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:

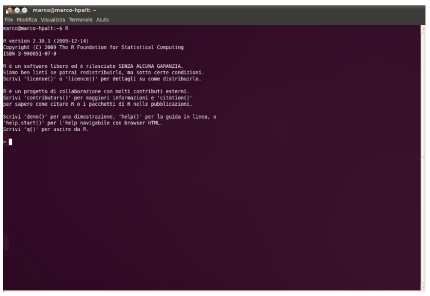

Figura 1: R Console in ambiente Linux

- 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