Curso de Gnuplot para cómputo científico

Abdiel E. Cáceres González (trad.)

Centro de Investigación y de Estudios Avanzados del IPN

Abstract

Este curso es una traducción de gran parte del curso de Gnuplot escrito por Tim Langlais y se ofrece como recurso para las personas de habla hispana. Se ofrcen ejemplos para graficar en 2D y en 3D usando la terminal de salida de su preferencia, la terminal de LaTeXque genera archivos .TEX , la terminal de PostScript para crear archivos de imagenes .PS y .EPS y la terminal que genera archivos útiles que puede leer MSWord. Finalmente, y esto es algo que no aparece en el curso original, se describe una manera de escribir programas en ANSI C usando compiladores como cc o gcc, para generar las gráficas desde el programa fuente. Se incluye también el código fuente de un programa en C completamente funcional.

Introducción

Gnuplot es un poderoso programa *freeware* para hacer gráficas con datos en 2D y en 3D. Gnuplot puede usarse en muchos ambientes computacionales, incluyendo Linux, IRIX, Solaris, Mac OS X, Windows y DOS. Gnuplot requiere las mínimas capacidades gráficas y puede usarse aún en una terminal de tipo vt100. Tiene una amplia variedad de opciones de salidas para que el usuario pueda usar las gráficas resultantes como lo desee, ya sea para ser visualizados o para incluirlos en sus propios documentos. Este curso está basado en la versión gnuplot 3.8h. Esta versión está displonible para muchos tipos de sistemas operativos en la página oficial de Gnuplot .

http://www.gnuplot.org

Este curso está complementado con el paquete "gnuplot-course.tar.gz", que contiene todos los scripts y datos para los ejemplos, también este mismo documento en formatos PDF y LaTeX.

Email address: acaceres@computacion.cs.cinvestav.mx (Abdiel E. Cáceres González (trad.)).

19 April 2004

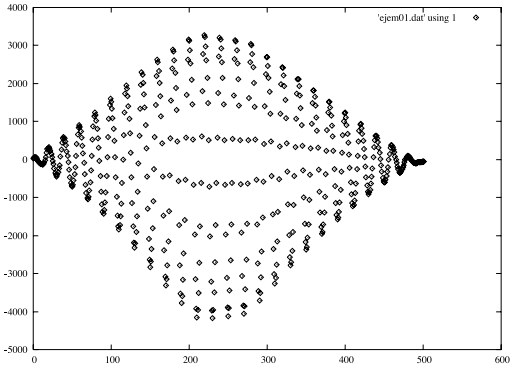


Fig. 1. `plot 'ejem01.dat' using 1`

Nota, el paquete "gnuplot-course.tar.gz" está disponible en mi página web `http://computacion.cs.cinvestav.mx/documents/gnuplot/cursos` con el nombre `gnuplot.tar.gz`.

1 Gráficas básicas en 2D

Para empezar a usar `gnuplot` primero deberá cambiarse de directorio a `cd /gnuplot/data`. En sistemas UNIX, si el programa fue correctamente instalado, simplemente deberá ejecutar:

```
unix% gnuplot
```

Esto hará que `gnuplot` inicie. Podrá ver un mensaje de iniciación que entre otras cosas menciona la versión que se está usando. Si una una versión anterior a la 3.8 algunos comandos pueden no funcionar y necesitará ver el manual especiífico para la versión que esté usando. En el paquete de archivos, podrá observar uno que se llama "ejem01.dat" que es un archivo de texto que tiene números en una sola columna, los números empiezan con la serie:

```
28.062000
52.172000
55.703000
64.281000
43.438000
6.781000
-31.281000
```

Para graficar estos datos, simplemente debe escribir

```
gnuplot> plot 'ejem01.dat' using 1
```

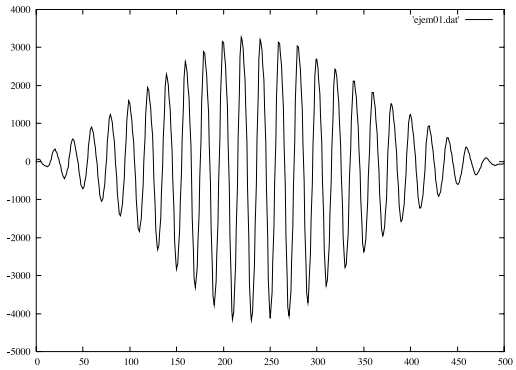


Fig. 2. `plot '../data/ejem01.dat' using 1`

`Gnuplot` pone la escala más conveniente para incluir todos los datos. Si no se especifica de antemano algo, `gnuplot` dibuja poniendo pequeños círculos (puntos). Al cambiarse de directorios en `gnuplot` debe tener en cuenta que el comando de `gnuplot cd` acepta `..` y `/` pero no acepta `~` . Dibijemos de nuevo estos datos pero poniendo más divisiones en el eje X y modificando la escala también en el eje X.

```
gnuplot> set style data lines
gnuplot> set xtics 0,50,1000
gnuplot> set xrange [0:500]
gnuplot> plot '../data/ejem01.dat' using 1
```

Hay muchas opciones para el estilo al dibujar los datos, las opciones son:

`lines`	`points`	`linespoints`	`dots`
`impulses`	`yerrorbars`	`xyerrorbars`	`steps`
`fsteps`	`histeps`	`boxes`	`boxerrorbars`
`boxxyerrorbars`	`vector`	`financebars`	`candlesticks`
`error lines`	`xerrorlines`	`yerrorlines`	`xyerrorlines`

Table 1
Opciones para los estilos del dibujo de los datos. Algunas opciones requieren listas de datos de 2 o más columnas

La sintaxis para escribir el estilo del dibujo de los datos es:

`gnuplot>set style data <style>`

Donde, claro, `<style>` toma alguno de los valores recién listados. El comando `set xtics` tiene 3 argumentos `<start>,<increment>, <end>`, y el comando para modificar el rango en X, `set xrange`, tiene 2 argumentos, `[<start>:<end>]`.

Y si lo que queremos graficar es una colección de datos multicolumnas como:

4.563000	5.078000	-1.719000	
-5.641000	3.148500	0.156000	
-0.781000	3.453500	-1.250000	
6.860000	8.836000	-2.188000	
0.953000	5.586000	0.469000	etc.

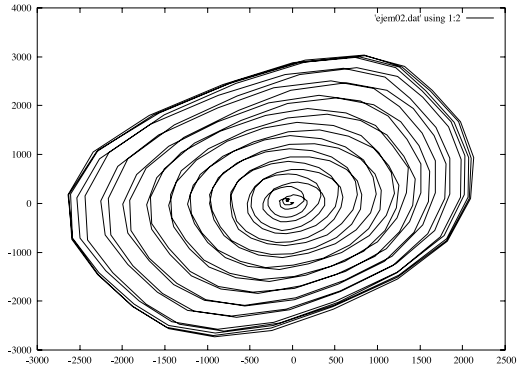


Fig. 3. `plot 'ejem02.dat' using 1:2`

¿Es posible dibujar solamente una columna de esos datos? Claro que sí, dibujaremos solamente la columna primera y segunda de los datos del archivo "ejem02.dat"

```
gnuplot> plot 'ejem02.dat' using 2
```

Antes de entrar este comando, debemos ejecutar esta secuencia de comandos:

```
gnuplot>reset
gnuplot>set style data lines
gnuplot> plot 'ejem02.dat' using 1:2
```

El comando `using` especifíca la columna que será dibujada, si los datos están en 3 columnas, hay muchas posibilidades de dibujar, por ejemplo al usar... `using 3:2` se dibujará la columna 3 en el primer eje X y la columna 2 en el segundo eje Y.

Si necesitáramos hacer operaciones con los datos de una o algunas columnas y esos resultados graficarlos como otra columna, es posible hacerlo en `gnuplot` Bueno, hay varias maneras de hacerlo, una de ellas es hacer las operaciones en otro programa como una hoja de cálculo. Pero, la manera más fácil (si ya se tiene `gnuplot`) es usar esas capacidades que se incorporaron desde la versión `3.7`. Vamos a suponer por ahora que el archivo 'ejem02.dat' contiene los datos de las componenetes x, y, z de la velocidad que es la que queremos dibujar. La velocidad está determinada mediante la expresión

$$v = \sqrt{v_x^2 + v_y^2 + v_z^2}$$

La manera de hacerlo en `gnuplot` es calcular esa velocidad como:

```
gnuplot> plot '3ch.dat' \
 > using (sqrt($1**2+$2**2+$3**2))
```

Note que la línea diagonal \ nos permite escribir en el siguiente renglón, y el comando se ejecutará hasta después de haber pulsado enter. También hay que notar que las columnas se especifican con $1, o $2, o $3 y las potencias con **, así 3^2 se escribe en `gnuplot` como `3**2`. Note que los paréntesis que acotan la expresión son importantes.

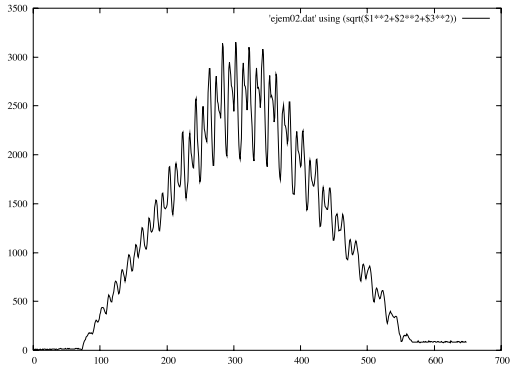


Fig. 4. `gnuplot> plot '3ch.dat' using (sqrt($1**2+$2**2+$3**2))`

2 Operadores, Constantes y Funciones

`Gnuplot` no solo lee datos de archivos, también grafica funciones analíticas. De manera que `gnuplot` ofrece los operadores usuales `+, -, *, /, **`, etc, y funciones como `sin(), cos(), log(), exp()`, etc. Grafiquemos la siguiente función $x^2 \sin x$ como recordará, la exponenciación se escribe en `gnuplot` con `**` :

```
gnuplot> set xrange [0:250]
gnuplot> plot sin(x)*(x**2)
```

`Gnuplot` supone que `x` es la variable independiente y `gnuplot` cambia los valores de x en el rango elegido $[0, 250]$ y grafica $x^2 \sin x$ sobre el eje Y.

La gráfica no parece tan suave como se supone de una función `sin`, lo que sucede es que `gnuplot>` evalúa la función con solamente algunos puntos (muy pocos para esta función), para cambiar el número de puntos que se deben graficar en el mismo rango de x, debemos modificar el numero de muestras:

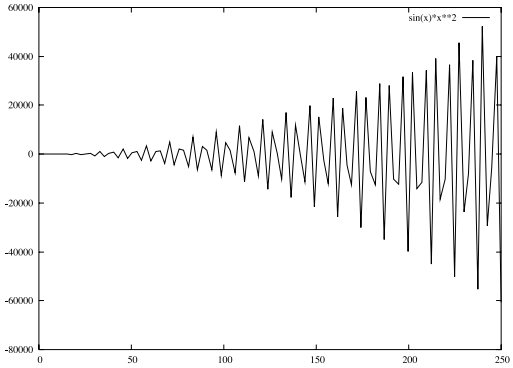


Fig. 5. `plot sin(x)*(x**2)`

```
gnuplot> set samples 1000
gnuplot> replot
```

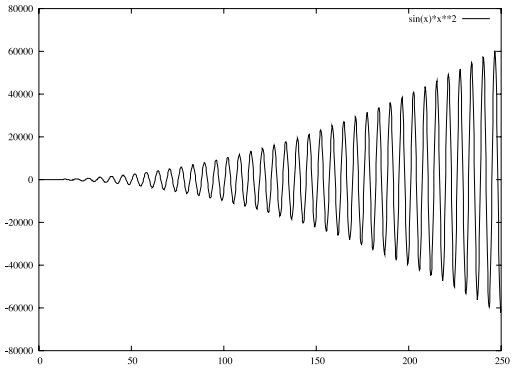


Fig. 6. `set samples 1000->replot`

¡Mucho mejor!, el comando `replot` repite el último comando `plot` usado. Con `gnuplot` se pueden definir constantes y funciones, supongamos la siguiente expresión:

$$\epsilon = \frac{\sigma}{E} + \left(\frac{\sigma}{K'}\right)^{\left(\frac{1}{n'}\right)}$$

Lo que queremos es graficar la función con ϵ (llamada `eps`) en el eje x y σ (llamada `sts`) sobre el eje y. La ecuación es válida en el rango $\sigma = [0 : 600]$. Primero, usaremos el comando `reset` para poner los valores iniciales, luego pondremos las condiciones necesarias en `gnuplot` para graficar paramétricamente (dando valores a algunos parámetros), entonces cambiaremos la variable independiente `t` para que sea `sts`. Como sabemos el rango válido para `sts`, podemos cambiar eso tambien. La función no está escrictamente definida en 0, así que podemos poner un valor muy pequeño en el límite inferior. Entonces vamos a definir desde `gnuplot` la función `eps(sts)` y las constantes `E, Kp, np`. Finalmente graficaremos la función.

```
gnuplot> reset
gnuplot> set parametric
    dummy variable is t for curves, u/v for surfaces
gnuplot> set dummy sts
gnuplot> set trange [1.0e-15:600]
gnuplot> eps(sts)=sts/E+(sts/Kp)**(1.0/np)
gnuplot> E = 206000.0
gnuplot> Kp = 1734.7
gnuplot> np = 0.2134
gnuplot> plot eps(sts), sts
```

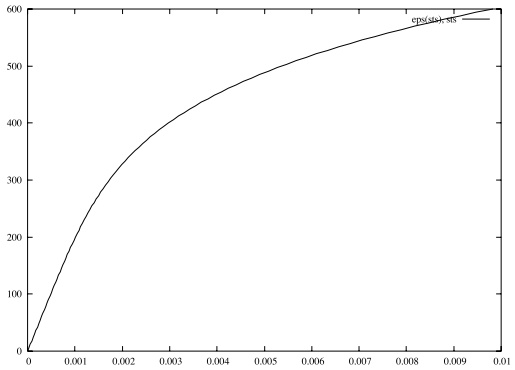


Fig. 7. `plot eps(sts), sts`

Note que `gnuplot` distingue entre letras mayúsculas y minúsculas: `E` es diferente que `e`. Para las gráficas con parámetros, el formato de los comandos para graficar es `plot <x(t)>, <y(t)>` donde `t` es la variable independiente y `x(t)` y `y(t)` pueden ser cualesquiera funciones de `t`. Las constantes se pueden introducir con formato decimal (600.456) o en notación científica (`1.0e-15`= $1.0 * 10^{-15}$). Las cadenas de caracteres no se permiten (p.ej. `a='tiempo'`).

En cualquier momento podemos saber qué funciones hemos definido, usando:

```
gnuplot> show functions

      User-Defined Functions:
      eps(sts)=sts/E+(sts/Kp)**(1.0/np)
```

También podemos saber el valor de las funciones almacenadas, el comando `show` funciona también para las variables:

```
gnuplot> show variables

      Variables:
      pi = 3.14159265358979
      E = 206000.0
```

```
Kp = 1734.7
np = 0.2134
```

Frecuentemente se deben comparar unos datos aislados con los puntos que definen alguna función. Este problema se resuelve muy bien dibujando la curva con lineas y los datos con puntos:

```
gnuplot> set xrange [0:0.01]
gnuplot> plot eps(sts), sts with lines, \
> 'ejem03.dat' with points
```

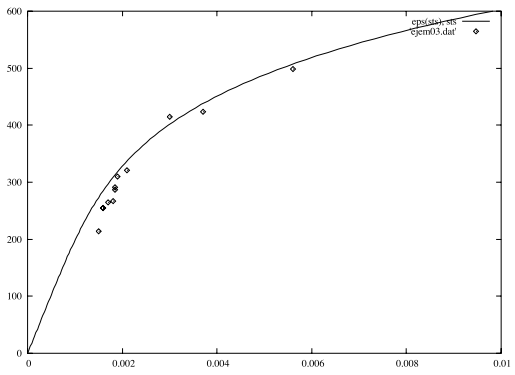


Fig. 8. `plot eps(sts), sts with lines, 'ejem03.dat' with points`

3 Dando formato a las gráficas

`Gnuplot` tiene varios parámetros que se pueden usar para cambiar la apariencia de la gráfica. Esos parámetros se pueden accesar usando el comando `set`.

`Gnuplot` ubica los identificadores de las gráficas de manera automática — llamados la clave (key)— en la esquina superior derecha de la gráfica. En nuestro ejemplo, la curva analítica pasa justo encima de la clave ocultando parcialmente las etiquetas de las líneas, haciendo ligeramente más difícil leerlas. Pero podemos cambiar el lugar de la clave usando: `set key <x>, <y>`. Cambiaremos también las etiquetas.

```
gnuplot> set key 0.007, 150
gnuplot> plot eps(sts), sts \
> title 'Curva analitica' with lines,\
> 'ejem03.dat' title 'Datos experimentales' \
> with points
```

Los parámetros `<x>` y `<y>` del comando `set key`se refieren a la ubicación del

sistema local de coordenadas de la gráfica.El comando `title` en `plot` *must* escribirse antes del comando `with`. Agregar el título de la gráfica y las etiquetas de los ejes tambiés es fácil.

```
gnuplot> set title 'Acero 1045 '
gnuplot> set ylabel 'Esfuerzo (MPa)'
gnuplot> set xlabel 'Tiempo (mm/mm)'
gnuplot> replot
```

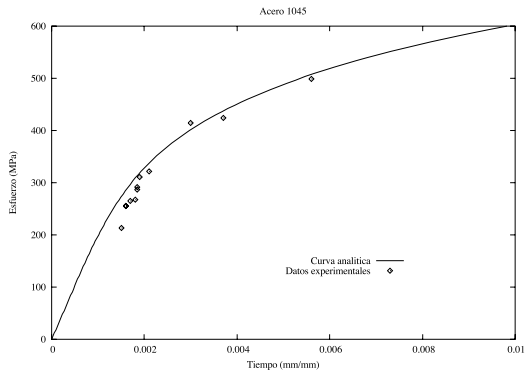


Fig. 9. `set title,..,set ylabel..,set xlabel...,replot`

Los comandos `set xlabel` y `set ylabel` toman algunos argumentos opcionales: `set xlabel 'string' <xoffset>, <yoffset>`. Los desplazamientos (offsets) se miden en caracteres. Agragaremos mas divisiones en el eje X, le pondremos una cuadricula y moveremos la etiqueta del eje Y más cerca de la gráfica actual.

```
gnuplot> set xtics -1, 0.001, 1
gnuplot> set yrange [1:1000]
gnuplot> set grid xtics
gnuplot> set grid ytics
gnuplot> set ylabel 'Stress (MPa)' 2, 0
gnuplot> replot
```

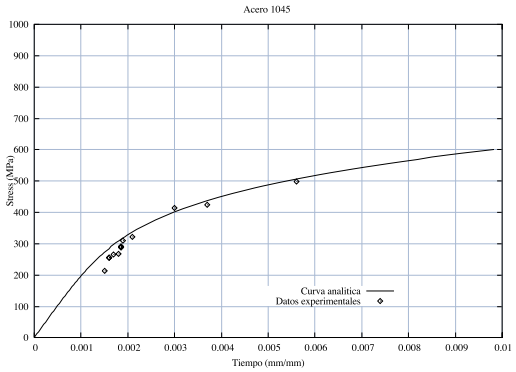


Fig. 10. `...set ylabel 'Stress (MPa)' 2, 0, ...replot`

Por default una cuadrícula será dibujada a lo largo del eje mayor. La opción `ytics` junto con la opción `xtics` le dicen a `gnuplot` que dibuje líneas tanto horizontales como verticales en cada marca en los ejes. Hay 3 parámetros básicos para los ejes:

(1) `set xrange [<x>:<y>]`, que permite especificar el rango visible
(2) `set autoscale` que obliga a `gnuplot` a usar el rango visible
(3) `set logscale <x|y>` que utiliza una escala logarítmica..

A la gráfica anterior le pondremos una escala logarítmica sobre el eje Y.

```
gnuplot> set logscale y
gnuplot> replot
```

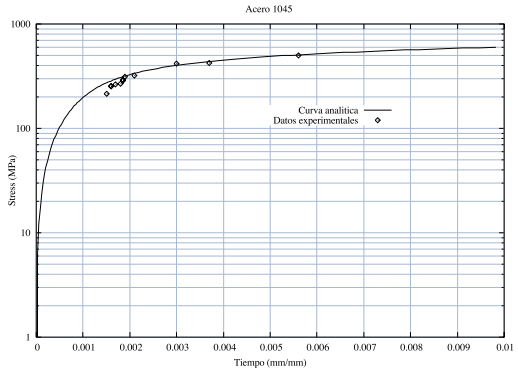


Fig. 11. `...set logscale y`

Con `gnuplot` también es posible personalizar la apariencia de las marcas en los ejes. Primero, desactivaremos la escala logarítmica del eje Y para luego establecer las marcas en el eje Y para que se lean en notación exponencial sin dígitos después del punto decimal.

```
gnuplot> unset logscale y
gnuplot> set format y '%.0e'
gnuplot> replot
```

El comando `format` usa la misma sintáxis de `printf()` del lenguaje `C`, así `set format x '%f wombats'` es un comando válido. También es posible especificar etiquetas no numéricas. Los paréntesis son necesarios cuando se especifícan etiquetas no numéricas.

```
gnuplot> set xtics ('low' 0, \
> 'medium' 0.005, 'high' 0.01)
gnuplot> replot
```

Si queremos cambiar el tamaño de las proporciones a su gráfica, puede usar el comando `set size`. El siguiente ejemplo crea una gráfica cuadrada.

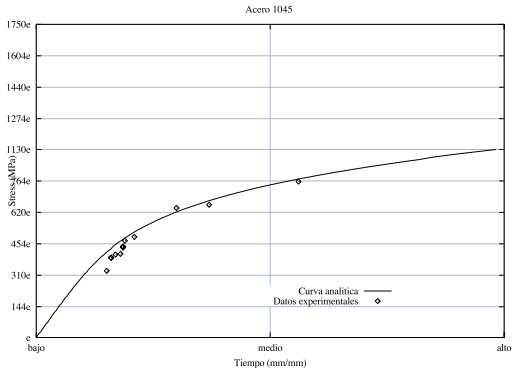


Fig. 12. `...set xtics ('low' 0, 'medium' 0.005, 'high' 0.01)...`

```
gnuplot> set size square
gnuplot> replot
```

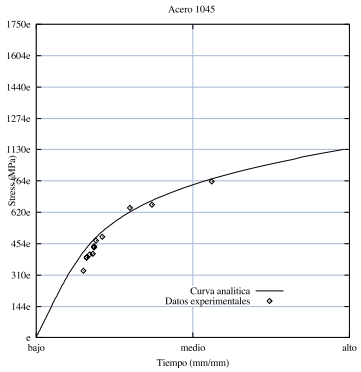


Fig. 13. `...set size square...`

4 Mas acerca de archivos de datos

`Gnuplot` permite gran flexibilidad en el formato de los archivos de datos. Por ejemplo, `gnuplot` permite líneas de comentarios en los archivos de datos. Las líneas comentadas se especifícan con un #. En consecuencia el siguiente archivo de datos

```
 # 3 fuentes de datos desde
# 5 Julio 1997
# exx exy eyy
4.563000 5.078000 -1.719000
-5.641000 3.148500 0.156000
-0.781000 3.453500 -1.250000
6.860000 8.836000 -2.188000
0.953000 5.586000 0.469000, etc.
```

es funcionalmente el mismo que este otro.

```
 4.563000 5.078000 -1.719000
-5.641000 3.148500 0.156000
-0.781000 3.453500 -1.250000
6.860000 8.836000 -2.188000
0.953000 5.586000 0.469000, etc.
```

Esta característica hace que sea posible poner encabezados en los archivos, para que podamos saber qué es lo que esos datos representan. Hasta ahora los datos que hemos usado estan separados por un espacio, pero también pueden ser serparados por comas.

4.563000,5.078000,-1.719000
-5.641000,3.148500,0.156000
-0.781000,3.453500,-1.250000
6.860000,8.836000,-2.188000
0.953000,5.586000,0.469000, etc.

En otra ventana, hagamos `cd` a "gnuplot/data/" y veamos que sucede cuando usamos el archivo "ejem04.dat". Podemos graficar la primera columna contra la segunda usando

```
gnuplot> reset
gnuplot> plot '3ch_comma.dat' using 1:2 \
> '%lf,%lf,%lf'
```

Donde `'%lf,%lf,%lf'` especifica el formato del archivo que se va a leer. La sintaxix es idéntica a `scanf` de `C`.

En ocasiones, se pueden agrupar diferentes conjuntos de datos en un mismo archivo. `Gnuplot` le permite graficar solo ciertos conjuntos en un archivo, no importa que tan grande sean los conjuntos, deberán estar separados por dos líneas en blanco, como en el siguiente ejemplo:

1 23.4
2 24.5
3 25.8
4 27.9

1 23.1
2 24.4
3 25.5
4 28.3

1 22.9

2 24.6
3 25.9
4 27.8

Podemos usar el comando `index` para especificar cual conjunto de datos queremos graficar. Si quiere grragicar solamente el segundo y el tercer conjunto:

```
gnuplot> reset
gnuplot> set style data linespoints
gnuplot> set xrange [0:5]
gnuplot> plot 'ejem05.dat' index 1:2 \
> using 1:2
```

Donde el índice `index 1:2` denota el conjunto 1 hasta el 2, la enumeración de los conjuntos empieza desde 0 .

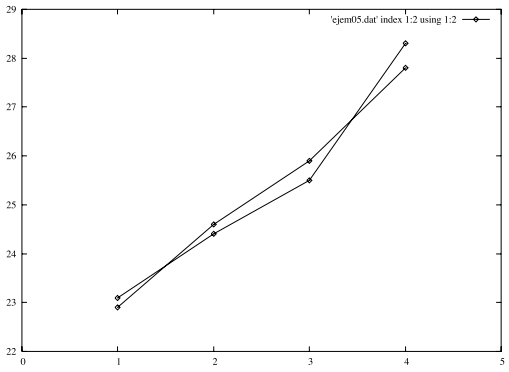


Fig. 14. `plot 'ejem05.dat' index 1:2 using 1:2`

5 Gráficas en 3D

`Gnuplot` también puede hacer gráficas en 3D con una lista de datos o con funciones analíticas como en el caso de las gráficas en 2D, las graficas son como la que resulta al interactuar con `gnuplot` con la siguiente lista de comandos.

```
gnuplot> set style data lines
gnuplot> set parametric
    dummy variable is t for curves, u/v for surfaces
gnuplot> set view 60, 60, 1, 1
gnuplot> set xlabel 'x'
gnuplot> set ylabel 'y'
gnuplot> set zlabel 'z'
gnuplot> splot u,u+v,sin(0.5*(u+v))
```

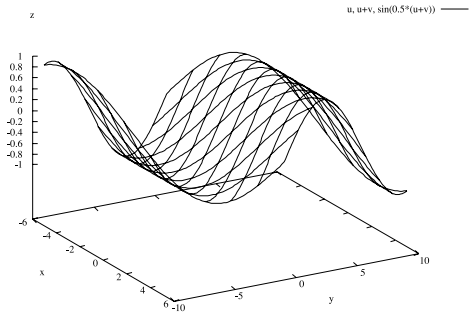


Fig. 15. `splot u,u+v,sin(0.5*(u+v))`

Las gráficas en 3D usan `splot` en lugar de `plot`, la 's' es por 'superficie'. La perspectiva se puede modificar usando los comandos `set view <rot_x>, <rot_y>,<rot_z>, <scale_x>,<scale_y>, <scale_z>`, que sirven para rotar las vistas en los ejes x, y y z y para escalar cada uno de los ejes.

`Gnuplot` También puede graficar con líneas ocultas

```
gnuplot> set hidden3d
gnuplot> splot u,u+v,sin(0.5*(u+v))
```

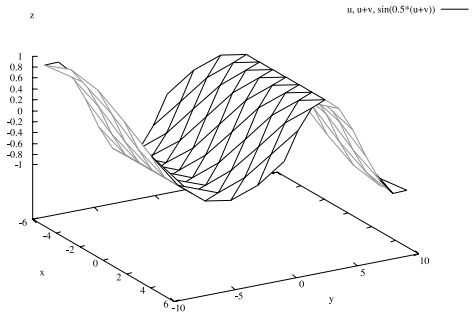


Fig. 16. `... set hidden3d ...`

También puede graficar los contornos basados en los valores del eje z (`gnuplot` no puede dibijar datos en 4D – solamente 3D mas un canal de datos), se puede hacer que se grafiquen los contornos, dibujados en la superficie, en la base, o en ambos lugares.

```
gnuplot> set contour both
gnuplot> splot u,u+v,sin(0.5*(u+v))
```

`Gnuplot` puede leer datos desde un archivo de texto.Pero antes vamos a quitar la opción de líneas ocultas y dejar los ejes como al inicio

```
gnuplot> unset hidden
```

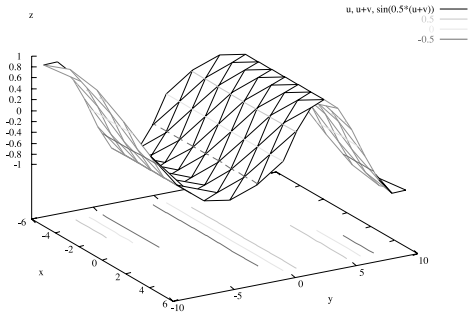


Fig. 17. ... `set hidden3d` ...

```
gnuplot> set style data lines
gnuplot> set view 60,20,1,1
gnuplot> set xtics -3000,1000,3000
gnuplot> set xlabel 'Eje Axial'
gnuplot> set ylabel 'Esfuerzo'
gnuplot> set zlabel 'Esfuerzo transversal'
gnuplot> unset key
gnuplot> splot 'ejem02.dat'
```

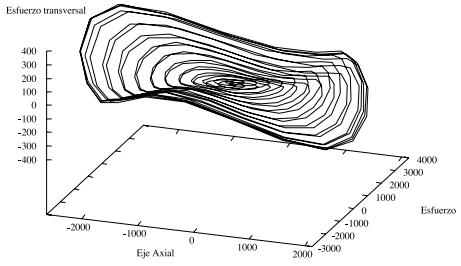


Fig. 18. ... `set hidden3d` ...

6 Grabando el trabajo hecho

El comando `save`de `gnuplot` es una manera fácil de grabar todos ajustes que hemos hecho a la gráfica, constantes, definiciones, definiciones de funciones y hasta el último comando plot, se graba enn un archivo. `Gnuplot` guarda la información en un archivo de texto en formato ASCII que puede leer `gnuplot` usando el comando `load <file>`. El siguiente es un ejemplo de lo que se puede grabar en `gnuplot`

```
set title "Acero 1045" 0,0
```

```
set notime
set rrange [-0 : 10]
set trange [0 : 600]
set urange [-5 : 5]
set vrange [-5 : 5]
set xlabel "Deformacion (mm/mm)" 0,0
set xrange [0 : 0.01]
set ylabel "Esfuerzo (MPa)" 0,0
set yrange [0 : 600]
set zlabel "" 0,0
set zrange [-10 : 10]
set autoscale r
set noautoscale t
set autoscale  y
set autoscale z
set zero 1e-08
eps(sts)=sts/E+(sts/Kp)**(1.0/np)
E = 206000.0
Kp = 1734.7
np = 0.2134
```

Notemos que `gnuplot` solamente graba una lista de comandos de `gnuplot` .

Hasta ahora, hemos interactuado con `gnuplot` tecleando los comandos en el prompt. Pero lo mejor y más fácil es interactuar con `gnuplot` desde un archivo de texto, editandolo desde nuestro editor de texto favorito, especialmente si estamos creando una gráfica compleja que poner y quitar parámetros y definir variables.

El siguiente archivo está en "gnuplot/data/script1.gp".

```
reset
set style data lines

# Empieza la seccion de parametros
set parametric
set trange [0:2.0*pi]

# establece el rango de la grafica
set xrange [-3500:3500]
set yrange [-3500:3500]

# pone todas las marcas en los ejes
set xtics -10000,1000
set ytics -10000,500
```

```
# pone el formato en las marcas de los ejes
set format '%g'

# Escribe el titulo en los ejes y en la grafica
set title 'Carga fuera de fase para el acero 1045'
set ylabel 'Deformacion lateral (mm/mm)'
set xlabel 'Deformacion axial (mm/mm)'
set key 2800,2800

# Establece la cuadricula
set grid

# Dibuja la grafica
plot 'ejem02.dat' u 1:2 \
  title 'experimental', \
  2500.0*cos(t), 2500.0*sin(t) \
  title 'teorica'
```

Ahora edite el archivo en nuestro editor favorito. Ahora vamos a cargarlo en `gnuplot` .

```
gnuplot> load 'script1.gp'
```

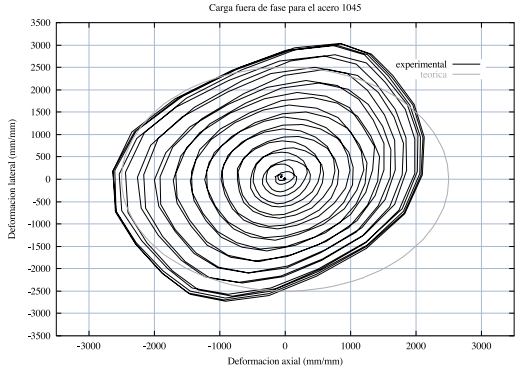


Fig. 19. `splot u,u+v,sin(0.5*(u+v))`

`Gnuplot` ignora las líneas en blanco y las líneas que empiezan con #. Todo lo demás `gnuplot` lo interpreta como un comando. Si se nos olvida algo o hemos cometido algún error `gnuplot` marca el error asi como

```
gnuplot> load 'script1.gp'
gnuplot> set frmat '%g'
             ^
       "script2.gp", line 11: valid ...
```

Usar archivos como el 'script1.gp' para interactuar con `gnuplot` es una buena idea porque los archivos sirven como registros de lo que hemos hecho.

7 Opciones de salida

`Gnuplot` ofrece muchas opciones de salida, las más usuales son las que generan archivos Postscript. El comando `set term corel` genera archivos postscrpt encapsulados .EPS mientras que usar la terminal `set term postscript` genera archivos postscript .PS Hay otras posibilidades, otros tipos de termiales. El comando `set term` muestra todas las posibles terminales.

```
gnuplot> set term

Available terminal types:
           aed512  AED 512 Terminal
           aed767  AED 767 Terminal
             aifm  Adobe Illustrator 3.0 Format
             aqua  Interface to graphics terminal server for
                   Mac OS X
         bitgraph  BBN Bitgraph Terminal
              cgm  Computer Graphics Metafile
            corel  EPS format for CorelDRAW
             dumb  printer or glass dumb terminal
              dxf  dxf-file for AutoCad (default size 120x80)
            eepic  EEPIC -- extended LaTeX picture environment
              emf  Enhanced Metafile format
            emtex  LaTeX picture environment with emTeX specials
         epslatex  LaTeX (Text) and encapsulated PostScript
     epson_180dpi  Epson LQ-style 180-dot per inch (24 pin) printers
      epson_60dpi  Epson-style 60-dot per inch printers
      epson_lx800  Epson LX-800, Star NL-10, NX-1000, PROPRINTER ...
              fig  FIG graphics language for XFIG graphics editor
             gpic  GPIC -- Produce graphs in groff using the gpic
                   preprocessor
          hp2623A  HP2623A and maybe others
           hp2648  HP2648 and HP2647
           hp500c  HP DeskJet 500c, [75 100 150 300] [rle tiff]
Press return for more:

             hpdj  HP DeskJet 500, [75 100 150 300]
             hpgl  HP7475 and relatives [number of pens] [eject]
           hpljii  HP Laserjet series II, [75 100 150 300]
             hppj  HP PaintJet and HP3630 [FNT5X9 FNT9X17 FNT13X25]
```

```
          imagen  Imagen laser printer
       kc_tek40xx  MS-DOS Kermit Tek4010 terminal emulator - color
       km_tek40xx  MS-DOS Kermit Tek4010 terminal
                   emulator - monochrome
            latex  LaTeX picture environment
               mf  Metafont plotting standard
              mif  Frame maker MIF 3.00 format
               mp  MetaPost plotting standard
          nec_cp6  NEC printer CP6, Epson LQ-800 [monocrome color
                   draft]
          okidata  OKIDATA 320/321 Standard
              pbm  Portable bitmap [small medium large] [monochrome
                   gray color]
             pcl5  HP Designjet 750C, HP Laserjet III/IV, etc.
                   (many options)
       postscript  PostScript graphics language [mode "fontname"
                   font_size]
          pslatex  LaTeX picture environment with PostScript \specials
            pstex  plain TeX with PostScript \specials
         pstricks  LaTeX picture environment with PSTricks macros
              qms  QMS/QUIC Laser printer (also Talaris 1200 and others)
            regis  REGIS graphics language
           selanar  Selanar
Press return for more:
            starc  Star Color Printer
              svg  W3C Scalable Vector Graphics driver
            table  Dump ASCII table of X Y [Z] values to output
      tandy_60dpi  Tandy DMP-130 series 60-dot per inch graphics
          tek40xx  Tektronix 4010 and others; most TEK emulators
          tek410x  Tektronix 4106, 4107, 4109 and 420X terminals
          texdraw  LaTeX texdraw environment
             tgif  TGIF X11 [mode] [x,y] [dashed] ["font" [fontsize]]
         tkcanvas  Tk/Tcl canvas widget [perltk] [interactive]
             tpic  TPIC -- LaTeX picture environment with tpic \specials
          unknown  Unknown terminal type - not a plotting device
            vttek  VT-like tek40xx terminal emulator
              X11  X11 Window System (identical to x11)
              x11  X11 Window System
             xlib  X11 Window System (gnulib_x11 dump)
gnuplot>
```

8 Gnuplot y LaTeX

Frecuentemente en el trabajo científico debemos incluir gráficas mostrando los resultados de algún proceso, LaTeX es uno de los editores de texto favorito de los científicos y `Gnuplot` ofrece varias maneras de interactuar con este procesador tan especial.

Algunas maneras son generar archivos de gráficas en formatos postscript encapsulados o postscript y luego pegarlos en LaTeX usando directivas de LaTeX

```
\begin{figure}[!!ht]
\begin{center}
\includegraphics[width=7cm]{miArchivo.eps}
\end{center}
\caption{pie de imagen}
\end{figure}
```

Para usar estos archivos debemos incluir los paquetes al inicio del documento LaTeX, después de la directiva `\documentclass[10pt]{elsart}`:

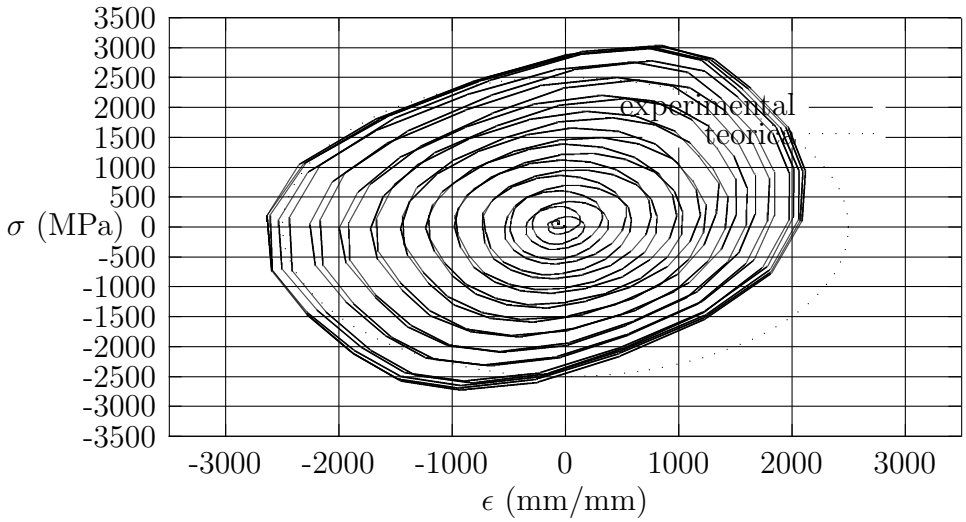
```
\usepackage{psfig}
\usepackage{graphicx}
\usepackage{psfrag}
```

Pero otra opción es usar la terminal `latex` para generar las gráficas. Esta terminal genera un archivo con formato latex y con extensión `.tex`, un archivo que puede leer LaTeX. La ventaja de usar esta manera es que podemos poner títulos en sintaxis de latex

```
gnuplot> load 'script1.gp'
gnuplot> set xlabel '$\epsilon$ (mm/mm)'
gnuplot> set ylabel '$\sigma$ (MPa)'
gnuplot> set output 'plot.tex'
gnuplot> set terminal latex
Options are '(document specific font)'
gnuplot> replot
gnuplot> set term aqua; replot
```

Luego insertar el documento en LaTeX

```
\begin{center}
\input{/data/plot.tex}
\end{center}
```



El problema de este método es que LaTeX no maneja muy bien las gráficas, entonces cuando la gráfica es muy compleja, entonces LaTeXse tarda un poco en compilar el archivo y se corre el riesgo de que se agoten las capacidades de LaTeX.

De manera que cuando las gráficas requieren de mucho cálculo, es preferible usar los métodos descritos anteriormente, es decir, generar un archivo .EPS o .PS y luego anexarlo al documento LaTeX.

Otra cosa que hay que notar es que en el penúltimo comando `set term aqua` cambia la terminal de salida a la terminal que usa mac os X para generar las gráficas (este documento se hizo en una computadora mac) se dibuja SIN poner los caracteres epsilon y sigma que genera LaTeX, así que en los archivos postscript no se verán los comandos LaTeX.

9 `Gnuplot` y MSWord

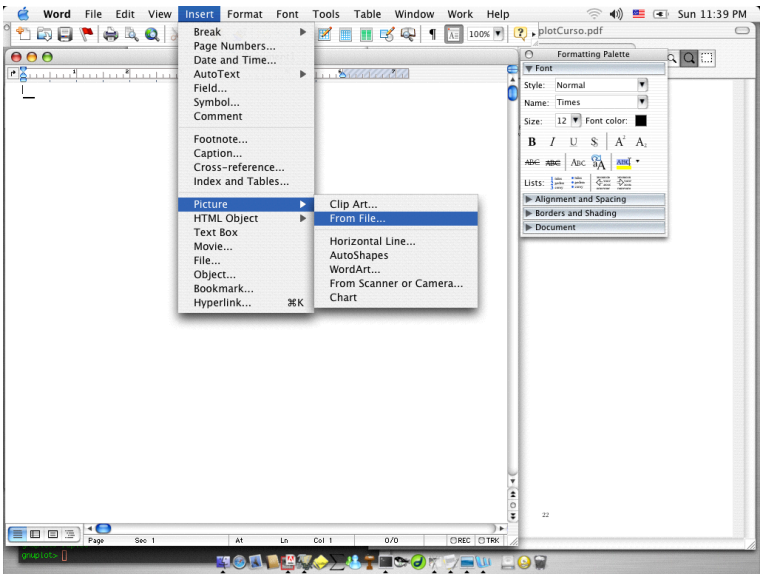
Aunque el formato postscript encapsulado es un formato adecuado para algunas aplicaciones, el formato `cgm` ("Computer Graphics Metafile") es preferido por las aplicaciones de Microsoft como Word y PowerPoint. Aunque en mi computadora (mac) no pude abrir un archivo con este formato, pero sí con el fomrato `.emf`"Enhanced Metafile Format"

```
gnuplot> load 'script1.gp'
gnuplot> set output 'plot.emf'
gnuplot> set terminal emf
Options are 'color dashed "Arial" 12'
```

```
gnuplot> replot
gnuplot> set term aqua; replot
```

Esto produce un archivo `.emf`que se llama "plot.emf" podemos incluir el archivo seleccionandolo a través de los menús

Insert → Picture → From File



Con el problema de que los títulos aparecen en letras blancas que no se pueden ver (por el color blanco del papel). Pero las últimas versiones de los productos de Microsoft también pueden abrir archivos postscript mejorados (.EPS)

10 Programando en `C` con salida para `Gnuplot`

Quizás una de las características más útiles para los científicos (o para los que algún dia lo serán y por lo pronto deben entregar tareas con los resultados graficados) es hacer un programa en C y graficar los resultados.

Una solución rápida y casi inmediata es generar un archivo de texto llamado "nombreArchivo.dat" y luego llamarlo desde `gnuplot` de la manera ya antes estudiada, con el comando `plot 'nombreArchivo.dat'`

Pero la pregunta de cualquier usuario de computadoras sería ¿puedo de alguna manera hacer eso de manera automática?, y la respuesta es ¡claro que sí! Una solución es usar las llamadas tuberias. Una tuberia es un proceso que establece un puente donde se envían mensajes a comandos externos (del entorno del

programa). Para estudiar de manera completa y profunda el uso de las tuberías en C, revise cualquier manual de C.

Enseguida transcribo un programa en ANSI C que es completamente funcional en las teminales UNIX compilándolo con `gcc` y luego haciendo `make` de la manera usual.

```
   #include <stdio.h>
   #include <stdlib.h>

   #define GNUPLOT_PATH "/usr/local/bin/gnuplot"

   int main (int argc, const char * argv[]) {
(1)  FILE *gp;
(2)  gp = popen(GNUPLOT_PATH, "w");
     if(gp == NULL){
       fprintf(stderr, "Oops, I can't find %s.", GNUPLOT_PATH);
       exit(EXIT_FAILURE);
       }
(3)  fprintf(gp, "set term aqua title \"Function A\" \n");
(4)  fprintf(gp, "set samples 2048 \n");
(5)  fprintf(gp, "plot [-512:512] -abs(x*sin(sqrt(abs(x))))+400");
(6)  fflush(gp); /* Don't forget to flush the buffer.
     getchar();
(7)  pclose(gp);
     return(EXIT_SUCCESS);
  }
```

En el programa anterior se muestran algunas líneas enumeradas en el márgen izquierdo, estos números no se deben escribir en el programa (y tampoco los paréntesis), solamente son índices para los siguientes comentarios.

(1) Se crea un apuntador de tipo archivo, con el identificador `gp`
(2) Se abre una tuberia en la ubicación definida por `GNUPLOT_PATH` en donde se puede escribir `"w"`
(3) Se escribe en la tuberia
(4) Se escribe en la tuberia
(5) Se escribe en la tuberia. Los tres anteriores puntos escriben la secuencia de comandos que se pueden ejecutar desde `gnuplot` , note que los cambios de líneas se escriben en el mismo formato que fprint
(6) Al terminar de usar la tuberia se debe "limpiar" para liberar la memoria usada por los mensajes enviados.
(7) Se cierra la tuberia.

Con esto terminamos el breve curso de `Gnuplot` .

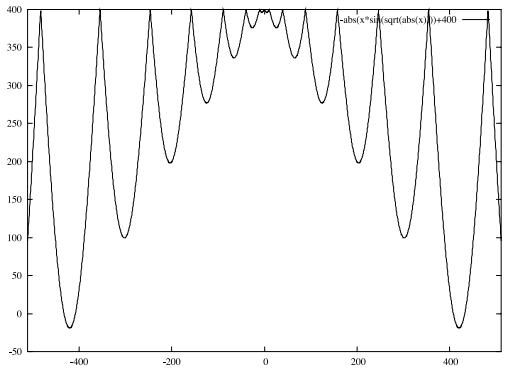


Fig. 20. `plot [-512:512] -abs(x*sin(sqrt(abs(x))))+400`

References

[1] `Gnuplot` Short Course. Documento disponible en internet en la página internet

`http://www.me.umn.edu/courses/shortcourses/gnuplot/`