",k,v[1]/v[2]))
end
context("\\stoptabulate")
\stopluacode
```

Note that we use p_n, q_n as a key for the dictionary `collect`, so we have just the set of results – i.e. we drop the same best approximations for different denominators. For a precision of 800 digits and a range of denominators between 4 and 50000 we have hence:

fraction	approx. value
333/106	3.1415094339623
104348/33215	3.1415926539214
16/5	3.2
13/4	3.25
22/7	3.1428571428571
355/113	3.141592920354
19/6	3.1666666666667
103993/33102	3.1415926530119

where the approximate values are due to the Lua floating point math.

3.3 Equations

Solving numeric equations in PARI/GP require more attention than other packages. The GP `solve(X=a,b,expr)` functions implements a very good algorithm but it works with one variable only and it fails if `expr` is not defined in $[a,b]$ and it hasn't a *variation* in $[a,b]$. This Lua wrapper `solve` tries to ensure that at in $[a,b]$ there is a variation, by evaluating the sign of `expr(a)*expr(b)`:

```
function solve(expr,X,a,b,prec)
   local av = pari.avma
   pari.gp_read_str(
    string.format(
     "default(realprecision,%s)",prec))
   local tr,res
   pari.gp_read_str(string.format("f(%s)=%s",
                    X,expr))
   tr = pari.gp_read_str(
    string.format(
     "if(f(%s)*f(%s)<0,1,0)",a,b))
   tr = pari.GENtostr(tr)
   tr = tonumber(tr)
   res = nil
   if (tr==1) then
    local expr=string.format(
     "solve(%s=%s,%s,%s)",X,a,b,expr)
    res = pari.gp_read_str(expr)
    res = pari.GENtostr(res)
   end
   return res,
    pari.GENtoTeXstr(
     pari.strtoGENstr(expr))
end
```

The following code tries to solve

$$x^5 + x^3\arctan(x) + 2x^2 + x + 1 = 0$$

for $x \in [-100{,}100]$ with a precision of 12 digits:

```
\startluacode
local solve = document.lscarso.solve
for a=-100,99,1 do
 local res,TeX,aa,bb =
   solve('x^5+atan(x)*x^3+2*x^2+x+1',
    "x",a,(a+1),12)
 if res  ~= nil then
  context(string.format(
   "$%s\\approx 0$ \\crlf
    for $x\\approx%s$\\par",
    TeX,res))
 else
     -- print("TeX="..TeX)
 end
end
\stopluacode
```

We have hence:
$x^5 + atan(x) * x^3 + 2 * x^2 + x + 1 \approx 0$
for $x \approx -1.47704735548$

PARI/GP has a rich set of functions for polynomials, and `solve` is not necessarily the best choice to find the roots of multivariate polynomials. The next and last example will show how to draw the real roots of $P[X,Y]$ with a given precision in a square region $[a,b] \times [a,b]$. First of all, we need to understand that with a fixed precision there is also an associated `zero`: with `precision=12` then `zero=1E-96`. Next, PARI/GP finds the complex roots of a univariate polynomial, so we need a `get_value` wrapper to evaluate $P(x,y)$ for $y \in [a,b]$ (with a given precision), so we have an expression in the x indeterminate that we will consider as a polynomial $P[X]$:

```
local function get_value(expr,X,a,prec)
   local avma = pari.avma
   pari.gp_read_str(string.format(
    "default(realprecision,%s)",prec))
   pari.gp_read_str(string.format(
    "%s=%s",X,a))
   local res = pari.gp_read_str(
    string.format("eval(%s)",expr))
   res  = pari.GENtostr(res)
   pari.avma = avma
   return res
end
```

The `polroots` function evaluates the roots and returns an array of roots where each root is separated into the real and complex components:

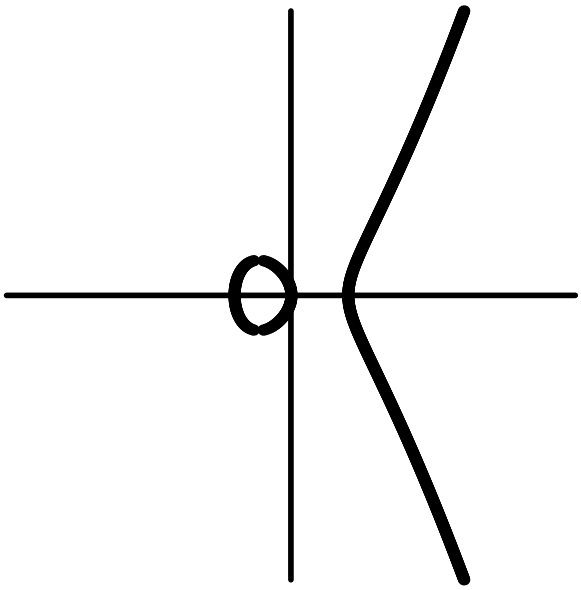
```
local function polroots(poly,prec)
   local avma = pari.avma
   pari.gp_read_str(string.format(
     "default(realprecision,%s)",prec))
   local poly = tostring(poly)
   local prec = tonumber(prec)
   local degree = pari.degree(
      pari.geval(pari.strtoGENstr(poly)))
   local roots = pari.roots(
    pari.geval(pari.strtoGENstr(poly)),prec)
   local res ={}
   for i=1,degree do
     local real_part,im_part =
      pari.GENtostr(pari.uti_gel(
      pari.uti_gel(roots,i),1)),
      pari.GENtostr(pari.uti_gel(
      pari.uti_gel(roots,i),2))
     res[#res+1]={real_part,im_part}
   end
   pari.avma = avma
   return res
end
```

Finally, we need to iterate y over $[a,b]$ and find the roots of $P[X]$. Instead of producing a table, we plot the values with the help of a MetaPost `\startMPpage...\stopMPpage` environment:

```
\startluacode
local poly = "x^3-x-y^2"
local step= 1/2^6
local results = {}
local limit = 5
local zero = '0.E-96'
local prec = 12
get_value = document.lscarso.get_value
polroots = document.lscarso.polroots
```

```
context("\\startMPpage")
context("pickup pencircle scaled 0.1pt;")
context(string.format("draw (-%s,0)--(%s,0);",
    limit,limit))
context(string.format("draw (0,-%s)--(0,%s);",
    limit,limit))
context("pickup pencircle scaled 0.2pt;")
for y=-limit,limit,step do
 local poly_x = get_value(poly,'y',y,prec)
 -- print("poly_x="..poly_x,y)
 local roots = polroots(poly_x,prec)
 for _,root in pairs(roots) do
  local real,imag = root[1],root[2]
  -- print("real="..real,"imag="..imag)
  if imag == zero then
     if real ==  zero then real = '0' end
     --print(string.format("(%s,%s)",real,y))
     context(string.format("draw (%s,%s);",
       real,y))
  end
 end
end
context("\\stopMPpage")
\stopluacode
```

With a precision of 12 digits and a square region of $[-5,5]$ we have then :



3.4 Solving a simple problem

In this last example we will show how to solve a simple problem of algebraic geometry, the approximation of a quarter of a circumference with a Bezier curve, with a mix of visual, numeric and symbolic techniques. The idea is to show the powerful of the extension *together* with the typographical capabilities of the TEX and MetaPost, rather than present some original ideas.

First, let's recall some basic definitions.
Given a field k (for us $k = \mathbb{Q}$ or $k = \mathbb{R}$), a *plain cubic Bezier curve* in parametric form is a subset $(x, y) \subset k \times k$ where

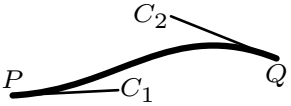
$$x = (1-t)^3 P_x + 3(1-t)^2 t C_{1x} + 3(1-t)t^2 C_{2x} + t^3 Q_x$$

$$y = (1-t)^3 P_y + 3(1-t)^2 t C_{1y} + 3(1-t)t^2 C_{2y} + t^3 Q_y$$

and $t \in [0,1] \subset k$. The points $\mathbf{P}$, $\mathbf{C_1}$, $\mathbf{C_2}$, $\mathbf{Q}$ are called *control points*; $\mathbf{P}$ is the *starting point* ($t = 0$), $\mathbf{Q}$ is the *ending point* ($t = 1$). MetaPost has a built in macro to draw cubic Bezier curves: for example this code

```
\starttext
\startMPpage
pair P,C[],Q;
P:=(0,0); C[1]:=(20,1);
C[2]:=(30,15); Q:=(50,7);
draw P..controls C[1] and C[2]..Q ;
\stopMPpage
\stoptext
```

draws:



A nice property is that the derivative of the curve at $\mathbf{P}$ and $\mathbf{Q}$ are $3(\mathbf{C_1} - \mathbf{P})$ and $3(\mathbf{Q} - \mathbf{C_2})$, as we can qualitatively see in the previous picture.

The next definition is the ring of polynomials in the two indeterminates $k[X, Y]$. Informally, it's the set of the finite expressions $P[X, Y]$ where $P[X, Y] = \sum_{i,j} a_{ij} X^i Y^j$ and $a_{ij} \in k$. The total degree of $P[X, Y]$ is $\max(i + j), a_{ij} \neq 0$ and if all a_{ij} are zero then we have the zero polynomial $\mathbf{0}$ (the zero of the ring $k[X, Y]$) which has degree undefined (but many authors set it to -1 or $-\infty$). Two polynomials $P_1[X, Y]$ and $P_2[X, Y]$ are identical if and only if $P_1[X, Y] - P_2[X, Y] = \mathbf{0}$; two polynomials with different degree are never identical.

Given $(x, y) \in k \times k$ and a polynomial $P[X, Y]$, with an abuse of notation we can consider the function $P(x, y) : k \times k \mapsto k$ where we "replace" X with x and Y with y in $P[X, Y]$ and then calculate the value ($P(x, y)$ it's an expression of k).

Now, a root of $P[X, Y]$ is a point $(x, y) \in k \times k$ for which $P(x, y) = 0$; given a $P[X, Y]$, the set of its roots is the *curve* of the polynomial and describe the properties of a curve is the main scope of the algebraic geometry (of course not only with two indeterminates and not only with $k = \mathbb{Q}$ or $k = \mathbb{R}$). In general it's a difficult task, but anyway we can already say something:

1. $P[X, Y] = X^2 + 1$ has no roots in $k \times k$ (remember that $k = \mathbb{Q}$ or $k = \mathbb{R}$);

2. $P[X, Y] = X$ has infinite roots because for every $y \in k$ $P(0, y) = 0$;

3. $P[X,Y] = \mathbf{0}$ (the zero polynomial) has infinite roots: for every point of $(x,y) \in k \times k$ $P(x,y) = 0$, so in this case the set of roots is $k \times k$.

Hence the curve of $P[X,Y]$ can be empty, infinite but properly included in $k \times k$, or all $k \times k$. Of great interest are also the set of intersections of two curves, which can be empty (i.e. $P_1[X,Y] = Y - 2$ and $P_2[X,Y] = Y$), finite ($P_1[X,Y] = X - Y$ and $P_2[X,Y] = X + Y$) and infinite ($P_1[X,Y] = \mathbf{0}$ and $P_2[X,Y] = Y$)

Let's consider $P[X,Y] = X^2+Y^2-1$. The "brute force" approach to have an idea of its curve $\mathcal{C}$ is, given a tolerance ϵ and a limit R, verify whenever $\left|x^2 + y^2 - 1\right| < \epsilon$ for $(x,y) \in [-R,R] \times [-R,R]$; a bit less primitive way is to solve $x_0^2 + y^2 - 1 = 0$ for y, with a fixed tolerance ϵ and the parameter $x_0 \in [-R,R]$. Both methods was shown in the previous sections, so here we take another way. Let's consider the set $P_t[X,Y] = Y - tX$ indexed by t; it's trivial to show that its curves are the lines $y = tx$. The *intersection* between $P(x,y) = 0$ and $P_t(x,y) = 0$ is also easy to calculate: just replace y^2 with $(tx)^2$. We have hence $x^2 + (tx)^2 - 1 = 0$ and

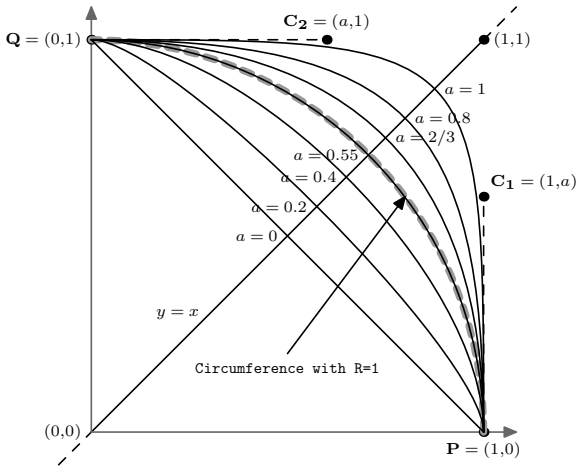
$$\mathcal{C} = \begin{cases} x^2 = & \dfrac{1}{1+t^2} \\ \\ y^2 = & \dfrac{t^2}{1+t^2} \end{cases}$$

or, after few simplifications,

$$\mathcal{C} = \begin{cases} x = & \pm\dfrac{1}{\sqrt{1+t^2}} \\ \\ y = & \pm\dfrac{t}{\sqrt{1+t^2}} \end{cases}$$

where t runs over k. This is called a *parametric form* of the curve $\mathcal{C}$, while $x^2+y^2-1=0$ is called the *implicit form* of the curve (the curve defined implicitly by X^2+Y^2-1). Although it was easy to obtain, it's not a nice formula, due to the presence of the square root factor; maybe we cannot avoid the fractions, but we would like to have integer exponents. In fact, in $k = \mathbb{Q}$, $\sqrt{1+t^2}$ not always is rational: for example for $t = 1/3$ we have the $\sqrt{10}$ factor which is not rational.

We can now come back to our problem, and make some assumptions. First, we will consider a quarter of a circumference $\mathcal{C}$ with $x \geq 0$ and $y \geq 0$ and radius 1. Next, we will assume that MetaPost is able to draw a circumference, that we will use as a reference. In this case, the best candidates seem to be the cubics with control points $\{\mathbf{P}, \mathbf{C_1}, \mathbf{C_2}, \mathbf{Q}\} = \{(1,0), (1,a), (a,1), (0,1)\}$ where $a \in [0,1]$ and hence we can draw these cubics for some values of a, just to see how they look like.



After few tries we can discover that a good approximation can be reach with $a = 0.55$, so we have some hints for a solution; but to better understand the problem we need to find the implicit form of a Bezier curve, at least for those ones that we are studying. Let's rewrite the parametric form:

$$\mathcal{C}_a = \begin{cases} x = & (1-t)^3 + 3(1-t)^2 t + 3(1-t)t^2 a \\ \\ y = & 3(1-t)^2 ta + 3(1-t)t^2 + t^3 \end{cases}$$

First let's note that for $t = 1/2$ we have

```
subst([(1-t)^3+3*(1-t)^2*t+3*(1-t)*t^2*a,
      3*(1-t)^2*t*a+3*(1-t)*t^2+t^3],t,1/2)

 = [3/8*a + 1/2, 3/8*a + 1/2]
```

i.e. the middle points of the curves are along the line $y = x$.

Next let's introduce the *hybrid* form

$$\begin{cases} -X + (1-t)^3 + 3(1-t)^2 t + 3(1-t)t^2 a \\ \\ -Y + 3(1-t)^2 ta + 3(1-t)t^2 + t^3 \end{cases}$$

We would like, if possible, to *eliminate* the parameter t from the hybrid form and obtain a polynomial $P_a[X,Y]$ indexed by a. The PARI-GP function `polresultant` is what we need:

```
PaXY=polresultant(
  -X + (1-t)^3+3*(1-t)^2*t+3*(1-t)*t^2*a,
  -Y + 3*(1-t)^2*t*a+3*(1-t)*t^2+t^3,t);
```

give us
$P_a[X,Y] = (27a^3 - 54a^2 + 36a - 8)X^3 + ((81Y - 108)a^3 + (-162Y + 207)a^2 + (108Y - 126)a + (-24Y + 24))X^2 + ((-81Y + 81)a^4 + (81Y^2 - 162Y + 81)a^3 + (-162Y^2 + 414Y - 252)a^2 + (108Y^2 - 252Y + 144)a + (-24Y^2 + 48Y - 24))X + ((81Y - 81)a^4 + (27Y^3 - 108Y^2 + 81Y)a^3 + (-54Y^3 + 207Y^2 - 252Y + 99)a^2 + (36Y^3 - 126Y^2 + 144Y - 54)a + (-8Y^3 + 24Y^2 - 24Y + 8))$

Now, if $P_a[X,Y]$ has degree 3 then it cannot be identical to $X^2 + Y^2 - 1$, and each approximation

will never be perfect (always remember that $k = \mathbb{Q}$ or $k = \mathbb{R}$ and that they have infinite elements). If we calculate the roots of the coefficient of X^3 with

```
factor(27*a^3 - 54*a^2 + 36*a - 8)
```

we have

```
[3*a - 2 3]
```

i.e $a_0 = 2/3$ is a root of order 3. So, maybe for $a_0 = 2/3$ we have a chance that $P_{a_0}[X, Y]$ has degree 2. Let's substitute a with 2/3 in $P_a[X, Y]$:

```
subst(PaXY,a,2/3)
```

The result is:

```
0.
```

This suggests that the previous parametrization is not valid if $a = 2/3$: in fact if we calculate `polresultant` with $a = 2/3$ we have

```
P23XY=polresultant(
   -X + (1-t)^3+3*(1-t)^2*t+3*(1-t)*t^2*(2/3),
   -Y + 3*(1-t)^2*t*(2/3)+3*(1-t)*t^2+t^3,t)
= X^2 + (-2*Y + 2)*X + (Y^2 + 2*Y - 3)
```

We should be not surprised: after all the curve with $a = 2/3$ is showed in the picture so we already know that *it is* a curve; only it has *order 2*:

```
[(1-t)^3+3*(1-t)^2*t+3*(1-t)*t^2*(2/3),
  3*(1-t)^2*t*(2/3)+3*(1-t)*t^2+t^3]

= [-t^2 + 1, -t^2 + 2*t]
```

i.e. $x = 1 - t^2$, $y = 2t - t^2$, $t \in$ [0,1] (actually, the picture shows that a cubic Bezier curve can be a segment ($a = 0$), a curve of second order ($a = 2/3$) or a curve of third order). But anyway again $X^2+(-2*Y+2)*X+(Y^2+2*Y-3)-X^2-Y^2+1$ is not the zero polynomial.

The conclusion then seems to be: $P_a[X, Y], a \in$ [0,1] *can never be equal to* $X^2 + Y^2 - 1$, or *with a Bezier curve we can never draw a circumference*. But we must consider that it depends from the result of the parametrization, and we have not proved that this is unique — or that one from `polresultant` is exact.

But let's go on. We have already seen that for $t = 1/2$ each curve has $x = y$; another idea from the picture comes if we consider the intersection of the circumference with the line $y = x$, i.e. $(x, y) = (1/\sqrt{2},1/\sqrt{2})$, and *impose* that our Bezier curves pass through $(1/\sqrt{2},1/\sqrt{2})$. This is because the curves in $\mathbf{P}$ and $\mathbf{Q}$ are already a good approximation of the circumference (they have similar derivatives) and they evolve smoothly, so if we choose the middle point as another constrain things *maybe* are better. Theoretically we should go from $P_a[X, Y]$ to $P_a(x, y) = 0$, substitute y with x, substitute x with $1/\sqrt{2}$ and calculate the roots of $P_a(1/\sqrt{2},1/\sqrt{2}) = 0$, but we can collect all these steps with

```
factor(subst(subst(PaXY,Y,X),X,1/sqrt(2)))
```

that gives

```
[a + 0.47140452079104846163831395 38 1]
[a + 0.47140452079101496042294785297 1]
[a - 0.55228474983079339840225163 09 1]
[a - 0.66666666666666666666666666685 1]
```

and we find that, for $a \in$ [0,1] and $a \neq 2/3$,

$a_0^* = 0.55228474983079339840225163 09$

is our only solution.

Now, for $t = 1/2$ the curves P_{a0} must pass in $x = y = 1/\sqrt{2}$, hence, from the parametric form,

$$\frac{3}{8}a + \frac{1}{2} = \frac{1}{\sqrt{2}}$$

which have a unique solution a_0

$$a_0 = 4\frac{\sqrt{2} - 1}{3}$$

It has a nice look (sometimes it is called the "1-2-3-4" formula): its value is

```
4*(sqrt(2)-1)/3
= 0.55228474983079339840225163 23
```

and it is a root of $P_a(x, y)$:

```
subst(subst(subst(PaXY,Y,X),X,1/sqrt(2)),
      a,4*(sqrt(2)-1)/3)
= 1.186446801E-27
```

Hence, whether we follow the implicit way or the parametric way the conclusion is the same: there is a unique Bezier curve that is an optimal approximation of a circumference. The parametric way give us a nice formula for the curve, the implicit way tell us that there are no Bezier curves that are circumferences.

Now, the last step. We would like to have an estimation of the error, i.e. how much our optimal cubic differs from the circumference. We can consider the distance between two points $\mathbf{P_1} = \{y = tx \cap \mathcal{C}\}$ and $\mathbf{P_2} = \{y = tx \cap P_{a_0}(x, y)\}$ and take the max of this distance as an indicator of the error. Recalling that $\mathcal{C}$ has radius 1, it's easy to show that $\vec{\mathbf{P_1}\mathbf{P_2}} = \vec{\mathbf{P_2}} - \vec{\mathbf{P_1}} = \vec{\mathbf{P_2}} - 1$ and the relative percentage error is

$$e_r = \frac{\vec{\mathbf{P_1}\mathbf{P_2}}}{\vec{\mathbf{P_1}}} 100 = 100\left(\vec{\mathbf{P_2}} - 1\right) \qquad (1)$$

The following code print the error e_r vs. the angle ϕ, where $y = x\tan(\phi)$, $0° \leq \phi < 90°$, and the $\max(e_r)$:

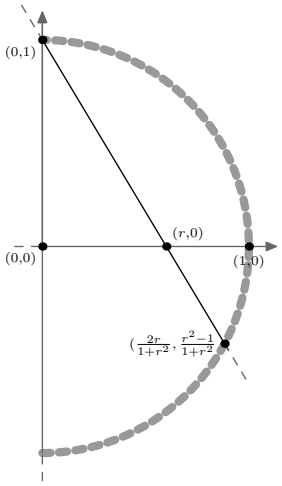
```
B=subst(subst(PaXY,a,
    4*(sqrt(2)-1)/3,),Y,m*X);
MAX=-100;
forstep(f=0,89,1,mf=tan(Pi/180*f);
  roots=polroots(subst(B,m,mf));
  forstep(i=1,3,1,
   xf=real(roots[i]);
  iif=imag(roots[i]);
  if(xf<=1.0 && xf>=0.0 && iif <1e-12,
    yf=mf*xf;  Pf=sqrt(xf^2+yf^2);
    if(MAX<(Pf-1)*100,MAX=(Pf-1)*100,MAX;);
    print(f,",",  (Pf-1)*100,",",MAX))))
```

The error is $e_r = 0.0272\%$, and confirms that a_0 is a good choice.

3.4.1 Some notes

A different parametrization of the circumference is given by the intersection of the line that join (0,1) and (*r*,0), $y = -\frac{1}{r}x + 1$, with $x^2 + y^2 - 1 = 0$:

$$\mathcal{C} = \begin{cases} x = & \dfrac{2r}{1+r^2} \\ \\ y = & \dfrac{r^2-1}{1+r^2} \end{cases}$$



Both x and y are rational if r is rational, so this shows that there are *infinite rational points on a circumference*.

Another question is when $\sqrt{1+t^2}$ is rational if t is rational. Let $t = t_1/t_2, t_1 \geq 0, t_2 \geq 0$: then

$$\sqrt{1+\left(\frac{t_1}{t_2}\right)^2} = \frac{n_1}{n_2} \iff \sqrt{t_1^2+t_2^2} = t_2\frac{n_1}{n_2}$$

i.e. the problem is to find t_1 and t_2 so that $\sqrt{t_1^2+t_2^2} = t_3$ is an integer, or $t_1^2 + t_2^2 = t_3^2$ with t_3 integer. There is an easy "3-4-5" formula to start with: $3^2 + 4^2 = 5^2$. If we rewrite it as $4^2 = 5^2 - 3^2$ or $4 \cdot 4 = (5-3)(5+3) = 2 * 8$, we see that 4 divides $(5+3)$, so $t_1 t_1 = (t_3 - t_2)(t_3 + t_2)$ and t_1 divides $(t_3 + t_2)$ is true at least for the triple $(t_1, t_2, t_3) = (3,4,5)$. Let's then *suppose* that t_1 divides (t_3+t_2), i.e. $t_3+t_2 = k_1 t_1$: with this condition we have

$$\begin{cases} t_1 = & (t_3 - t_2)k_1 \\ \\ k_1 t_1 = & (t_3 + t_2) \end{cases}$$

and hence

$$t_3 = \frac{t_1}{2}\frac{k_1^2+1}{k_1}$$

that is an integer if $t_1 = l_1 k_1$ and $l_1(k_1^2+1)$ is even, with l_1 integer.

The last note is about the field k. We have used $k = \mathbb{Q}$ or $k = \mathbb{R}$ were the distinction between $P[X,Y]$ and $P(x,y)$ seem to be artificial, but there are fields k where distinction matter. For example the classes of residues modulo p with p prime is a field $\mathbb{F}_p$ with the usual operation "$+, \times$" as in $\mathbb{Q}$ or $\mathbb{R}$. Let's consider $\mathbb{F}_5$: denote each class of residue with $\{\overline{0}, \overline{1}, \overline{2}, \overline{3}, \overline{4}\}$ (hence $\overline{4}\cdot\overline{3} = \overline{12} = \overline{2}$ and $\overline{4} + \overline{3} = \overline{7} = \overline{2}$).
The polynomial $P[X] = X(X-\overline{1})(X-\overline{2})(X-\overline{3})(X-\overline{4}) = X^5 + \overline{4}X$ it's not the zero polynomial of $\mathbb{F}_5[X]$, but $P(x) = \overline{0}$ for every element in $\mathbb{F}_5$.

4 Conclusion

One of the main advantages of ConTEXt MkIV is the clear separation between Lua code and TEX code, and in this case it's a very useful thing that we can import a pari-lua script into ConTEXt MkIV without too much work to adapt it to the ConTEXt MkIV machinery — i.e. we have an high degree of code reuse. PARI/GP has also a nice TEX formatter, even if in some situations things are a bit crude.

On the other side, solving numerical problems *always* requires some amount of theoretical analysis *before* doing the computation, as in the case of `solve` — in some circumstances PARI/GP abruptly aborts if it finds an error. Some computations can require a long time to finish, and given that ConTEXt MkIV is a multi pass system a caching mechanism should be provided to solve these situations.

Numeric results *can* (but they shouldn't) depend on the compiler and/or platform, but it seems that from this point of view that PARI/GP is really platform-independent.

References

URL `http://pari.math.u-bordeaux.fr`.

URL `http://sagemath.org`.

URL `http://www.nongnu.org/gsl-shell`.

URL `http://www.inf.puc-rio.br/~roberto/pil2`.

URL `http://swig.org`.

URL `http://www.mingw.org`.

URL `http://www.lilypond.org/gub`.

URL `http://www.luatex.org/svn/trunk/manual/luatexref-t.pdf`.

URL `http://en.wikipedia.org/wiki/Leibniz_formula_for_pi`.

▷ Luigi Scarso
`luigi.scarso at gmail dot com`

Eventi e novità

TUG 2011

Contrariamente agli annunci precedenti, TUG 2011 non si terrà in Egitto. La situazione politica e civile egiziana ha consigliato di cambiare la sede in cui tenere il convegno annuale.

Il convegno 2011, dal tema "TEX nell'era dell'e-Book" si terrà dunque a Trivandrum, India, con River Valley Technologies tra gli sponsor principali.

Dal 19 al 21 ottobre gli autori accettati presenteranno i loro articoli. Come di consueto, gli atti saranno disponibili il un secondo tempo, visto che la scadenza per la consegna degli articoli è il 31 ottobre.

Gli interessati a seguire il convegno (o gli autori) potranno arrivare direttamente all'aeroporto di Trivandrum, ben collegato con voli facenti scalo in Medio Oriente (Dubai, Abu Dhabi, Qatar).

5th ConTEXt User Meeting 2012

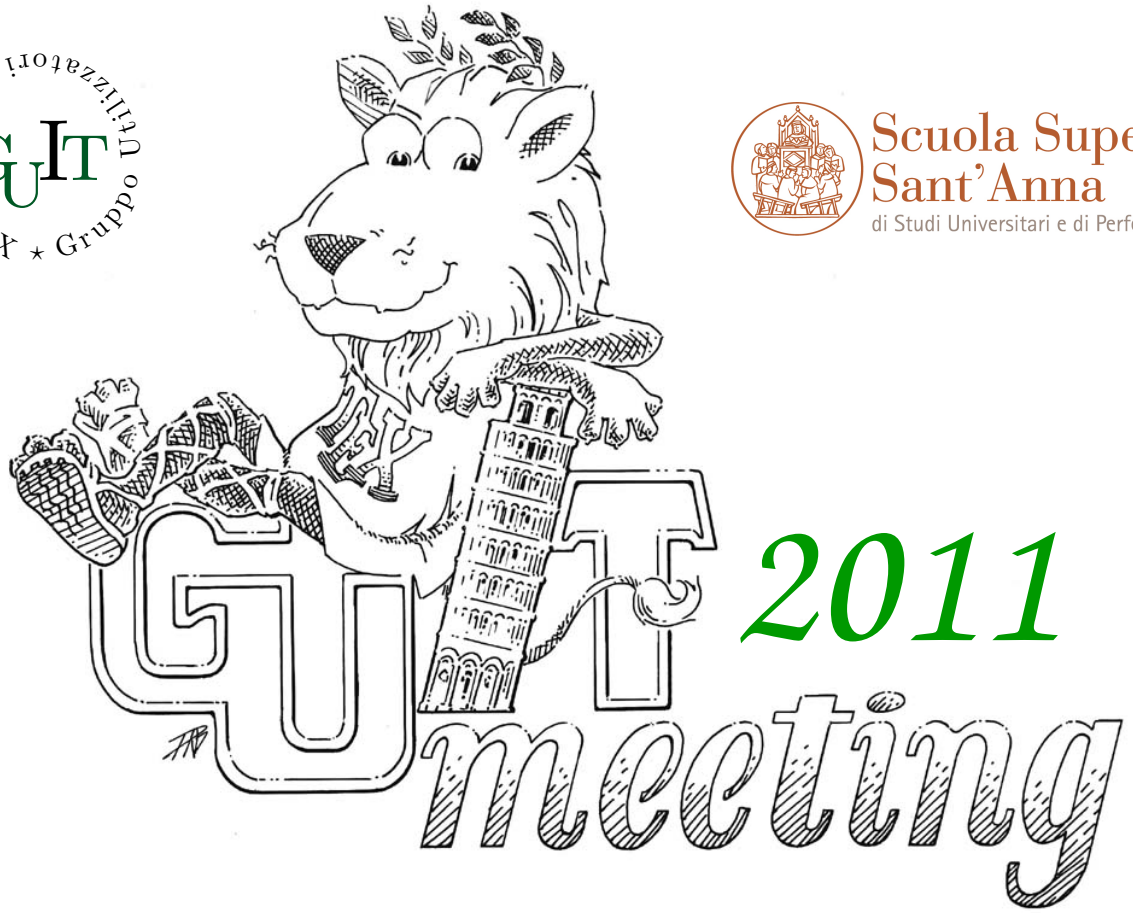
Dal 19 al 24 settembre 2011 a Bessenge, Belgio si svolgerà il quinto meeting degli utenti ConTEXt.

Nel meeting gli autori presentano tecniche, idee, esperienze e soluzioni sia nell'uso di ConTEXt che di LuaTEX. Tutti gli interventi saranno seguiti da tutorial sui programmi citati.

6th ConTEXt User Meeting 2012

Nel 2012, sempre a ottobre, dall'8 al 12, si terrà a Breskens, Olanda, il sesto meeting degli utenti ConTEXt. L'organizzazione sarà curata dall'NTG e DANTE.

Questa rivista è stata stampata
presso Logo S.r.l. Servizi di Stampa Digitale, Borgoricco (PD)
su carta ecosostenibile Vision Trend White
prodotta da Steinbeis Temming Papier GmbH & Co.



Ottavo convegno nazionale su TEX, LATEX e tipografia digitale

Pisa, 15 ottobre 2011

Aula Magna, Scuola Superiore Sant'Anna

Call for Paper

Sabato 15 ottobre 2011 presso l'Aula Magna della Scuola Superiore Sant'Anna di Pisa si terrà l'ottavo Convegno annuale su TEX, LATEX e tipografia digitale organizzato dal Gruppo Utilizzatori Italiani di TEX. Il Convegno sarà un momento di ritrovo e di confronto per la comunità LATEX italiana, tramite una serie di interventi atti sia a contribuire all'arricchimento sia a supportarne lo sviluppo.
Maggiori informazioni sul Convegno e sulle modalità di presentazione degli interventi sono disponibili all'indirizzo:

`http://www.guit.sssup.it/guitmeeting/2011/`

ArsTEXnica

Numero 11, Aprile 2011

9 771828 236001 10011