MANUALE GNUPLOT

A cura di Giuseppe Ciaburro

http://www.ciaburro.it

info@ciaburro.it

Indice

Manuale di Gnuplot

Figura 1: Giuseppe Ciaburro

0.1 Introduzione

GNUPLOT è un programma che permette di tracciare grafici bi e tridimensionali. La caratteristica principale di tale programma è che ci permette di ottenenere dei grafici da dati formattati con molta semplicità.Gnuplot è gratuito si trova su tutte le distribuzioni linux, ma è possibile utilizzarlo anche sulle piattaforme windows.

• Installazione

-Linux E' possibile trovare il file .rpm che ci permette tramite i programmi presenti nelle versioni di linux di installarlo facilmente. Provate al seguente url:

`http://krone.physik.unizh.ch/~droz/RPMS/i386/`

-Windows Basta scompattare il file .zip in una directory ed è fatto.Il file si trova:

`http://www.wavecomputers.net/~lane/Math/GNUPlot/`

0.2 Avvio

L'avvio del programma si ottiene digitando "gnuplot" al prompt di linux oppure cliccando sull'icona relativa per la versione a 32bit;a questo punto si apre la seguente finestra(nel caso della versione a 32 bit):

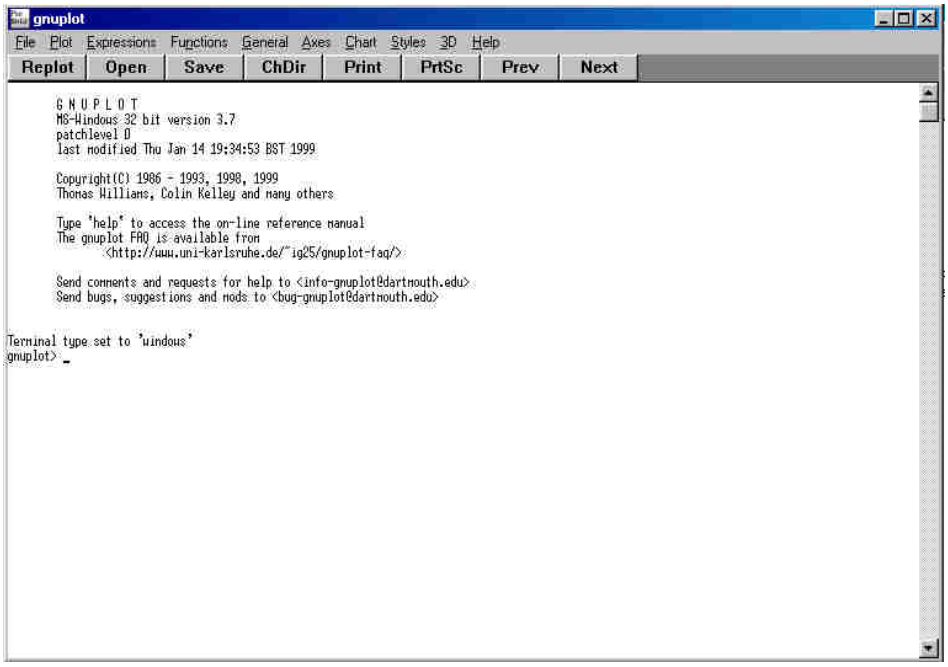

Figura 2: Finestra di Avvio

A questo punto vedrete lampeggiare un segno _ sulla linea di comando e per effettuere delle operazioni dovrete digitare i comandi nella versione per linux, mentre potrete usufruire di una finestra per i comandi nella versione a 32 bit.

- Commenti

Commenti sono sostenuti come segue: se il simbolo # appare in una linea GNUPLOT ignorerà il resto della linea. Non avrà questo effetto se racchiuso in virgolette, in numeri (incluso numeri complessi), in sostituzioni del comando, etc.

- Espressioni

In generale, qualsiasi espressione matematica accettata da C, FORTRAN, Pascal o BASIC è valida. La precedenza di questi operatori è determinata dalle specificazioni del linguaggio di programmazione C. Spazi bianchi (gli spazi e tab) sono ignorati nelle espressioni.

Costanti complesse possono essere espresse come $\{< real >, < im >\}$, dove e devono essere costanti numeriche. Per esempio, $\{3, 2\}$ rappresenta 3 + 2i.

0.3 Comandi

In questa sezione saranno elencati tutti i comandi disponibili in gnuplot in ordine alfabetico. E' da notare che in molti casi possono essere usate delle abbreviazioni per i comandi, come ad esempio:

```
"p f(x) w l" al posto di "plot f(x) with lines"
```

Bisogna fare molta attenzione a tali abbreviazioni per non incorrere in ambiguità. L'elenco dei comandi è il seguente:

cd
call
clear
exit
help
if
load
pause
plot
print
pwd
quit
replot
reread
reset
save
set-show
splot

- CD

Il comando cd cambia la directory di lavoro.

Sintassi:

cd " "

Il nome della directory deve essere incluso tra virgolette.

Esempi:

```
cd 'subdir'
cd ".."
```

• CALL

Il comando call è identico al comando load con una eccezione,si possono aggiungere 10 parametri supplementari.Nella file che dovrà essere chiamato i parametri saranno identificati con il segno del dollaro $ seguito da un numero compreso tra 0-9 .

```
Syntax:
      call "<input-file>" <parameter-0> <parm-1> ... <parm-9>
```

Il nome del file di input deve essere racchiuso tra virgolette,ed è consigliato che anche i parametri siano racchiusi tra virgolette.

Esempio:

Se il file 'calltest.gp' contiene la linea:

```
print "p0=$0 p1=$1 p2=$2 p3=$3 p4=$4 p5=$5 p6=$6 p7=x$7x"
```

digitare il comando:

```
call 'calltest.gp' "abcd" 1.2 + "'quoted'" -- "$2"
```

che produrrà al display:

```
p0=abcd p1=1.2 p2=+ p3='quoted' p4=- p5=- p6=$2 p7=xx
```

Cioè `$0=abcd $1=1.2` etc.

• CLEAR

Il comando clear annulla lo schermo corrente o l'output device così come specificato dall'output . Questo di solito genera un formfeed su apparecchiature di hardcopy. Usare set terminal per settare il tipo di device.

Esempio:

```
set multiplot
plot sin(x)

set origin 0.5,0.5
set size 0.4,0.4
clear
plot cos(x)
set nomultiplot
```

In questo modo tutti i settaggi effettuati nella costruzione del primo grafico saranno stati annullati e si procederà alla costruzione di un nuovo grafico.

• EXIT

I comandi exit e quit e il carattere di Fine-file permettono di uscire da GNUPLOT. Tutti questi comandi annulleranno l'output device (come il comando clear) prima di uscire.

•HELP

Il comando help apre la finestra di aiuto on-line.Per specificare le informazioni sulle quali è richiesto l'aiuto usare la sintassi seguente:

```
help {<topic>}
```

Se `<topic>` non è specificato un breve messaggio è visualizzatoe si apre la finestra principale di aiuto.

•IF

Il comando if predispone che il comando successivo sia eseguito a patto che si verifichi una data condizione:

Sintassi:

```
if (<condition>) <command-line>
```

la condizione sarà valutata.Se risulterà vera allora il comando sarà eseguito, altrimenti sarà ignorato. Esempio:

```
pi=3

if (pi!=acos(-1)) print "?Fixing pi!"; pi=acos(-1); print pi
```

questo produrrà al video:

```
?Fixing pi!
    3.14159265358979
```

ma se

```
if (1==2) print "Never see this"; print "Or this either"
```

questo non produrrà nulla.

•LOAD

Il comando load esegue tutte le linee del fiel di input specificato.I file creati con il comando SAVE potranno in seguito eseguiti con il comando LOAD. Il comando LOAD deve essere l'ultimo comando

in una linea in cui compaiono più comandi.

```
Syntax:
      load "<input-file>"
```

Il nome del file di input deve essere racchiuso tra virgolette.Esempi:

```
load 'lavoro.gnu'
load "funzione.dat"
```

•PAUSE

Il comando PAUSE fornisce a video qualsiasi testo sia associato al comando e metterà il sistema in pausa per un periodo specificato oppure fino a quando il tasto return non sarà pigiato.

```
Syntax:
     pause <time> {"<string>"}
```

Dove `<time>` può essere un intero o una espressione. Scegliendo -1 si metterà in pausa fino a quando il tasto return non sarà pigiato,0 non produrrà pausa, un qualsiasi numero intero positivo N metterà il sistema in pausa per N secondi.

Esempi:

```
pause -1    # Aspetta fino a quando il tasto return è pigiato
pause 3     # asèetta tre secondi
pause -1  "pigia return per continuare"
```

•PLOT

Il comando plot ci permette di tracciare grafici bidimensionali, da funzioni oppure da file di dati.

Sintassi:

```
plot {<ranges>}
     {<function> | {"<datafile>" {datafile-modifiers}}}
     {axes <axes>} {<title-spec>} {with <style>}
     {, {definitions,} <function> ...}
```

dove il nome del file di dati deve essere racchiuso tra virgolette.

Esempi:

```
plot sin(x)
plot f(x) = sin(x*a), a = .2, f(x), a = .4, f(x)
plot [t=1:10] [-pi:pi*2] tan(t), \
     "data.1" using (tan($2)):($3/$4) smooth csplines \
              axes x1y2 notitle with lines 5
```

•PRINT

Il comando print stampa il valore della variabile expression sullo schermo.

```
Syntax:
      print <expression> {, <expression>, ...}
```

•PWD

Il comando PWD stampa sullo schermo il nome della directory di lavoro con il relativo cammino.

•QUIT

Il comando QUIT serve per abbandonare una sessione di lavoro, tale comando azzera l'output device prima di uscire.

•REPLOT

Il comando REPLOT senza argomenti ripete l'ultimo comando plot o splot. Mentre gli argomenti specificati dopo il comando replot verrando inclusi nel nuovo tracciato.

•REREAD

Il comando REREAD effettua la rilettura di un file, quindi esegue il comando load.

Esempi:

Supponiamo che il file "looper" contenga i comandi

```
a=a+1
plot sin(x*a)
pause -1
if(a<5) reread
```

e dal prompt di gnuplot di digitino i seguenti comandi

```
a=0
load 'looper'
```

Il risultato sarà il tracciamento di quattro grafici.

•RESET

Il comando RESET determina il ritorno ai valori di default di tutte le opzioni imposte con il comando SET.

•SAVE

Il comando SAVE salva le seguenti caratteristiche:

-funzioni definite dall'utente -variabili -set options -plot

Sintassi:

```
save  {<option>} '<filename>'
```

dove option è una funzione, una variabile oppure un set.

Esempi:

```
save 'lavoro.gnu'
save functions 'funzione.dat'
save var 'variabile.dat'
save set 'opzione.dat'
```

•SET-SHOW

Tale comando serve per settare tutte le opzione necessarie per il tracciamento a video del grafico. Assi, angoli ,format,bordi etc.

•SPLOT

Il comando splot serve per tracciare grafici tridimensionali.

```
Syntax:
        splot {<ranges>}
        <function> | "<datafile>" {datafile-modifiers}}
        {<title-spec>} {with <style>}
        {, {definitions,} <function> ...}
```

dove il nome del file di dati deve essere racchiuso tra virgolette.

0.4 Line-editing

Le versioni di GNUPLOT per Unix, Atari, VMS, MS-DOS e OS/2 supportano l'editing da linea di comando. Inoltre è presente un meccanismo di memorizzazione di tutti i comandi editati, e permette che comandi precedenti siano compilati, e rieseguiti. Dopo che la linea del comando è stata compilata, un newline registrerà malgrado tutto la linea intera da dove il cursore è posizionato.

I comandi di editing sono come segue:

Line-editing:

```
^B trasporta indietro un singolo carattere.
^F si muove in avanti di un singolo carattere.
^A si muove all'inizio della linea.
^E si muove alla fine della linea.
^H e Del cancellano il carattere precedente.
^D cancella il carattere corrente.
^K cancella dalla posizione corrente alla fine di linea.
^L,^R ridisegna la linea nel caso sia rovinata.
^U cancella la linea intera.
^W cancella l'ultima parola.
```

Storia: (per storia si intende la sequenza di comandi editati al prompt di gnuplot dall'inizio della sessione).

```
^P si muove indietro attraverso la storia
^N si muove in avanti attraverso la storia.
```

Su PC l'uso di un programma di TSR come DOSEDIT o CED può essere utile per l'editing di linea. Le chiavi seguenti possono essere usate su PC se readline è usato:

```
Freccia sinistra   -  come  ^B.
Freccia destra     -  come  ^F.
Ctrl + Left  Arrow -  come  ^A.
Ctrl + Right Arrow -  come  ^E.
Up    Arrow        -  come  ^P.
Down  Arrow        -  come  ^N.
```

0.5 Funzioni

Le funzioni in GNUPLOT sono le funzioni corrispondenti della biblioteca matematica di Unix, a patto che tutte le funzioni accettino numeri interi,reali, e complessi. La funzione sgn è prevista come in Basic.

abs
acos
arg
asin
atan
besj0
besj1
besy0
besy1
ceil
cos
cosh
erf
erfc
exp
floor
gamma
ibeta
inverf
igamma
imag
invnorm
int
lgamma
log
log10
norm
rand
real
sgn
sin
sinh
sqrt
tan
tanh

- abs

La funzione abs fornisce il valore assoluto dell'argomento. Il valore ritornato è dello stesso tipo dell'argomento.
Per argomenti complessi, abs(x) è definito come la lunghezza di x nel piano complesso [i.e., sqrt(real(x)**2 + imag(x)**2)].

- acos

La funzione acos calcola l'inverso del coseno del suo argomento; acos fornisce il risultato in radianti.

• arg

arg calcola la fase di un numero complesso, in radianti.

•asin

La funzione asin calcola l'inverso del seno del suo argomento; asin fornisce il risultato in radianti.

• atan

La funzione atan calcola l'arco della tangente (inverso tangente) del suo argomento. atan fornisce il risultato in radianti.

•besj0

The besj0 function returns the j0th Bessel function of its argument. besj0 expects its argument to be in radians.

• besj1

The besj1 function returns the j1st Bessel function of its argument. besj1 expects its argument to be in radians.

• besy0

The besy0 function returns the y0th Bessel function of its argument. besy0 expects its argument to be in radians.

• besy1

The besy1 function returns the y1st Bessel function of its argument. besy1 expects its argument to be in radians.

• ceil

La funzione ceil calcola il numero intero più piccolo ¿= del suo argomento. Per numeri complessi, ceil fornisce il numero intero più piccolo, rispetto alla parte reale del suo argomento.

• cos

La funzione cos calcola il coseno del suo argomento; cos si aspetta che il suo argomento sia in radianti.

• cosh

La funzione cosh calcola il coseno iperbolico del suo argomento; cosh si aspetta che il suo argomento sia in radianti.

• erf

La funzione erf valuta la funzione errore della parte reale del suo argomento. Se l'argomento è un valore complesso, il componente immaginario è ignorato.

• erfc

The erfc function returns 1.0 - the error function of the real part of its argument. Se l'argomento è un valore complesso, il componente immaginario è ignorato.

• exp

La funzione exp valuta l'esponenziale del suo argomento .

• floor

Valuta il più grande numero intero non più grande del suo argomento. Per numeri complessi, valuta il più grande numero intero non più grande della parte reale del suo argomento.

• gamma

Valuta la funzione gamma della parte reale del suo argomento. Per numero intero n, gamma(n+1) = n! . Se l'argomento è un valore complesso, il componente immaginario è ignorato.

• ibeta

La funzione ibetavaluta la funzione beta incompleta delle parti reali dei suoi argomenti;
`p, q> 0 e x in [0:1]`
Se gli argomenti sono complessi, i componenti immaginari sono ignorati.

• inverf

La funzione inverf valuta l'inverso della funzione errore della parte reale del suo argomento.

• igamma

La funzione igamma valuta la funzione gamma incompleta delle parti reali dei suoi argomenti;
`a > 0 e x >= 0`
Se gli argomenti sono complessi, i componenti immaginari sono ignorati.

• imag

La funzione imag trasforma la parte immaginaria del suo argomento come un numero reale.

• invnorm

La funzione invnorm fornisce l'inverso della funzione di distribuzione normale della parte reale del suo argomento.

• int

La funzione int fornisce la parte intera del suo argomento, troncata a zero.

• lgamma

La funzione lgamma valuta il logaritmo naturale della funzione gamma della parte reale del suo argomento. Se l'argomento è un valore complesso, il componente immaginario è ignorato.

• log

Valuta il logaritmo naturale (in base e) del suo argomento.

• log10

La funzione log10 valuta il logaritmo (base 10) del suo argomento.

• norm

Valuta la funzione della distribuzione normale (o Gaussian) dellaparte reale del suo argomento.

• rand

La funzione rand fornisce un numero casuale nell'intervallo [0:1] usando la parte reale del suo argomento come un iniziatore. Se l'iniziatore è ¡0 la sequenza è reinizializzata. Se l'argomento è un valore complesso, il componente immaginario è ignorato.

• real

Valuta la parte reale del suo argomento.

• sgn

La funzione sgn fornisce 1 se il suo argomento è positivo, -1 se il suo argomento è negativo, e 0 se il suo argomento è 0. Se l'argomento è un valore complesso, il componente immaginario è ignorato.

• sin

Valuta il seno del suo argomento; sin si aspetta che il suo argomento sia in radianti.

• sinh

La funzione sinh valuta il seno iperbolico del suo argomento; sinh si aspetta che il suo argomento sia in radianti.

• sqrt

La funzione sqrt calcola la radice quadrata del suo argomento.

• tan

Valuta la tangente del suo argomento;tan si aspetta il suo argomento in radianti.

• tanh

La funzione tanh valuta la tangente iperbolica del suo argomento; tanh si aspetta che il suo argomento sia in radianti.

0.6 Esempio

Sarà molto chiaro l'utilizzo del programma gnuplot per relizzare un diagramma ,da dati salvati su un file, grazie all'aiuto dell'esempio seguente.

Supponiamo di integrare l'equazione della diffusione monodimensionale attraverso il programma diffusion (presente nella sezione programmi di questo sito)scritto in fortran 90, il quale ci salva i dati in un file. Vediamo come utilizzare gnuplot per tracciare un diagramma da tali dati.

1)Apriamo gnuplot;

2)vogliamo tracciare un digramma 2D quindi il comando da usare sarà PLOT

```
gnuplot > plot
```

quindi dobbiamo caricare il file 'concCh4.dat' (ricordiamo le virgolette e clicca sul file per analizzare la forma). Naturalmente se il file non risiede nella directory da cui abbiamo lanciato gnuplot ci dovremo trasferire in essa con il comando cd su linux attraverso le note finestre sotto windows.

```
gnuplot > plot 'concCh4.dat'
```

a questo punto cliccando su invio abbiamo gia il nostro grafico:

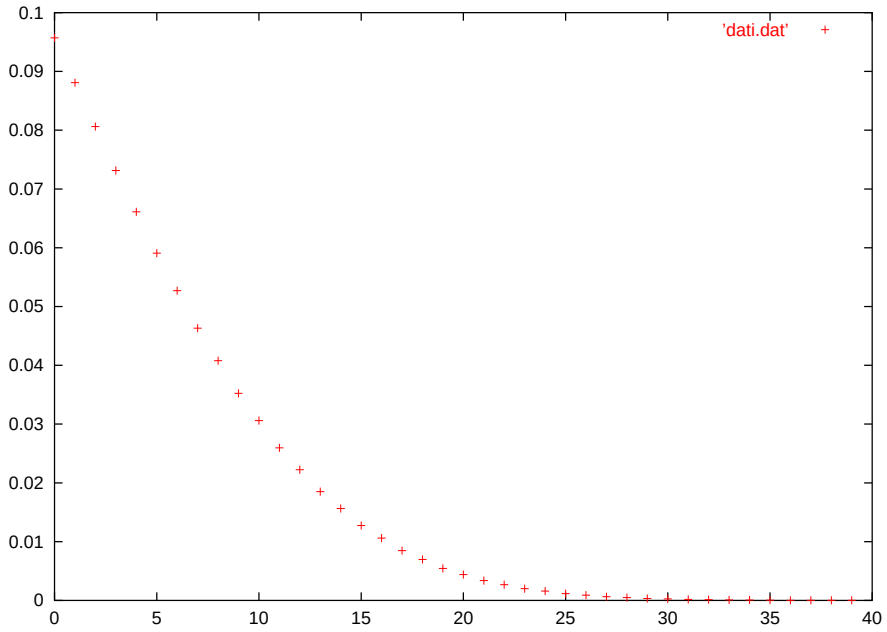

Figura 3: File di dati

Ma cerchiamo di manipolarlo per migliorarlo.

3)Tracciamolo innanzitutto con una linea e non solo con punti

```
gnuplot > plot 'concCh4.dat' with linespoints
```

avremmo anche potuto utilizzare una notazione abbreviata

```
gnuplot > p 'concCh4.dat' w lp
```

Per ottenere lo stesso risultato con l'uso delle finestre nella versione a 32 bit:

Menù Plot $\Longrightarrow$ plot; Menù Plot $\Longrightarrow$ Data filename; Menù Plot $\Longrightarrow$ with stile $\Longrightarrow$ Lines and Points

4)Aggiungiamo delle etichette sugli assi:

Asse x: set xlabel "Numero di celle" 0,0

le ultime due cifre servono per la posizione della etichetta. Nella versione a 32 bit Menù Axes $\Longrightarrow$X label.

Si procede in modo analogo per la y.

set ylabel "Frazione molare del Metano" 0,0

Passiamo ad aggiungere un titolo:

set title "Diffusione del Metano lungo il raggio" 0,0

Nella versione a 32 bit Menù Chart $\Longrightarrow$ Set title.

Abbiamo ottenuto in questo modo il seguente diagramma:

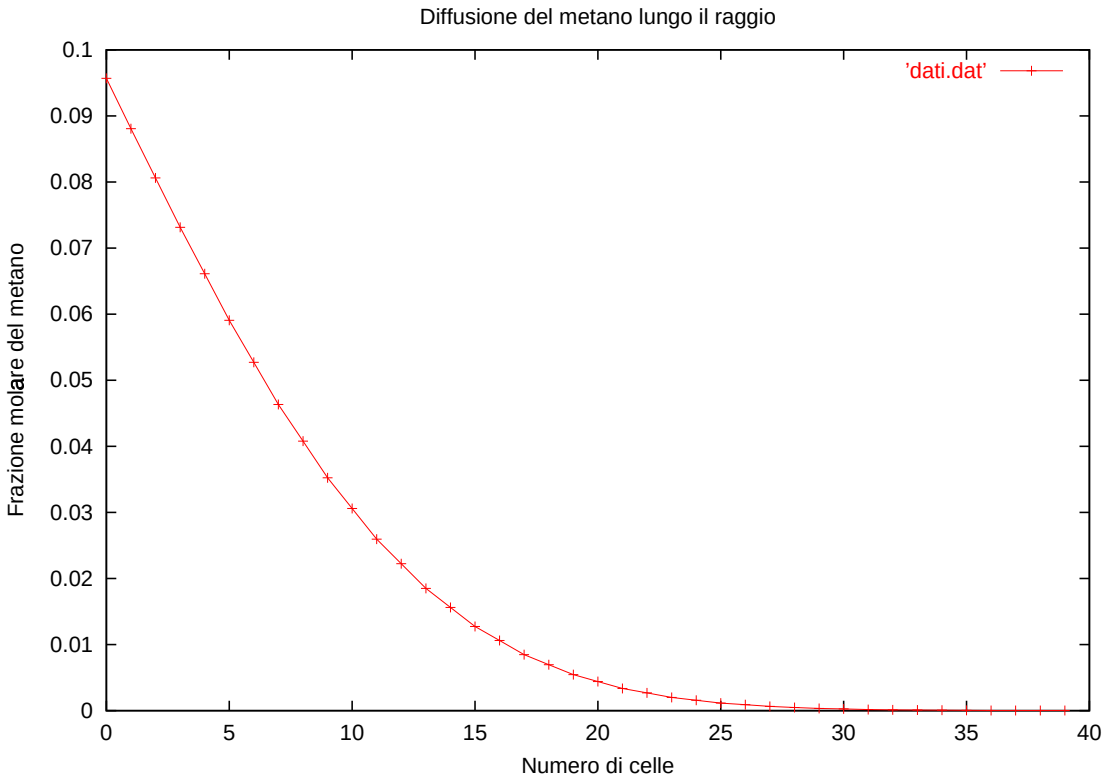

Figura 4: File di dati

5)Infine occorre salvare la figura; per fare questo è necessario decidere il formato.
Consiglio il formato .eps perchè è facilmente inseribile in file latex; per salvare in un file .eps fare quanto segue:

set terminal postscript color
set output "nomefile.eps"
replot

in questo modo il diagramma tracciato sarà salvato nel file nomefile.eps.

0.7 File di dati

Per poter tracciare delle curve da dati ottenuti da altri programmi è necessario salvare i dati in un file con estensione .dat.Ogni curva deve essere individuata da due colonne separate da uno spazio, che rappresentano i valori rispettivamente di ascissa ed ordinata,la curva successiva deve essere espressa in modo analogo al di sotto lasciando tra di esse due righe vuote.Per una più rapida comprensione consideriamo un esempio. Supponiamo le due curve siano rapresentate dal set di dati seguenti:

```
1 3
2 5
3 10
4 12


1 5
2 15
3 16
4 20
```

salviamo tali valori nel file prova.dat. Digitiamo quindi:

```
plot 'prova.dat' with linespoints
```

nell'ipotesi che il file si trovi nella directory corrente. Il risultato è il seguente:

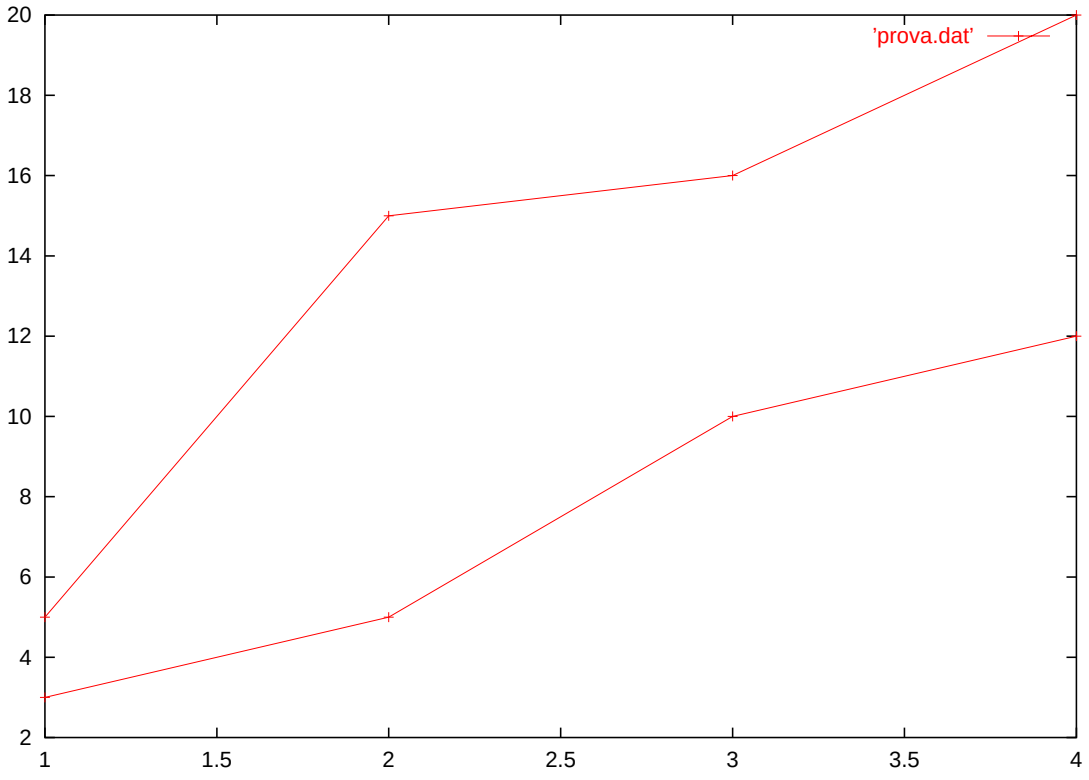

Figura 5: File di dati