Manejo de gráficas y datos experimentales a través de gnuplot

Andrés M. Vargas H.[*]

1. Introducción

La representación gráfica de una función matemática permite analizar la dependencia entre las variables de una forma más sencilla que la propia expresión matemática, de manera similar sucede con los datos experimentales, un conjunto de datos tabulados no provee con sencillez la dependencia entre las variables, no pasa lo mismo con las gráficas, éstas últimas permiten realizar de diversas formas el análisis de los resultados, pues se puede observar la forma que adquieren los datos, además, arrojan una idea de qué expresión matemática puede ser aplicable para la descripción de los resultados con lo que se puede realizar una comparación con lo que la teoría prevé.

2. ¿Qué es gnuplot?

gnuplot es un software [1] diseñado para el manejo de gráficas basado en líneas de comando, mediante este software es posible crear gráficas en 2 y 3 dimensiones (1 y 2 variables respectivamente), dar formato y exportar imágenes de alta calidad para ser insertadas en textos académicos y en impresión de gran formato, en este documento se dará una breve introducción al manejo de las funciones básicas de *gnuplot*[2] entre las cuales se encuentra el manejo de gráficas basadas en datos experimentales y el método para exportarlas en formato vectorial.

2.1. Instalación en Linux

Para realizar la instalación en sistemas GNU/Linux en la terminal se ejecuta el siguiente comando[3]:

```
# aptitude install gnuplot gnuplot-doc
```

Con lo que instalará *gnuplot* y su respectiva documentación, si desea contribuir al desarrollo de *gnuplot* o acceder al código fuente visite la siguiente página web : http://www.gnuplot.info/.

2.2. Instalación en MS-Windows

A través de su navegador web acceda a: http://sourceforge.net/projects/gnuplot/files/gnuplot allí es posible encontrar la última versión del software para éste sistema operativo, una vez realice la descarga, descomprima el archivo *zip,* el ejecutable se encuentra en: .../gnuplot/binary/gnuplot.exe.

3. Funciones analíticas

gnuplot permite realizar gráficas de funciones analíticas, existen básicamente dos modos: el primero consiste en definir una función, lo que implica asignar una etiqueta por ejemplo, $f(x)$, $g(x)$, $h(x,y)$, a determinada expresión matemática, a través de esto se guarda la expresión en la memoria con lo que

[*]Correcciones y sugerencias pueden ser remitidas a la dirección de correo electrónico: amvargash@correo.udistrital.edu.co

[1] Los autores definen a *gnuplot* como *freeware* acceda a /usr/share/doc/gnuplot/FAQ.pdf.gz para consultar la información al respecto.

[2] Versión utilizada *gnuplot* 4.4.0

[3] Si su equipo se encuentra sin conexión a internet recomiendo visitar la página web del proyecto keryx: http://www.keryxproject.org/.

después se puede graficar la función simplemente llamándola con la etiqueta que se le ha asignado, *gnuplot* distingue entre mayúsculas y minúsculas por lo que $A(x) \neq a(x)$, el otro modo consiste en graficar sin haber definido la función previamente, en este caso es necesario indicar la expresión matemática luego del comando, a continuación se describe el proceso.

3.1. Funciones en gnuplot

gnuplot provee al usuario de una serie de funciones matemáticas, trigonométricas, e hiperbólicas, el cuadro 1 muestra las funciones definidas en *gnuplot*.

Llamada	Devuelve
exp(x)	exponencial de x (e^x)
log(x)	logaritmo natural de x ($log_e(x)$)
log10(x)	logaritmo en base 10 de x ($log_{10}(x)$)
abs(x)	valor absoluto de x
x**n	potencia n de x (x^n)
sqrt(x)	Raíz cuadrada de x ($\sqrt{x}$)
sin(x)	seno de x
cos(x)	coseno de x
tan(x)	tangente de x
asin(x)	arcoseno de x
acos(x)	arcocoseno de x
atan(x)	arcotangente de x
sinh(x)	seno hiperbólico de x
cosh(x)	coseno hiperbólico de x
tanh(x)	tangente hiperbólica de x
asinh(x)	arcoseno hiperbólico de x
acosh(x)	arcocoseno hiperbólico de x
atanh(x)	arcotangente hiperbólica de x

Tab. 1: Funciones Matemáticas

También puede observar todas las funciones definidas en *gnuplot* y su respectiva descripción ejecutando:

```
gnuplot>help functions
```

Al trabajar con funciones trigonométricas por defecto *gnuplot* asume que los argumentos están dados en radianes, para saber esta información se ejecuta el comando:

```
gnuplot>show angles
```

Para realizar el cambio a grados, ejecute:

```
gnuplot>set angles degrees
```

Para realizar el cambio a radianes, ejecute:

```
gnuplot>set angles radians
```

Como lo habrán notado con pocas referencias de inglés las líneas de comando de *gnuplot* se tornan "intuitivas".

3.2. Definiendo funciones.

Para definir una función en *gnuplot*, se debe citar una letra de función y sus variables dependientes, por ejemplo, para definir una función armónica sinusoidal cuyo argumento es la variable x se ejecuta:

```
gnuplot>g(x)=sin(x)
```

Se puede utilizar la suma (+), multiplicación (∗), división (/) y evaluación de funciones, por ejemplo, si se define otra función

```
gnuplot>h(x)=exp(x)
```

Pueden definirse otras funciones compuestas de *g(x)* y *h(*x), así:

```
gnuplot>v(x)=h(x)+g(x)
gnuplot>a(x)=h(x)/g(x)
gnuplot>i(x)=h(g(x))
```

De esta manera, *v(x)=sin(x)+exp(x)*, *a(x)=sin(x)/exp(x)* e *i(x)=exp(sin(x))*, para ver las funciones que han sido definidas se utiliza el comando:

```
gnuplot>show functions
```

Hay casos en los que será necesario definir constantes, esto se realiza ejecutando el nombre de la constante seguido de su valor:

```
gnuplot>w=2.286
```

Se puede ver el listado de constantes definidas ejecutando:

```
gnuplot>show variables
```

La constante π (pi) está definida previamente en *gnuplot* con un valor de 3,14159265358979, lo que representa una gran utilidad al combinarlas con las funciones, así:

```
gnuplot>f(x)=sin(w*pi*x)
```

La función $f(x) = sin(2{,}286 * 3{,}14159 * x)$, para establecer funciones que definen superficies, se utiliza una letra de función y se especifica su dependencia de las variables x e y, así:

```
gnuplot>B(x,y)=x*sin(y)
```

3.3. Graficando

Las funciones que dependen de una sola variable se grafican utilizando el comando *plot*, así:

```
gnuplot>plot f(x)
```

donde *f(x)* pudo haber sido definida previamente o escribiendo la función justo después del comando así:

```
gnuplot>plot x**2
```

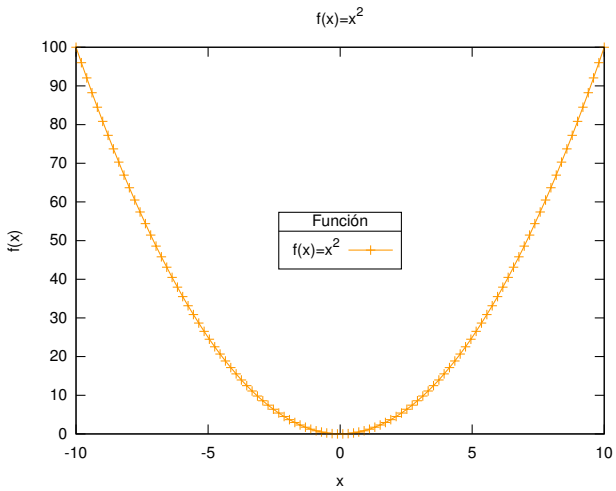

Fig. 1: Gráfica $f(x) = x^2$

Graficando de ésta manera la función x^2 (ver figura 1).

Existen diversos estilos para realizar el trazo de la función, por ejemplo, al ejecutar:

```
gnuplot>plot x**2 with lines lt 7
```

Se obtiene el trazo de x^2 con líneas (*lines*) de color negro (*lt 7*) y no rojo que es el tipo *lt 1*, también:

```
gnuplot>plot x**2 with points lt 3
```

Se obtiene el trazo de x^2 con puntos (*points*) en vez de líneas (*lines*) con el estilo *lt 3* (puntos de color azul), o de manera simultánea:

```
gnuplot>plot x**2 w linespoints lt 7
```

Se obtendrá el trazo de x^2 con puntos y líneas, note que *gnuplot* reconoce la letra *w* como si fuera *with*, también se puede abreviar *lines* con *l* y *points* con *p*, una forma alternativa de escribir el anterior comando es:

```
gnuplot>plot x**2 w lp lt 7
```

Puede graficar dos o más funciones en el mismo plano coordenado (ver figura 2) separándolas mediante comas (,), así:

```
gnuplot>plot sin(x/2),cos(x/2),x/10
```

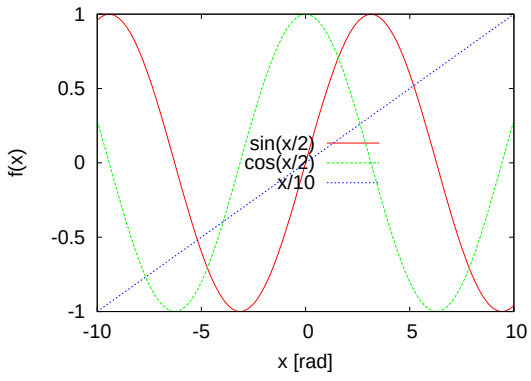

Fig. 2: Graficando múltiples funciones

Las funciones que dependen de dos variables se grafican utilizando el comando *splot*, así:

```
gnuplot>splot h(x,y)
```

donde h(x,y) pudo haber sido definida previamente o escribiendo la función justo después del comando así:

```
gnuplot>splot x*y with pm3d
```

Graficando de ésta manera la función $x * y$, mediante *with pm3d* se reemplaza el conjunto de líneas por una superficie continua. (ver figura 3).

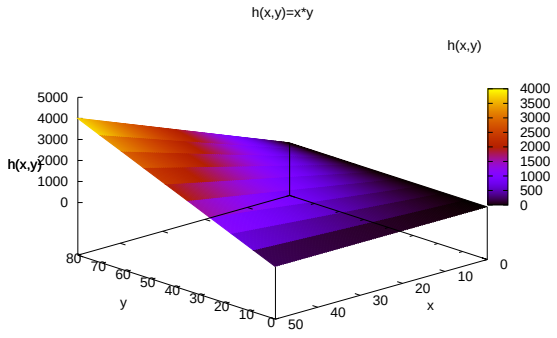

Fig. 3: Gráfica de la función $h(x, y) = x * y$

Hasta el momento todas las gráficas se han realizado sobre el plano cartesiano, sin embargo es posible utilizar coordenadas polares, esto se consigue a través de:

```
gnuplot>set polar
```

Donde se usa t (en vez de x) como variable muda, un ejemplo para definir las funciones es $f(t) = \sin(2 * t)$. Para hacer más comprensible la gráfica es necesario ajustar la cuadrícula a polar lo que se consigue mediante:

```
gnuplot>set grid polar
gnuplot>plot f(t)
```

La figura 4 muestra un ejemplo, para continuar trabajando con coordenadas rectangulares se ejecuta:

```
gnuplot>unset polar
```

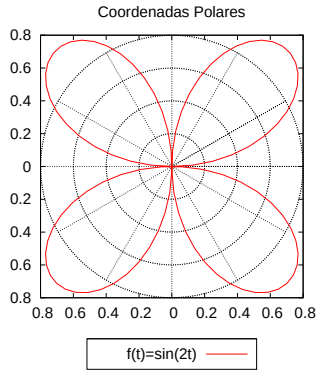

Fig. 4: Coordenadas Polares $f(t) = sin(2 * t)$

4. Datos experimentales

Cuando se trabaja con datos experimentales se debe crear un archivo de texto plano (archivos con extensión *.dat* o *.txt*) que contenga en columnas dichos datos, puede crear previamente la tabla de resultados en una hoja de cálculo como la haría normalmente, luego pegar los datos en un editor de texto plano y guardar el archivo, es importante que este archivo se encuentre en el directorio en que se ejecuta *gnuplot* (para usuarios de MS-Windows será necesario ubicar estos archivos en la carpeta que se encuentra el ejecutable *gnuplot.exe*, para usuarios GNU/Linux bastará con que ejecuten la terminal desde el directorio en el cual tienen sus archivos, si se encuentra en *gnuplot* y desconoce el directorio desde el cual se ejecuta, ejecute *gnuplot>pwd*), en esta sección se muestra breve-

mente cómo realizar la gráfica de estos datos y cómo ajustarlos a una expresión matemática.

4.1. Gráficas y linealización de datos experimentales

Para ajustar datos experimentales a determinado tipo de funciones se utiliza el comando *fit* que traduce ajuste o arreglo, observe un archivo (nombrado 'datos.dat') de datos experimentales que ha sido creado con el fin de exponer el método:

```
#file 'datos.dat'
#Nominal power (40W)
#COL1:Distance(d ± 0,001)m
#COL2:Illumination (i ± 0,01)klux
#COL3:Illumination (i ± 1)lux
#COL4:Luminous Flux (Lumen)
#---------------------------
0.070   3.37    3370    0.028
0.070   3.29    3290    0.028
0.070   3.23    3230    0.027
0.070   3.31    3310    0.028
0.070   3.23    3230    0.027
0.140   1.25    1250    0.011
0.140   1.25    1250    0.011
0.140   1.23    1230    0.010
0.140   1.25    1250    0.011
0.140   1.22    1220    0.010
0.210   0.62    620     0.005
0.210   0.62    620     0.005
0.210   0.64    640     0.005
0.210   0.65    650     0.005
0.210   0.62    620     0.005
0.280   0.38    380     0.003
0.280   0.38    380     0.003
0.280   0.37    370     0.003
0.280   0.38    380     0.003
0.280   0.39    390     0.003
0.350   0.24    240     0.002
0.350   0.24    240     0.002
0.350   0.25    250     0.002
0.350   0.24    240     0.002
0.350   0.24    240     0.002
```

Las líneas que comienzan por el símbolo # son ignoradas por *gnuplot* por lo que se utilizan para incluir comentarios en los archivos de datos, estos datos están organizados en columnas, *gnuplot* reconoce las columnas con números que aumentan hacia la derecha, estos datos corresponden a una medición de los valores que adquiere la iluminación (columna 2) cuando se varia la distancia (columna 1), la columna 3 muestra la conversión de kilolux *klux* a *lux* y la columna 4 muestra el flujo luminoso calculado a partir de la iluminación. Se grafica usando:

```
gnuplot>plot 'datos.dat' using 1:3
```

La opción que se ha agregado es *using 1:3*, lo que hará que se grafique la columna 1 en el eje horizontal y la columna 3 en el eje vertical (aquí se utilizará únicamente los datos obtenidos para la iluminación en *lux* -columna 3- para las diferentes distancias -columna 1-) , ejecutando esto se obtiene la gráfica 5.

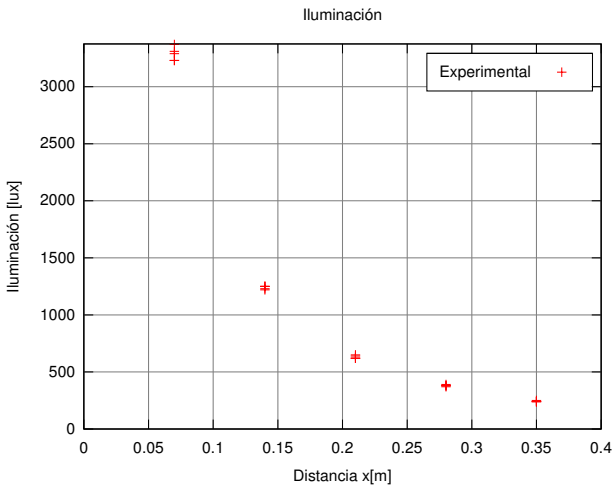

Fig. 5: Gráfica del archivo 'datos.dat'

En la figura 5 es posible observar que los datos pueden ajustarse a una función inversa de la forma $f(x) = a/x^b$, así se realiza una definición:

```
gnuplot>f(x)=a/x**b
```

El comando *fit* permite hallar la mejor configuración de $f(x)$ que se ajusta a los datos, ejecutando:

```
gnuplot>fit f(x) 'datos.dat' \
using 1:3 via a,b
```

Esta orden permite hallar los valores de a y de b (*via a,b*) que mejor se ajustan a los datos experimentales que se encuentran en las columnas 1 y 3 (*using 1:3*) del archivo 'datos.dat' (la barra inclinada \ permite seguir introduciendo la orden en la siguiente línea), es necesario tener cuidado al trabajar con archivos de múltiples columnas, en cada comando que se llame al archivo se deberá especificar cuáles columnas serán usadas. En la información que arroja el comando *fit* se encuentra:

```
Final set of parameters
=======================
a               = 62.2868
b               = 1.4927
Asymptotic Standard Error
=========================
+/- 3.047        (4.892%)
+/- 0.01919      (1.286%)
```

Después de los cálculos usando mínimos cuadrados, la función $f(x)$ que mejor se ajusta a los datos es aquella en la cual a tiene un valor de 62.28 y b de 1.49, el comando *fit* define en *gnuplot* las variables a y b con sus respectivos valores así que la función $f(x) = a/x^b$ estará definida como: $f(x) = 62{,}28/x^{1{,}49}$ (compruebe esta afirmación ejecutando *show variables* y *show functions*). Para finalizar, se grafican los datos experimentales y su respectiva linealización mediante:

```
gnuplot> plot 'datos.dat' using 1:3 \
title "Experimento", f(x) title \
sprintf("%.2f / ( x^{%.2f} )",a,b)
```

La gráfica de $F(x)$ se rotula a partir de los valores obtenidos para las variables a y b utilizando sintáxis del lenguaje de programación C[4], Obteniendo la figura 6.

[4] sprintf("%.2f / (x^{%.2f})",a,b) : Imprime la primera variable que se encuentra después de la primer coma (la variable a) que es de tipo *float* con 2 decimales de precisión (%,2f) luego imprime "/ (x ^{" e imprime la siguiente variable (en éste caso b) también con dos decimales y finalmente imprime "})", recuerde que la secuencia "x^{n}" es interpretada como x^n.

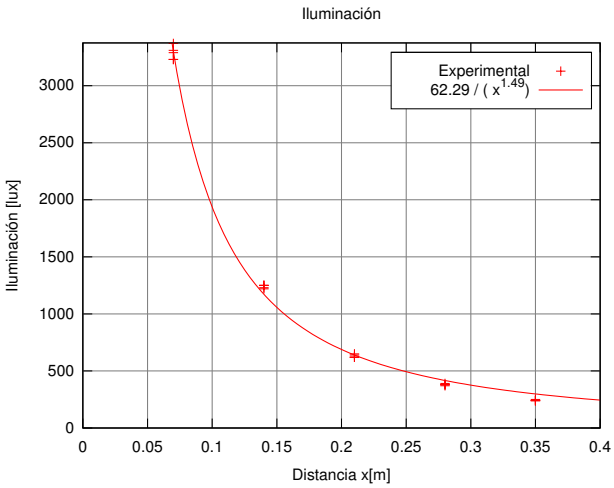

Fig. 6: Gráfica del archivo 'datos.dat' y se respectiva linealización.

El comando *fit* puede utilizarse para las diferentes funciones y combinaciones con que *gnuplot* trabaja, esta es una de las ventajas que ofrece *gnuplot* frente a las hojas de cálculo que trae un número limitado de funciones a las cuales pueden ajustarse los datos que en ellas se grafica, se hubiera podido tener unos datos con la forma $f(x) = k\exp(-ax)sin(bx)$ (como lo puede ser al trabajar con osciladores amortiguados) y usar *fit* para hallar los valores de los diferentes coeficientes k, a, y b.

4.2. Gráficas con márgenes de incertidumbre

Ninguna medición realizada con instrumentos físicos es exacta, a lo sumo se puede establecer un intervalo en el cual se puede garantizar que se encuentra la medida, esto gráficamente repercute en que cada dato no puede estar representado por un punto sino que debe estar representado por un símbolo que garantice que existe un intervalo y que debe dar una

idea de qué tan grande es con respecto a la medición, entre mayor sea el intervalo la medida va ser menos confiable pues significa que la incertidumbre es mayor, el símbolo + en la figura 5 y 6 es el que *gnuplot* toma por defecto y marca un punto, no un intervalo, a continuación se muestra como ajustar el ancho y la altura del símbolo "Ξ" de tal manera que la gráfica indique directamente el margen de incertidumbre, observe un archivo de datos experimentales:

```
#inc.dat
#COL1: Distancia [cm]
#COL2: Incertidumbre en
#la medida de la distancia [cm]
#COL3: Tiempo [s]
#COL4: Incertidumbre en
#la medida del tiempo [s]
#-------------------------
20      2       1.2     0.1
31      3       2.3     0.2
41      2       3.6     0.5
49      1       4.1     0.5
53      2       5.2     0.3
60      1       6.0     0.1
68      2       7.2     0.8
78      1       8.1     0.1
80      5       9.3     0.1
```

El objetivo es graficar estos datos, en la columna 1 se muestra la distancia recorrida por un móvil en centímetros, en la columna 2 se muestra la incertidumbre para cada medición de distancia, en la columna 3 se muestra el tiempo que requirió el móvil para recorrer cada distancia en segundos y la columna 4 muestra la incertidumbre en la medida del tiempo. Para lograr el objetivo propuesto se ejecuta:

```
gnuplot>plot 'inc.dat' using 3:1:4:2
with xyerrorbars
```

Primero, *'inc.dat'* es el nombre del archivo de texto plano, observe que la sección *using* tiene un formato *using* x:y:Δx:Δy, (donde Δx y Δy son las columnas que contienen los datos relativos a la incertidumbre en la medida de x e y respectivamente) así se graficará la columna 3 en el eje horizontal, la columna 1 en el eje vertical, la columna 4 se utilizará para modificar la anchura de Ξ y la columna 2 modificará su altura. El resultado se muestra en la figura 7.

gnuplot puede operar con las columnas por ejemplo si se hubiera requerido graficar los datos anteriores de distancia, en metros en lugar de centímetros se hubiera reemplazado en *using* 1 por ((\$1)*10**-2) y 2 por ((\$2)*10**-2) ya que $1cm = 10^{-2}m$, \$1 lo que hace es llamar a cada dato de la columna 1 y con el realiza la operación indicada entre los paréntesis externos, en este caso lo multiplica por 10^{-2}. El formato *using* quedaría:

```
using 3:((($1)*10**-2):4:((($2)*10**-2)
```

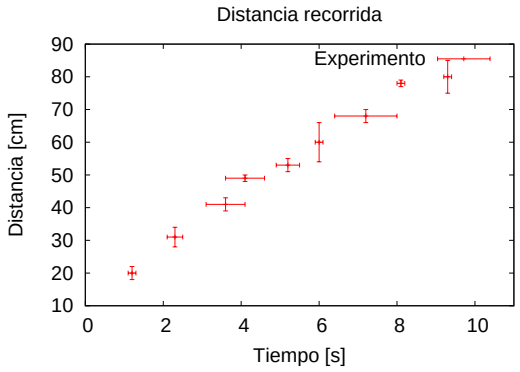

Fig. 7: Gráfica con márgenes de incertidumbre

5. Trazando campos vectoriales

Con *gnuplot* es posible trazar campos vectoriales, un campo vectorial es la asignación de un vector a cada punto del espacio, para ello debe crear un archivo de datos que contenga los columnas:

- x
- y

- Componente horizontal del campo para el punto (x, y)
- Componente vertical del campo para el punto (x, y)

En el ejemplo observado en la figura 8 se obtiene de un archivo que contiene las columnas:

x	y	E_x	E_y

Para esto se utilza el comando:

```
gnuplot>plot "./archivo_de_datos.dat" \
using 1:2:($3*0.1):($4*0.1) \
with vectors
```

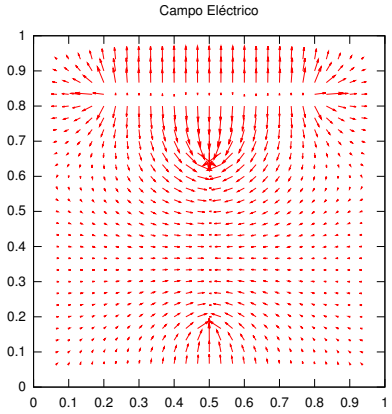

Fig. 8: Representación de un campo vectorial

Es posible observar que el formato para vectores es el siguiente: x:y:E_x:E_y[5], sin embargo debido a que es posible que los valores de E_x y E_y sean de un orden de magnitud elevado lo que se hizo mediante ($3∗0,1) y ($4 ∗ 0,1) es graficar el $10\,\% = 0{,}1$ de ese valor.

6. Dando formato a las gráficas

Una gráfica sin su respectiva rotulación, es una gráfica vacía, son bellas líneas sin contenido, a continuación se muestran los comandos necesarios para establecer la información que permita identificar y hacer legible una gráfica:

El título de la gráfica se establece a través del comando:

```
gnuplot>set title "Título"
```

Se establece el intervalo del eje horizontal (el dominio) que será mostrado, a través de:

```
gnuplot>set xrange[de:hasta]
```

y el rango del eje vertical (el rango) con:

```
gnuplot>set yrange[de:hasta]
```

donde '*de*' y '*hasta*' son los valores del inicio y el fin del rango respectivamente, se recomienda establecer el intervalo para el eje horizontal dejando que *gnuplot* adecúe el intervalo mostrado en el eje vertical. Para observar los cambios que han sido realizados, se actualiza la información del tablero mediante el comando:

```
gnuplot>replot
```

Por defecto, cuando se grafican datos experimentales la etiqueta de estos datos es el nombre del archivo de texto plano, para cambiarla basta con ejecutar:

```
gnuplot>plot 'datos.dat' title 'Nombre'
```

gnuplot genera configuraciones por defecto en cuanto a los intervalos en que se divide cada eje (no confundir con el intervalo mostrado) , pueden cambiarse a través de:

```
gnuplot>set xtics 1
```

se mostrará en el eje horizontal divisiones de 1.

```
gnuplot>set ytics 0.5
```

se mostrará en el eje vertical divisiones de 0.5.

Se etiquetan los ejes, mediante

[5] También es posible graficar campos vectoriales en 3 dimensiones para ello necesitará un archivo de seis columnas x:y:z:E_x:E_y:E_z y usará el comando *splot* en lugar de *plot*.

```
gnuplot>set xlabel "Este es mi eje x"
gnuplot>set ylabel "Este es mi eje y"
```

Se muestra la cuadrícula con:

```
gnuplot>set grid
```

Se ubica la etiqueta que muestra los nombres de las funciones a través de:

```
gnuplot>set key center
```

donde *center* puede ser remplazado *left*, o *right*. A veces resulta conveniente enmarcar ésta etiqueta para que no se confunda la representación de la función con un dato más, esto se logra a través de:

```
gnuplot>set key box
```

O mejor resulta extraerlo de la gráfica mediante una de las siguientes opciones:

```
gnuplot>set key lmargin center
```

La ubica en el centro de la izquierda de la gráfica.

```
gnuplot>set key bmargin left
```

La ubica abajo de la gráfica hacia la izquierda, puede cambiar *left* por *right* o *center*.

```
gnuplot>set key rmargin center
```

La ubica en el centro del lado derecho de la gráfica .

```
gnuplot>set key tmargin right
```

La ubica arriba de la gráfica en la esquina derecha, puede cambiar *right* por *left* o *center*.

El formato mínimo requerido para una gráfica se muestra en la figura 9, todos los comandos que se han expuesto en esta sección se han usado para dar formato a las gráficas del presente texto, en ellas es posible observar el resultado.

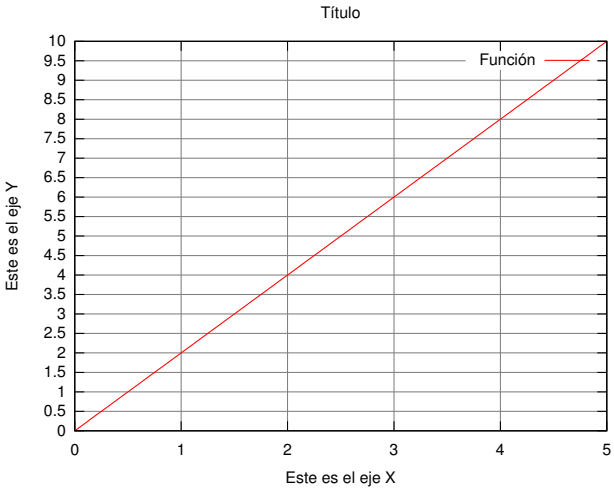

Fig. 9: Gráfica con formato básico.

7. Guardando el trabajo en gnuplot

Hasta el momento todo lo que se ha realizado se encuentra en la memoria *RAM* del sistema, existen dos formas de guardar la información con *gnuplot*, una de ellas consiste en exportar la imagen que ha sido realizada para permitir su posterior inserción en documentos, a continuación se explicará cómo obtener una imagen en formato vectorial, una vez lo haya logrado, exportarla en formato *png* o *jpg* no representará para usted mayor problema, la otra forma de guardar el trabajo realizado es crear un archivo de texto con el conjunto de comandos que hicieron posible obtener la gráfica.

7.1. Exportando las gráficas

Para visualizar los formatos en que pueden ser exportadas las gráficas realizadas ejecute el comando:

```
gnuplot>set terminal
```

Exportaré la imagen en formato *ps* cuya resolución puede modificarse sin pérdida de calidad (ya que es una formato vectorial) lo que provee un medio efectivo para diseño de gran formato, para visualizar la

imagen que será exportada ejecute el comando plot seguido de la función o los datos a graficar, seleccione el formato *ps* ejecutando:

```
gnuplot>set t postscript color enhanced
```

Esta linea determina que el formato es postscript y que la imagen será policromática, note que la letra t es una abreviación reconocida por *gnuplot* de *terminal*, si no se escribe *color* es posible que la imagen exportada sea monocromática, la opción *enhanced* habilita el uso de exponentes (en general, habilita la codificación para postscript), así toda secuencia de carácteres entre *^{}* serán tomados como exponentes, por ejemplo si tiene:

```
gnuplot>set title "x^{1/2}"
```

Aunque en el tablero de *gnuplot* se muestre x^{1/2} en el archivo postscript se verá $x^{1/2}$[6].

```
gnuplot>set output "archivo.ps"
```

Esta última linea de comando crea el archivo *'archivo.ps'* en el directorio de trabajo, luego ejecutando:

```
gnuplot>replot
```

Se envía la información al archivo, el directorio de trabajo (donde se guardó el archivo) puede ser consultado a través del comando:

```
gnuplot>pwd
```

Una vez ejecutado el comando *set terminal postscript*, la ventana de gráficas de *gnuplot* (el tablero) no se actualizará, esto se debe a que todas las instrucciones se actualizan en al archivo "archivo.ps" utilizando *plot* y *replot*, si desea regresar nuevamente al trabajo en el tablero de *gnuplot* se ejecuta:

```
gnuplot>set terminal wxt
```

[6] Una vez instalado el paquete *gnuplot-doc*, puede obtener más información acerca del formato *ps* en /usr/share/doc/gnuplot-doc/ps_guide.ps.gz

Con lo que se activa la ventana de gráficos como salida. El formato *postscript* tiene problemas para la presentación de caracteres especiales como lo son las letras tildadas ya que para ello es necesario escribirlas en formato postscript (por ejemplo para que la salida sea una ó - o tildada - se debe escribir "\363"), si utiliza este tipo de caracteres es preferible usar el formato SVG, que puede activar mediante:

```
gnulot>set terminal svg
```

Repitiendo los mismos comandos para exportar la imagen.

7.2. Creando scripts de gnuplot

Un script es básicamente un archivo de texto que tiene una secuencia de comandos, de esta manera es posible realizar una combinación de comandos mediante un editor de texto plano y luego cargarlo para que *gnuplot* ejecute ésta secuencia, se suele usar la extensión **.gp* para identificar los scripts de *gnuplot*, a continuación presento un script que he nombrado 'script.gp' para que conozca el formato:

```
# file script.gp
set title "The Gamma Factor"
c=3
f(x)=1/sqrt(1-x**2/c**2)
set xrange[0:3]
set xtics 0.5
set xlabel "Speed v [*10^8 m/s]"
set ylabel "Gamma"
set yrange[0:7]
set key box
set key height 1
set key left
set key title "Function"
set grid
set terminal postscript color enhanced
set output 'gamma.ps'
plot f(x) title "1/(1-v^{2}/c^{2})^{1/2}"
set terminal wxt
replot
```

Es posible ver que el archivo de texto *'script.gp'* contiene un listado de comandos, de igual forma que en

los archivos de datos, cualquier secuencia de caracteres hasta el final de la línea que se encuentre luego de # , son ignorados por *gnuplot*, el archivo que contiene la secuencia de comandos debe encontrarse en el directorio desde donde se ejecuta *gnuplot* (recuerde el comando *pwd*) para indicarle a *gnuplot* que cargue el script se ejecuta:

```
gnuplot>load 'script.gp'
```

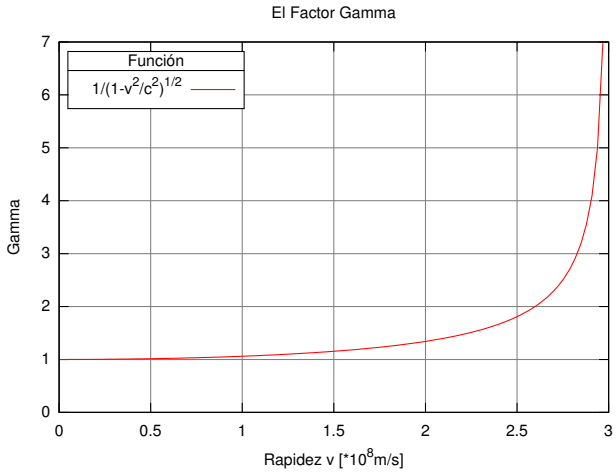

Fig. 10: El factor Gamma (Lorentz)

Obteniendo la figura 10. Una vez adquiera la costumbre de trabajar con scripts en vez de ejecutar comando a comando en *gnuplot* le resultará de utilidad:

```
gnuplot>reset
```

Con lo que se vuelve a iniciar *gnuplot* en ceros, borrando la información de variables, funciones, formatos, etc. úselo cada vez que ejecute un nuevo script.

8. Licencia de Distribución

Nota final: Aquí se ha abarcado *someramente* el manejo de *gnuplot*, utilice la documentación disponible para mayor profundidad, este documento se realizó utilizando exclusivamente software libre por lo que resulta necesario realizar un reconocimiento a las personas que hacen posible la existencia, soporte y mantenimiento de los proyectos: LyX http://www.lyx.org/, Inkscape http://www.inkscape.org/, GNU/Linux Trisquel http://trisquel.info/ y tantos otros. Éste y otros recursos del mismo autor se encuentran disponibles en: http://licamfis.comze.com/.

> "Vive como si fueras a morir mañana, aprende como si fueras a vivir siempre" Gandhi.