Gráficas en dos dimensiones en Maple

José Luis Torres Rodríguez*

Marzo 2011

Las funciones soportadas por Maple pueden ser divididas en varios grupos principales. Uno de estos grupos lo constituyen las funciones de despliegue de gráficas y animaciones. Este sistema proporciona varios comandos que tienen la capacidad de desplegar diferentes tipos de funciones; por ejemplo, pueden generar gráficas de funciones explícitas, implícitas, funciones paramétricas, gráficas de puntos, tanto en dos como en tres dimensiones. En esta hoja se presentan las principales instrucciones proporcionadas para despliegue de gráficas en dos dimensiones, algunas de las diferentes opciones que permiten modificar el aspecto de éstas, así como la manera de manipular las regiones correspondientes. También se describe la nueva función **interactive** del paquete **plots**, la cual nos facilita la creación de una gráfica a través de la nueva herramienta gráfica **Interactive Plot Builder** incluida en Maple 8, la cual describiremos más adelante.

1. Gráficas de funciones explicitas

Maple nos permite desplegar gráficas de funciones explicitas de una variable por medio de la función **plot**. Su sintaxis es la siguiente:

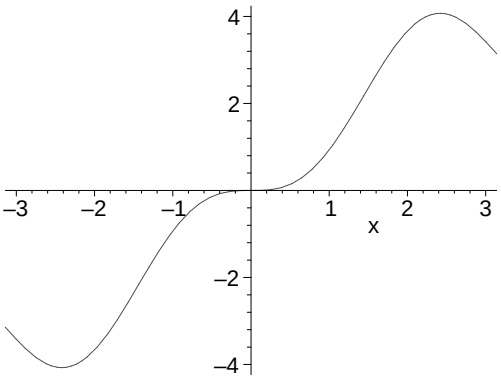
plot(f, r, ops)

Donde **f** es la función o expresión que se desea graficar (de una variable), **r** es el rango en el cual se desea desplegar dicha función, y **ops** son las opciones que se aplicarán a la gráfica. Veamos un ejemplo:

```
> f := x -> x + sin(x) - sin(2*x);
```

$$f := x \to x + \sin(x) - \sin(2\,x)$$

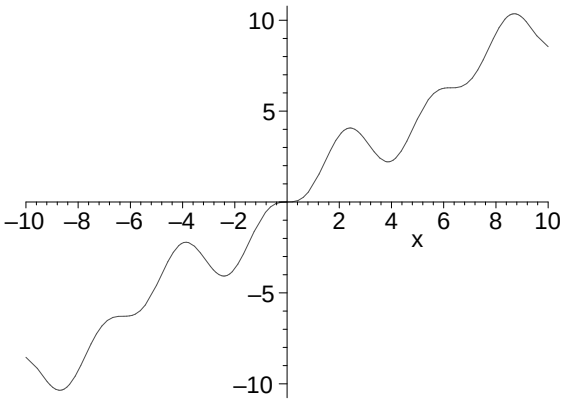
```
> plot(f(x), x=-Pi..Pi);
```



*Coordinación de Cómputo, Facultad de Ciencias, UNAM

Nótese que el rango es expresado en la forma ***"x=a..b"***, donde ***"a"*** y ***"b"*** son los extremos del rango en el cual se desea el despliegue y además $a < b$. Este rango puede omitirse, colocando simplemente la variable de la cual depende la función o expresión.

```
> plot(f(x), x);
```



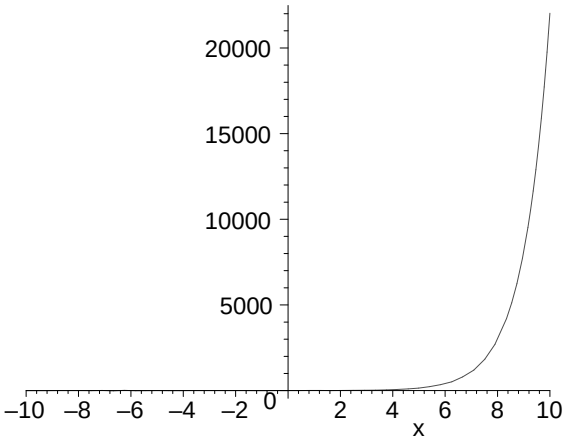
En este caso, el rango tomado por omisión es **"x = -10..10"**.

Cuando se expresa un rango para el eje x, el rango del eje y queda determinado por los valores que toma la función al evaluarla en los diferentes puntos de la variable. Esto puede provocar que las gráficas aparezcan deformadas o que su aspecto sea muy diferente del que realmente tiene la función dada, lo cual puede impedir apreciar el verdadero comportamiento de ésta. Por ejemplo:

```
> g := x -> sin(x^3)*exp(exp(cos(x))) + exp(x);
```

$$g := x \to \sin(x^3)\, e^{(e^{\cos(x)})} + e^x$$

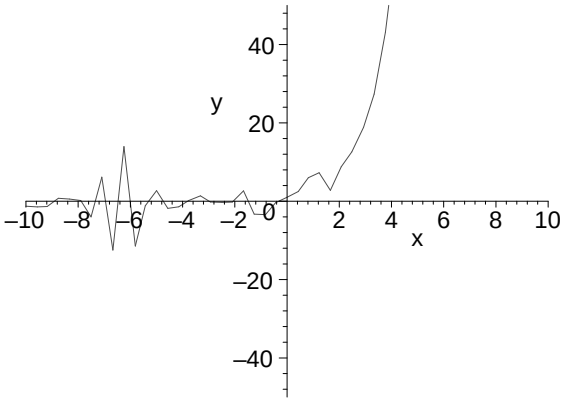
```
> plot(g(x), x=-10..10);
```



La función **plot** nos permite indicar explicitamente un rango para el eje y, con lo cual es posible restringir el despliegue a lo largo de este eje. Este rango puede expresarse como ***"y=c..d",*** o bien simplemente como ***"c..d"***, sin colocar la variable y.

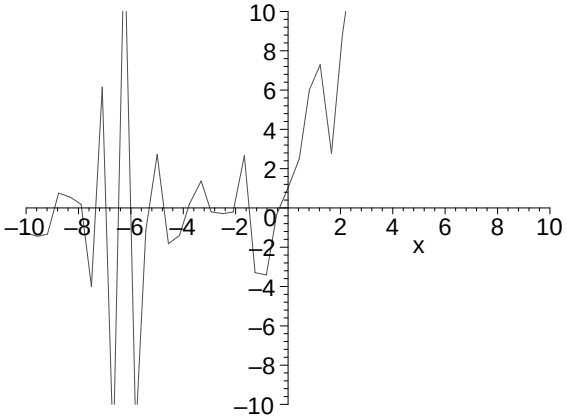
Apliquemos esto último a la gráfica anterior:

```
> plot(g(x), x=-10..10, y=-50..50);
```



Esta es una forma en la cual puede apreciarse mejor el verdadero aspecto de la gráfica; incluso, si restringimos aun más el rango para el eje ***y***, obtenemos la siguiente gráfica más detallada:

```
> plot(g(x), x=-10..10, -10..10);
```



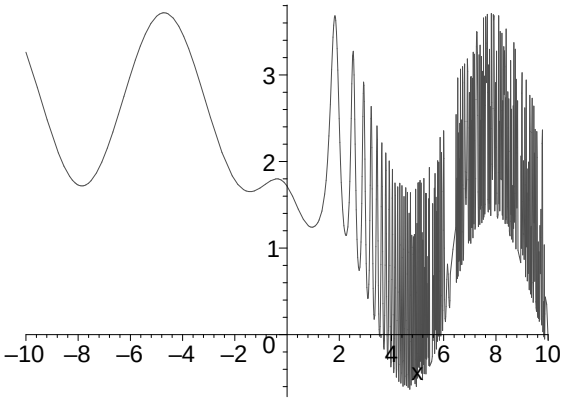
Otra forma de apreciar el aspecto real de la gráfica es utilizando el botón 1:1, para deplegarla a escala normal (este botón aparece en el menú contextual al seleccionar una región gráfica). Es importante señalar que, de forma predeterminada, las gráficas aparecen con una cierta distorsión, esto se debe a que son desplegadas dentro de una región cuadrada y por lo tanto al generarlas Maple las ajusta para que se adapten a esta región en la cual se mostraran al usuario.

También debe tomarse en cuenta que en algunas gráficas es necesario restringir no el eje ***y***, sino el eje ***x*** para obtener un despliegue más fiel de la función. Por ejemplo:

```
> k := x -> sin(x) + exp(cos(exp(x)));
```

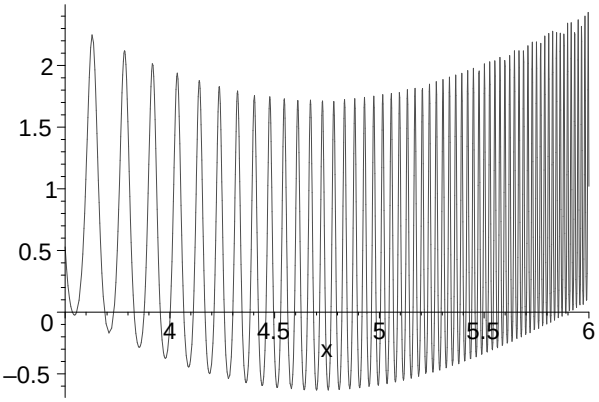
$$k := x \rightarrow \sin(x) + e^{\cos(e^x)}$$

```
> plot(k(x), x=-10..10);
```



Si restringimos el rango en x a **"3.5 .. 6"** podemos apreciar mejor el comportamiento de esta función en este intervalo.

```
> plot(k(x), x=3.5..6);
```



En general, para desplegar una gráfica, utilizando por ejemplo **plot(h(x), x=a..b)**, Maple procede a grandes rasgos de la siguiente forma:

- Primero, toma el intervalo **"x = a..b"** y lo particiona en un conjunto de puntos (el valor predeterminado es 50). A esta partición se le conoce con el nombre de ***"malla"***.
- A continuación, se evalua la función **h(x)** en cada uno de los puntos de la malla.
- Con estos datos se forman un conjunto de parejas ordenadas de la forma:

  $[x_i,\ \mathrm{f}(x_i)]$, donde x_i es un punto de la malla.
- A continuación se crea una estructura que contiene los puntos obtenidos, el color con el que se desplegará la gráfica, el título de ésta (en caso de que se deseé colocar alguno), el tipo de los ejes, el ancho y estilo de la linea utilizada, etc.

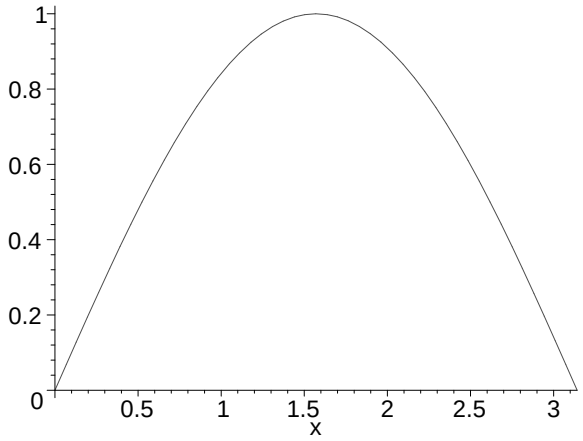
- Finalmente, Maple toma esta estructura y la despliega en la pantalla, uniendo los puntos contenidos en dicha estructura con el tipo de linea adecuada y colocando el resto de los elementos especificados, como los ejes, títulos y demás.

Esta estructura, generada por Maple para crear una gráfica, puede ser asignada a una variable para su despliegue o manipulación posterior. Veamos el siguiente ejemplo:

```
> g1 := plot(sin(x), x=0..Pi):
```

Posteriomente podemos desplegar esta gráfica con la instrucción **display** del paquete **plots**, o simplemente invocandola en la linea de comandos.

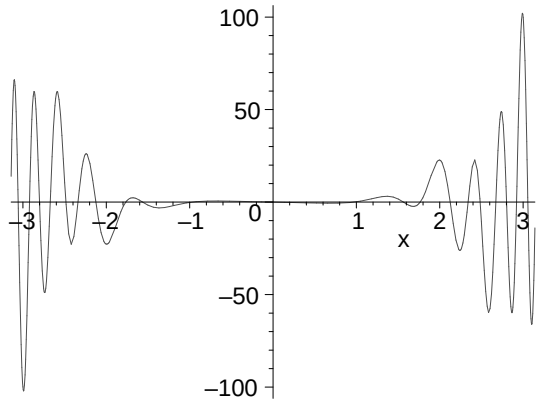
```
> plots[display](g1);
```



Nota: cuando se asigna una gráfica a una variable es recomendable usar como terminador **":"**, de lo contrario se desplegará toda la estructura generada para el despliegue.

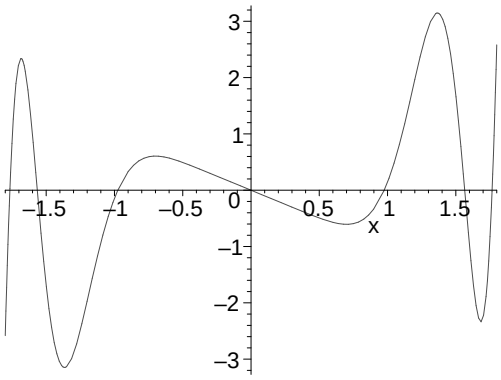
Veamos el siguiente ejemplo:

```
> plot(x^4*sin(x^3) - x^3*cos(x^2) + x^2*sin(x) - x, x=-Pi..Pi);
```



Nótese que en esta gráfica tampoco se aprecia el comportamiento de la función alrededor de $x = 0$. Podemos restringir nuevamente el rango en este eje para poder apreciar mejor la parte que no es clara:

```
> plot(x^4*sin(x^3) - x^3*cos(x^2) + x^2*sin(x) - x, x=-1.8..1.8);
```



Generalmente las gráficas generadas por Maple son desplegadas en la misma hoja de trabajo, esto puede modificarse de tal forma que sean mostradas en una ventana independiente; para ello solicitamos la opción **Preferences** del menú **File**; la cual desplegará la ventana que se muestra a continuación. en ella solicitamos la ficha **Plotting** y en la región **Plot Display** seleccionamos **Window** (esta opción se encuentra colocada en **Inline** de manera predeterminada), como se muestra en la Figura 1.

Figura 1: Ventana de preferencias

A una gráfica en dos dimensiones se le puede asignar una leyenda, la cual se despliega en la parte inferior de la misma. Esta leyenda puede asignarse seleccionando la gráfica y solicitando la opción **Edit Leyend** en el menú **Leyend**. En la ventana mostrada en la Figura 1, en la región **Display Legends**, se puede indicar a Maple si estas leyendas deben o no mostrarse, el mismo resultado puede obtenerse si seleccionamos la gráfica y solicitamos la opción **Show Leyend** en el menú **Leyend**. Otra manera en la que podemos hacer estas operaciones es oprimiendo el botón derecho del ratón sobre la gráfica, esto desplegará un menú contextual, en él seleccionamos el submenú **Leyend**.

En esta misma ventana, en la región **Print Quality**, podemos establecer la calidad con la que deben imprimirse las gráficas cuando imprimimos el documento.

2. Principales opciones aplicables a gráficas en dos dimensiones

Existe un conjunto de opciones que pueden aplicarse tanto a **plot** como a la mayoria de las instrucciones que generan gráficas en dos dimensiones, con el fin de obtener mejores despliegues. A continuación describiremos algunas de ellas.

2.1. Restricción del rango vertical

Esta opción fue tratada en parte en la sección anterior, es bastante útil al desplegar funciones cuya gráfica se dispara verticalmente, ya que nos permite ***"recortar"*** dicha gráfica de tal forma que podamos apreciar mejor su comportamiento en un área menor a la mostrada de manera predeterminada.

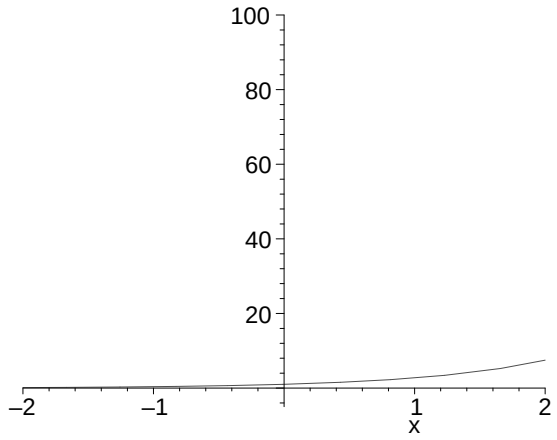
Además de lo visto anteriormente, **plot** proporciona la opción **view** que nos permite especificar los rangos horizontal y vertical para el despliegue. Su sintaxis es:

plot(f, x, view=[x1..x2, y1..y2])

Los rangos proporcinados a través de esta opción determinan el área del plano que se deberá mostrar en la gráfica.

Veamos el siguiente ejemplo:

```
> plot(exp(x), x, view=[-2..2, -5..100]);
```



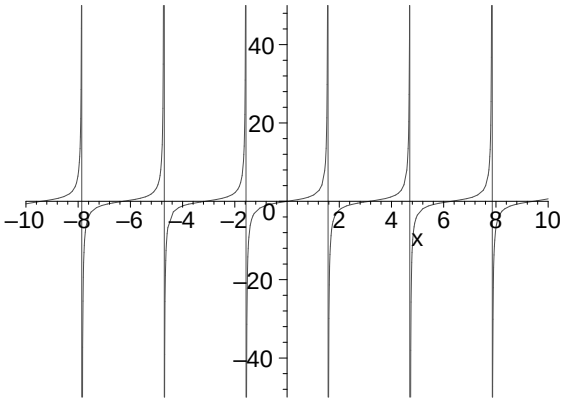
Los valores predeterminados que toma esta opción son **[-10 .. 10, fmin .. fmax]**. Donde **fmin** y **fmax** son los valores mínimo y máximo, respectivamente, obtenidos al evaluar la función en los puntos del primer rango.

2.2. Desplegar solo las secciones en las cuales la función es continua

Esto puede hacerse utilizando la opción **discont=true**, con la cual Maple despliega la gráfica solo en los intervalos del rango en los cuales la función es continua. Si no se incluye esta opción, las discontinuidades pueden aparecer en forma de lineas verticales.

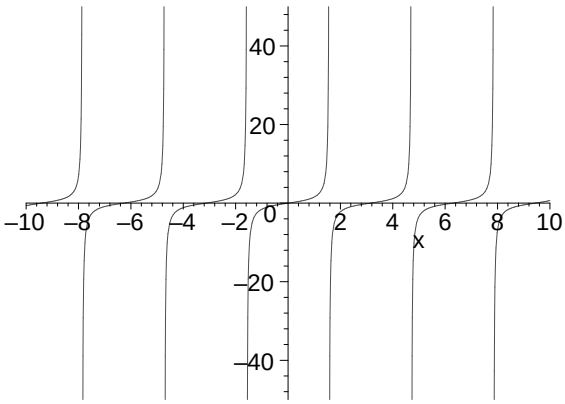
Para ejemplificar esto veamos el siguiente caso:

```
> plot(tan(x), x=-10..10, -50..50);
```



En esta gráfica se muestran algunas lineas verticales en los puntos donde existen discontinuidades. Agregaremos la opción **discont=true**:

```
> plot(tan(x), x=-10..10, -50..50, discont=true);
```



Nótese como en esta última gráfica no se despliegan las lineas verticales, las cuales corresponden precisamente a los puntos en los cuales la función es discontinua. Para desplegar esta gráfica, al incluir la opción antes mencionada, Maple utiliza la instrucción **discont**, la cual determina las discontinuidades de una función dada, y a continuación divide el intervalo a graficar en un conjunto de subintervalos en los cuales la función es continua.

La sintaxis de la instrucción **discont** es:

discont(f,x)

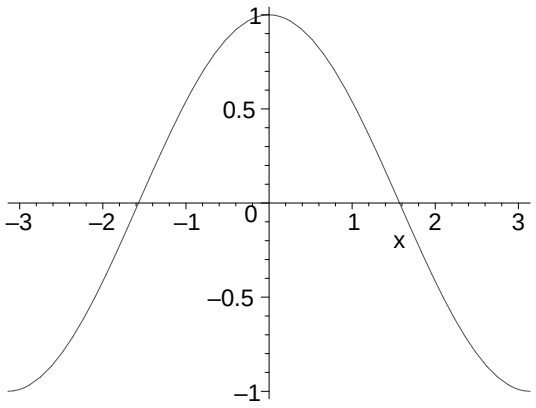
donde **f** es una función (o expresión) y $\boldsymbol{x}$ es la variable de la cual depende. Consultesé su ***página de ayuda***.

2.3. Colocar diferentes tipos de ejes

La opción **axes** nos permite colocar diferentes tipos de ejes en el despliegue de una gráfica en dos dimensiones. Los valores que puede tomar esta opción son:

- **normal** Este es el valor predeterminado. Muestra los ejes de la siguiente forma:

```
> plot(cos(x), x=-Pi..Pi, axes=normal);
```



Esta opción también puede aplicarse directamente del menú contextual para regiones gráficas. Al seleccionar una de estas regiones, en este menú aparecen una serie de botones que permiten modificar el aspecto de las gráficas, uno de ellos es el botón  , el cual (en el caso de las gráficas en dos dimensiones) despliega los ejes de la misma forma que al aplicar la opción **axes**=**normal**. Otra forma en la que puede aplicarse es colocando el ratón sobre la grafica y oprimiendo el botón derecho, en el menú contextual que aparece seleccionamos el submenú **Axes**, en el cual podemos seleccionar los mismos tipos de ejes que se describen en esta sección.

A continuación se muestran otras opciones aplicables:

- **frame**. Despliega el eje $\boldsymbol{x}$ en el borde inferior de la gráfica y el eje $\boldsymbol{y}$ en el borde izquierdo. Esta opción también puede ser aplicada directamente desde el menú contextual para regiones gráficas, a través del botón  .
- **boxed**. Muestra los ejes en forma de un marco que contiene a la gráfica. Se puede aplicar también mediante el botón  , de la barra contextual.
- **none**. Suprime el despliegue de los ejes en la gráfica. También puede ser aplicada por medio del botón  , de la barra contextual.

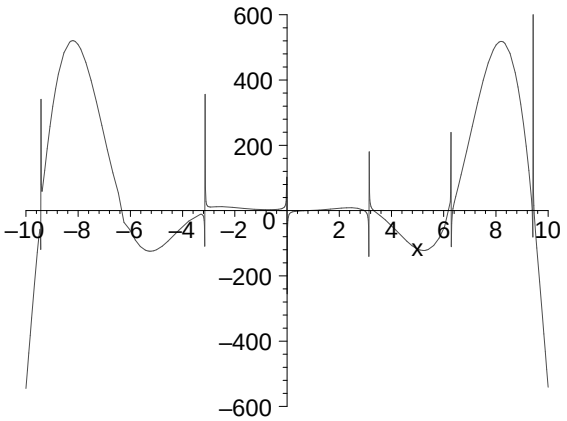
2.4. Modificación de la malla

Las funciones para despliegue de gráficas en dos dimensiones soportan dos opciones que nos permiten modificar la malla utilizada, estas son **numpoints** y **resolution**. La primera nos permite especificar el número mínimo de puntos que formarán la malla (el valor predeterminado es 50) y la segunda nos permite especificar el número máximo (por omisión 200).

Al generar la estructura de una gráfica, Maple evalua la función en los puntos de la malla y despues los une con una linea para presentar el despliegue. Pero, ¿qué sucede si la función tiene una discontinuidad en un punto que no está considerado en la malla?

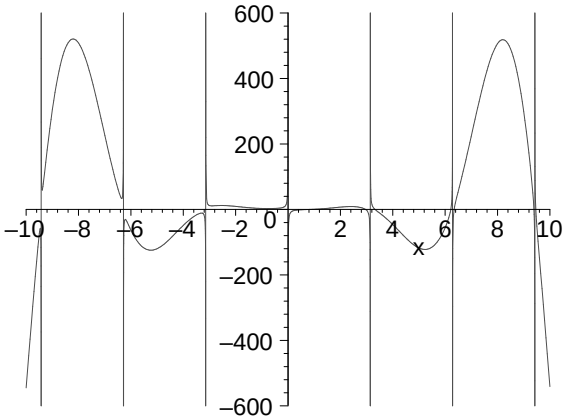
La función **plot** (entre otras) tiene la capacidad de utilizar más puntos en la malla en las regiones en las cuales se encuentran las discontinuidades. Sin embargo, en funciones que oscilan demasiado esto no es suficiente, en estos casos es conveniente incrementar el número de puntos en la malla. Por ejemplo:

```
> plot(sin(x)*x^3 - 1/sin(Pi - x) + 1, x=-10..10, -600..600);
```



Ahora incrementaremos el número de puntos de la malla a 400:

```
> plot(sin(x)*x^3 - 1/sin(Pi - x) + 1, x=-10..10, -600..600,
> numpoints=400);
```



Al utilizar una malla más fina la gráfica aparece más detallada, incluso pueden apreciarse discontinuidades que anteriormente no se mostraban.

2.5. Asignación de colores a la gráfica

Por omisión Maple despliega las gráficas en color rojo (excepto cuando se despliegan varias funciones), pero puede utilizarse la opción ***"color = c"*** para especificar uno diferente. Esta constante puede asumir los valores predefinidos que se muestran en la tabla 1.

aquamarine	cyan	khaky	pink	turquoise
black	gold	magenta	plum	violet
blue	green	maroon	red	wheat
brown	gray	navy	sienna	white
coral	grey	orange	tan	yellow

Cuadro 1: Colores predefinidos para gráficas

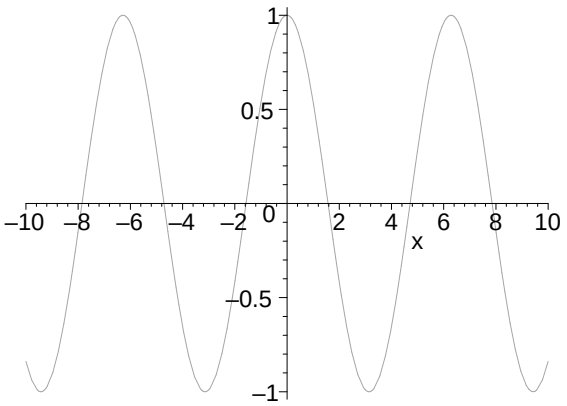
También es posible expresar el color en formato ***RGB***, utilizando la función:

COLOR(RGB, r, v, a)

donde ***"r"***, ***"v"*** y ***"a"*** especifican las componentes rojo, verde y azul, respectivamente, del color deseado (estos componentes solo pueden tomar valores entre cero y uno).

Por ejemplo veamos la siguiente gráfica:

```
> plot(cos(x), x, color=COLOR(RGB, .737, .737, 0));
```



Una forma más conveniente de definir colores utilizando el modo **RGB** es a través de la definición de ***"macros"***. Por ejemplo:

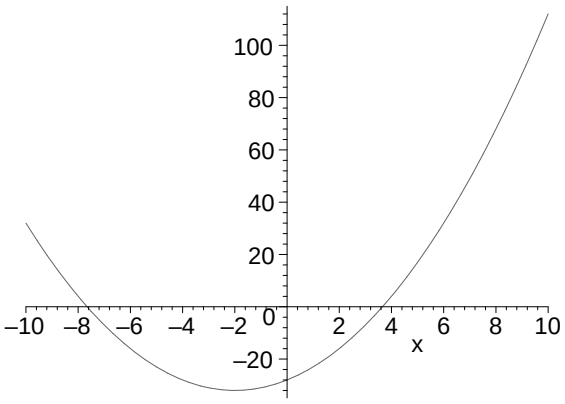
macro(c1=COLOR(RGB, C-R, C-G, C-B))

Esta instrucción define una macro de nombre **c1**, la cual representa el color dado por las componentes RGB recibidas como segundo, tercero y cuarto argumentos. Este nuevo color creado puede ser usado de la siguiente forma, en el despliegue de una gráfica:

```
> macro(c1 = COLOR(RGB, .437, .437, 0));
```

c1

```
> plot(x^2 + 4*x - 28, x, color=c1);
```



Otras opciones utiles, aplicables a este tipo de gráficas incluyen:

- **Colocar etiquetas en los ejes**. Esto puede hacerse a través de la opción **labels=[etqx, etqy]**. Donde **"etqx"** y **"etqy"** son las etiquetas para el eje x y el eje **y** respectivamente. Por ejemplo, la siguiente opción:

  labels=[velocidad, tiempo]

  desplegará la etiqueta **"velocidad"** en el eje x y **"tiempo"** en el eje y. Estas etiquetas deben estar dadas en forma de cadenas en caso de que contengan espacios en blanco. Puede utilizarse también la opción **labelfont=[f, e, t]**, para especificar un tipo de fuente, estilo y tamaño de las etiquetas de los ejes (vease la siguiente opción para una descripción de los argumentos **f**, **e** y t).

- **Utilizar diferentes tipos de fuentes**. Maple nos permite utilizar varios fuentes, estilos y tamaños en los textos desplegados en una gráfica. Esto puede aplicarse con la opción **font=[f, e, t]**. Donde **"f"** es el tipo de fuente, el cual puede ser **TIMES**, **COURIER**, **HELVETICA** o **SYMBOL**; **"e"** determina el estilo, el cual varía dependiendo de la fuente. Para **TIMES** los estilos disponibles son **ROMAN**, **BOLD**, **ITALIC** y **BOLDITALIC**; para **HELVETICA** y **COURIER** el estilo puede ser **BOLD**, **OBLIQUE** o **BOLDOBLIQUE**, o bien puede ser omitido. La fuente **SYMBOL** no acepta ningun tipo de estilo. Finalmente el argumento **"t"** indica el tamaño de la fuente en puntos. Por ejemplo, si se aplican las opciones:

  labels=[velocidad, 'eje y'], font=[TIMES, ITALIC, 14]

  a una gráfica en dos dimensiones, se desplegarán las etiquetas **"velocidad"** y **"eje y"**, utilizando la fuente **TIMES**, en estilo **ITALIC** y con un tamaño de 14 puntos.

- **Modificar la fuente para los ejes.** Podemos especificar el tipo de fuente a ser utilizado para los números que se despliegan a lo largo de los ejes, a través de la opción **axesfont=[f, e, t]**. Donde **"f"**, **"e"** y **"t"** son como en la opción **font**.

- **Colocar un título a la gráfica.** Esto puede hacerse a través de la opción **title=ct**, donde **"ct"** es la cadena de caracteres que se colocará como título. Además, podemos incluir también la opción **titlefont=[f, e, t]**, para determinar el tipo de fuente, estilo y tamaño con el que se desplegará este título. Los argumentos de **titlefont** son como en la opción **font**.

- **Utilizar lineas de diferente grosor y estilo.** El grosor de las lineas puede ser dado a través de la opción **thickness=g**, donde **"g"** indica el grosor y puede tomar valores enteros entre cero y tres. El valor predeterminado es cero, el cual corresponde al tipo de linea más delgada utilizada en el despliegue de este tipo de gráficas, el grosor de la linea aumenta conforme al valor de **"g"**.

  Otra opción aplicable es **linestyle=n**, donde **"n"** indica el estilo de la linea. Los valores cero y uno generan lineas continuas, mientras que los valores más grandes generan lineas punteadas de diferentes formas.

Todas las opciones descritas anteriormente pueden ser aplicadas a cualquier tipo de gráfica en dos dimensiones, tanto las generadas por **plot** como las generadas por otras funciones.

Para obtener una lista completa de las opciones aplicabes a funciones en dos dimensiones consúltese la hoja de ayuda **?plot[options]**, o bien **?plot,options**.

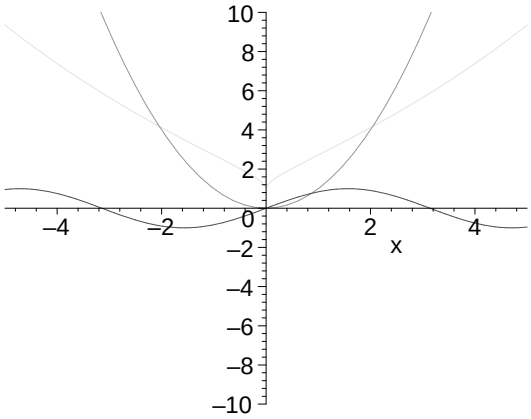
3. Despliegue de multiples funciones explicitas

Maple nos permite desplegar las gráficas de varias funciones en un mismo despliegue, colocando éstas como argumento en forma de conjunto, es decir:

plot({f1, f2, f3,...}, x=a..b, opciones)

Veamos el siguiente ejemplo:

```
> plot({sin(x), x^2, exp(sqrt(abs(x)))}, x=-5..5, -10..10);
```



Nótese que en este caso Maple asigna automáticamente un color diferente a cada gráfica. Además, no es necesario que dichas funciones dependan de la misma variable al definirlas. Por ejemplo:

```
> f1 := x -> sin(x);
```

$$f1 := \sin$$

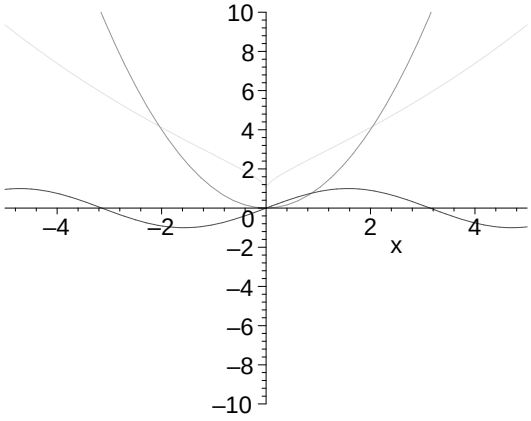
```
> f2 := r -> r^2;
```

$$f2 := r \to r^2$$

```
> f3 := s -> exp(sqrt(abs(s)));
```

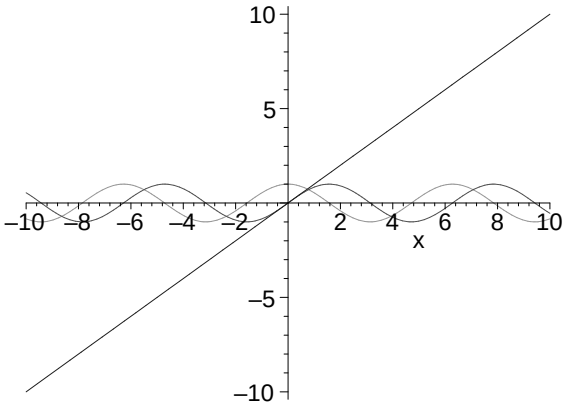
$$f3 := s \to e^{\sqrt{|s|}}$$

```
> plot({f1(x), f2(x), f3(x)}, x=-5..5, -10..10);
```



Cuando se despliegan varias funciones en una misma gráfica, se puede asignar un color a cada función mediante la opción **color**, de la siguiente forma:

```
> plot({sin(x), cos(x), x}, x, color=[blue, brown, green]);
```



Otra manera de graficar varias funciones en un mismo despliegue es por medio de la función **display**, esto se abordará más adelante.

4. Gráficas de puntos

La instrucción **plot**, además de funciones, también nos permite generar gráficas de puntos. Éstos deben estar dados en forma de una lista de parejas, de la siguiente manera:

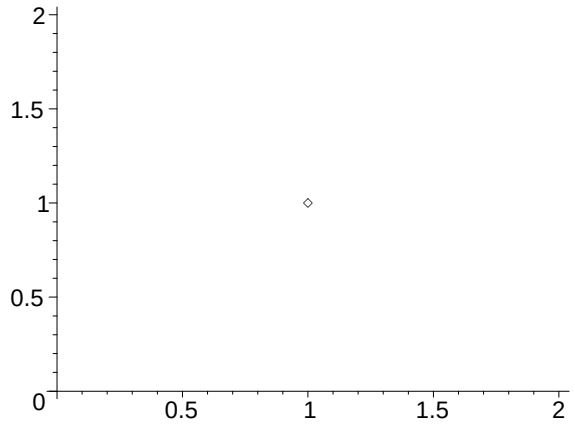
plot([[x1, y1], [x2, y2], [x3, y3], ...])

Esta función toma cada uno de los puntos en la lista y genera una gráfica uniendolos con una linea. Para que estos puntos puedan apreciarse es necesario incluir la opciones **style=point**, lo cual indica a Maple que se trata de una gráfica de puntos. También se puede incluir **symbol=ts**, para especificar el tipo de símbolo que se usará para mostrar cada punto, y la opción ***symbolsize=n*** que permite especificar el tamaño del símbolo (por omisión 10). Como tipo de símbolo puede usarse:

BOX, CROSS, CIRCLE, POINT, DIAMOND

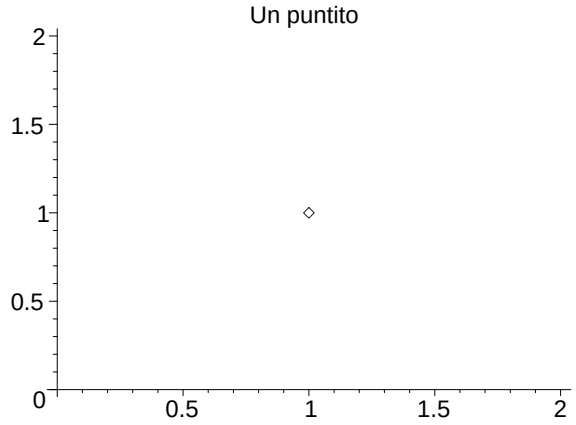
Por ejemplo:

```
> plot([[1, 1]], style=point, symbol=DIAMOND, symbolsize=15);
```



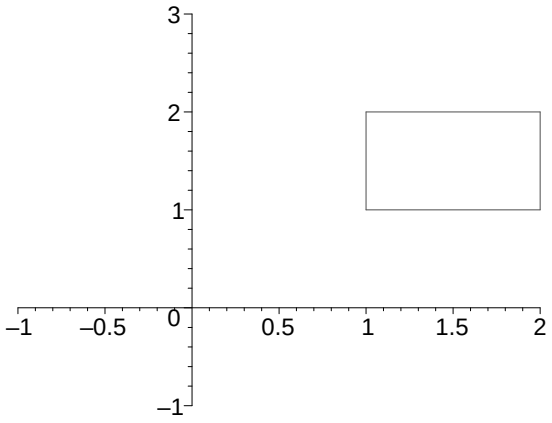
A estas gráficas podemos aplicarles cualquiera de las opciones descritas anteriormente. Por ejemplo :

```
> plot([[1, 1]], style=point, symbol=DIAMOND, symbolsize=18,
> color=blue, title=‘Un puntito‘);
```



Veamos ahora la siguiente gráfica.

```
> plot([[1, 1], [1, 2], [2, 2], [2, 1], [1, 1]], view=[-1..2,
> -1..3]);
```



En este caso, al colocar los puntos en forma de una lista estos son unidos entre sí por medio de una linea (nótese que no estamos inluyendo la opción **style**=**point**).

5. Funciones paramétricas

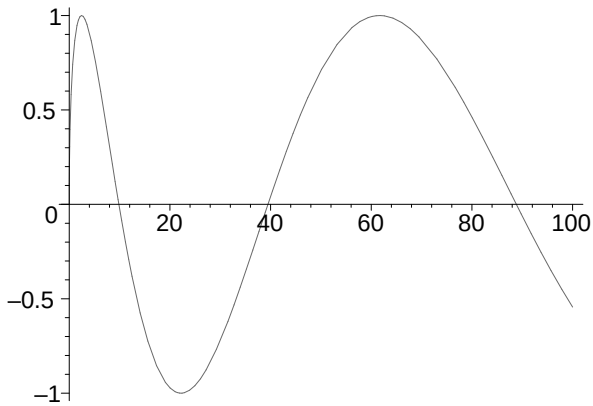
Otra de las capacidades de la función **plot** es que nos permite desplegar gráficas de funciones dadas en forma paramétrica. La sintaxis es:

plot([r(x), t(x), rango], opciones)

Donde **"r(x)"** y **"t(x)"** son las componentes paramétricas de la función a graficar. A este tipo de gráficas se les pueden aplicar todas las opciones vistas previamente.

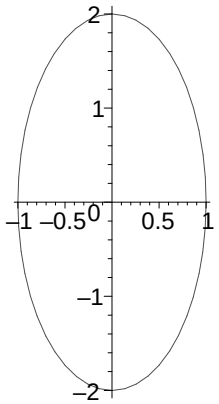
Veamos un ejemplo:

```
> plot([t^2, sin(t), t=0..10], color=magenta);
```



De la misma manera podemos desplegar otras gráficas:

```
> plot([sin(t), 2*cos(t), t=0..2*Pi], scaling=constrained);
```



Una forma en la que podemos graficar una función dada en la forma **"y=f(x)"**, es por medio de la parametrización **"[x, f(x)]"**.

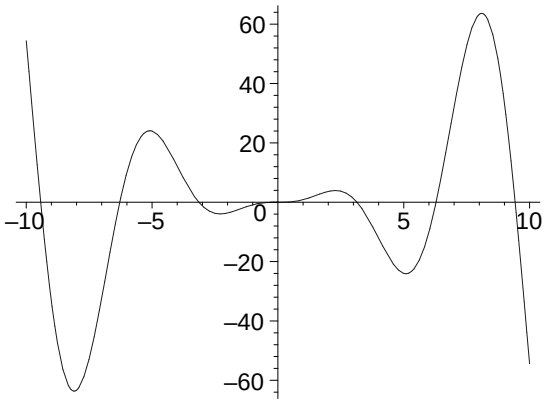
En el siguiente ejemplo deplegaremos la gráfica de " $\sin(x)\,x^2$", parametrizada como:

"$[x, \sin(x)\,x^2]$**"**

```
> f := x -> sin(x)*x^2;
```

$$f := x \to \sin(x)\,x^2$$

```
> plot([x, f(x), x=-10..10], color=blue);
```



6. Coordenadas polares

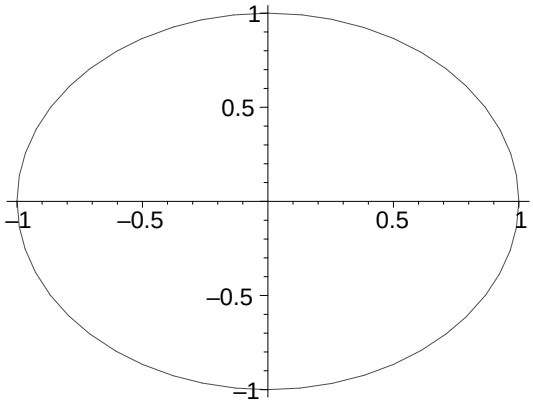
En Maple, las gráficas en coordenadas polares son manejadas como gráficas de funciones paramétricas, a las cuales se les agrega la opción **coords=polar**.

Su sintaxis es la misma que en el caso de las funciones paramétricas, es decir:

plot([r(t), theta(t), rango], coords=polar, otras_opciones)

En este caso **"r(t)"** representa el radio y **"theta(t)"** el ángulo de la función. Por ejemplo:

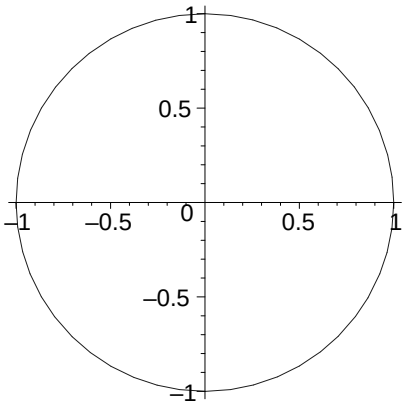
```
> plot([1, t, t=0..2*Pi], coords=polar);
```



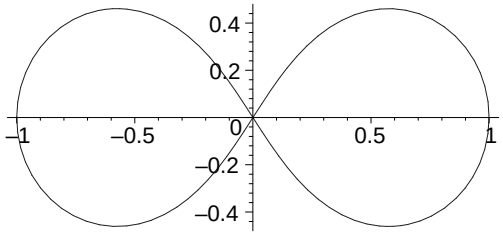
Esta gráfica corresponde a un círculo de radio ***"r(t)=1"***.

Para poder apreciar el efecto de esta opción, comparense las siguientes gráficas:

```
> plot([sin(t), cos(t), t=0..2*Pi], scaling=constrained,
> color=blue);
```



```
> plot([sin(t), cos(t), t=0..2*Pi], coords=polar, scaling=constrained,
> color=blue);
```



7. Gráficas en dos dimensiones con el paquete Plots

Este es uno de los paquetes más robustos de Maple, proporciona un conjunto de funciones especializadas para la generación de diversos tipos de gráficas, como veremos a continuación.

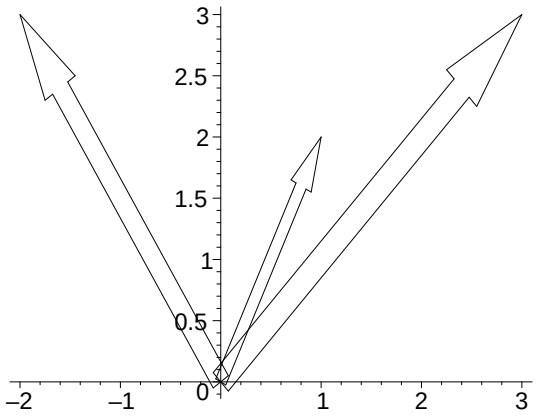
Antes de hacer uso de estas funciones debemos cargar el paquete con la instrucción: **with(plots)**.

7.1. Gráficas de vectores

Este tipo de gráficas pueden generarse por medio de la instrucción **arrow(v)**. Donde **"v"** es un vector, un conjunto de vectores o una lista de vectores, cada uno de los cuales puede ser dado en la forma: **[x, y]** o **<x, y>**. También puede ser generado por medio de las funciones: **array([x, y])**, o bien, **Vector([x, y])**.

Veamos un ejemplo:

```
> arrow({<-2, 3>, [1, 2], <3, 3>});
```



Algunas de las opciones que se pueden aplicar a este tipo de gráficas son:

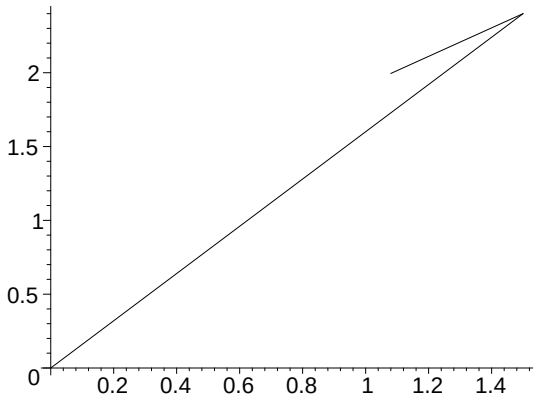
- **shape = harpoon, arrow, double_arrow**.

  Esta opción determina el tipo de flecha que será utilizada, el valor predeterminado es **double_arrow** (existe otra opción conocida como **cylindrical_arrow**, la cual solo puede ser usada en el despligue de vectores en tres dimensiones). Por ejemplo:

```
> v1 := array(1..2, [1.5, 2.4]);
```

$$v1 := [1{,}5,\ 2{,}4]$$

```
> arrow(v1, shape=harpoon);
```



- **width = n**. Esta opción determina el ancho del vector. Por ejemplo:

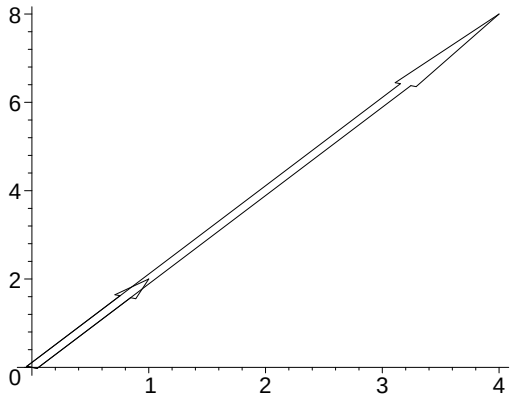
```
> v2 := Vector([1, 2]);
```

$$v2 := \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

```
> v3 := Vector([4, 8]);
```

$$v3 := \begin{bmatrix} 4 \\ 8 \end{bmatrix}$$

```
> arrow({v2, v3}, shape=double_arrow, width=0.1);
```



7.2. Gráficas de complejos

Otra de las funciones incluidas en el paquete **plots** es **complexplot**. Esta función permite obtener gráficas en dos dimensiones de una o más funciones, expresiones, procedimientos, funciones paramétricas o listas de numeros complejos. Su sintaxis es:

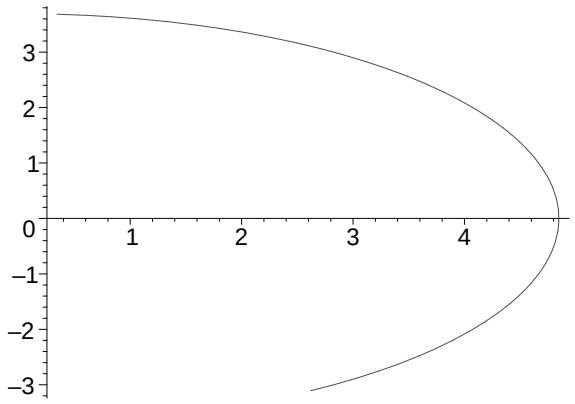
complexplot(f, x=a..b)

Donde **f** es una función de una variable, la cual representa un mapeo de los reales en los complejos; ***"x=a..b"*** especifica el rango en el cual se gráficara. Por ejemplo:

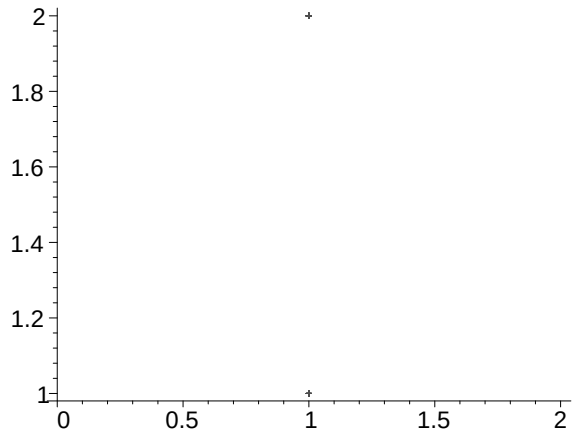
```
> fc := x -> Pi*(cos(x + I));
```

$$fc := x \to \pi \cos(x + I)$$

```
> complexplot(fc(x), x=-1.5..1);
```




```
> complexplot([1 + 2*I, 3 + 4*Pi*i, sin(Pi/2) + I], x=-10..10,
> style=point, symbolsize=18);
```



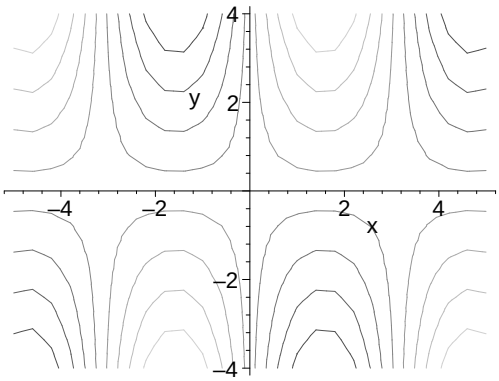
7.3. Gráficas de contornos

Este tipo de gráficas pueden ser generadas por medio de la instrucción:

contourplot(f, x=a..b, y=c..d)

Donde **f** es la función o expresión a gráficar (de dos variables), mientras que los intervalos incluidos determinan un área rectangular en la que se desplegará la gráfica. Por ejemplo:

```
> contourplot(sin(x)*y, x=-5..5, y=-4..4);
```



7.4. Mapas de densidad

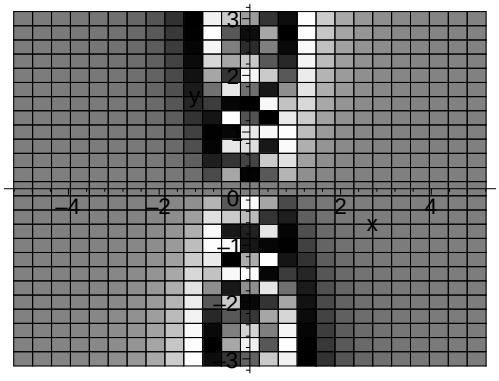
Éstos pueden obtenerse por medio de la instrucción **densityplot**. Su sintaxis es:

densityplot(f, x=a..b, y=c..d)

densityplot(f, a..b, c..d)

Donde **f** es una función o expresión (de dos variables) y los intervalos determinan un área rectangular en la que se desplegará la gráfica. Por ejemplo:

```
> densityplot(sin(y/x^3), x=-5..5, y=-3..3);
```



7.5. Despliegue de gráficas con estructuras diferentes

Este paquete nos proporciona la función **display**, la cual nos permite mostrar varias funciones en un mismo despliegue, sin importar que tengan estructuras diferentes. Su sintaxis es:

display([g1, g2, g3, ...])

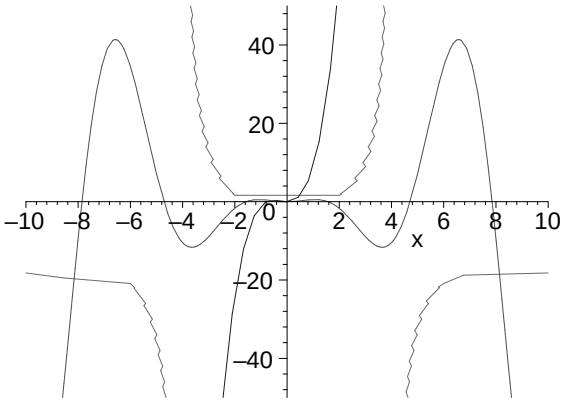
Donde ***"g1, g2, g3, ..."*** son estructuras gráficas previamente creadas. Por ejemplo:

```
> g1 := plot(x^2*cos(x), x, -50..50, color=brown):

> g2 := plot(4*x^2 + 5*x^3, x, -50..50, color=blue):

> g3 := implicitplot(x - 1/x = y*sin(x), x=-50..50, y=-50..50,
> color=magenta):

> plots[display]([g1, g2, g3]);
```



7.6. Gráficas de funciones implícitas

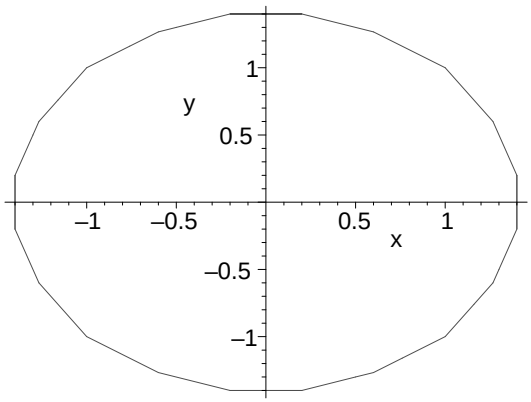
Otra función contenida en **plots** es **implicitplot**, la cual nos permite obtener gráficas de funciones cuya regla de correspondencia está dada de manera implícita. Su sintaxis es:

implicitplot(f, x=a..b, y=c..d)

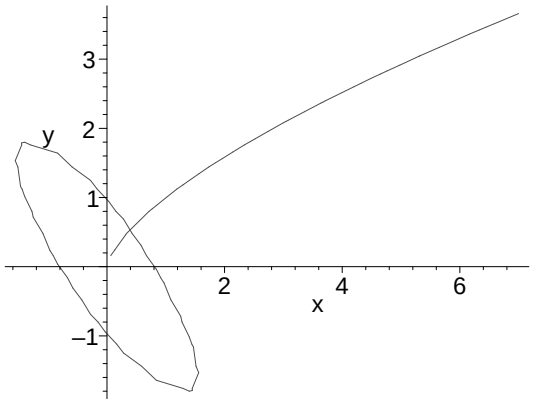
donde **f** es la función o expresión en términos de dos variables (pueden ser varias expresiones o funciones dadas en forma de conjunto), mientras que **x** y **y** determinan los intervalos para los que se desplegará la gráfica.

Veamos los siguientes ejemplos:

```
> implicitplot(x^2 + y^2 = 2, x=-5..5, y=-5..5);
```




```
> implicitplot({x = y*sqrt(y), (x + y)^2 = cos(x)}, x=-7..7,
> y=-4..4);
```



7.7. Gráficas de desigualdades

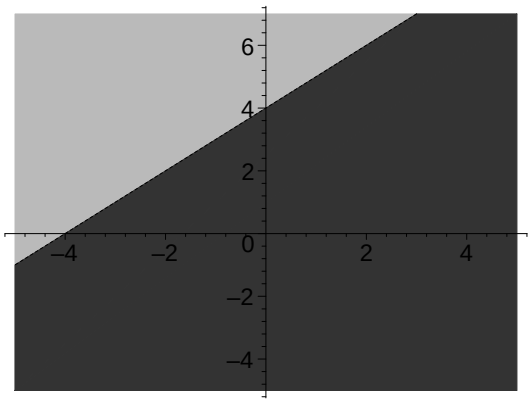
Este tipo de gráficas pueden generarse con la función **inequal**. Su sintaxis es:

inequal(exp, x=a..b, y=c..d)

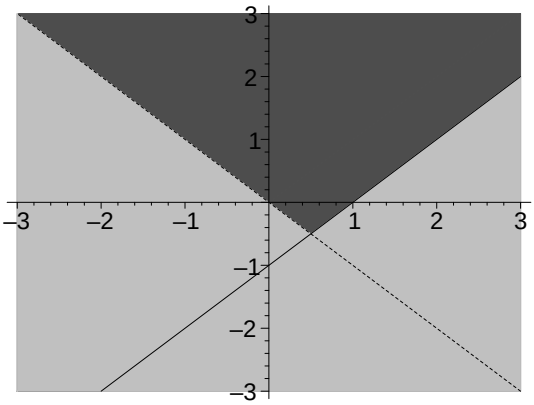
Donde **exp** es una ecuación, inecuación o un conjunto de éstas. Los intervalos incluidos determinan un área rectangular en la que se desplegará la gráfica.

Veamos los siguientes ejemplos:

```
> inequal((x + 1) < (y - 3), x=-5..5, y=-5..7);
```




```
> inequal({x + y > 0, x - y <= 1}, x=-3..3, y=-3..3,
> optionsfeasible=(color=red), optionsexcluded=(color=grey) );
```



En este último ejemplo, la opción: **"optionsfeasible"**, determina el color con el cual se desplegará el área donde las desigualdades se cumplen; mientras que la opción **"optionsexcluded"** determina el color para el área donde las desigualdades no se cumplen. Por omisión, esta función despliega en color claro el área en que las desigualdades se cumple y en color obscuro el área en la que no se cumplen.

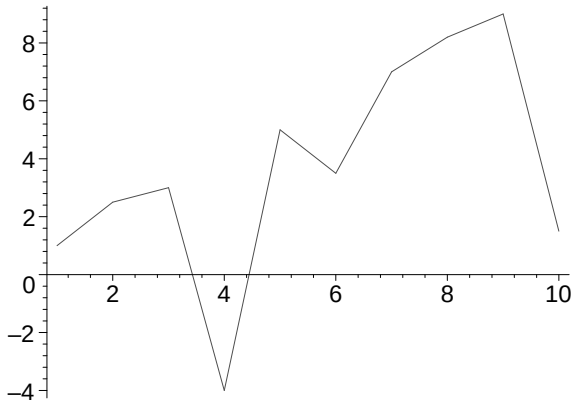
7.8. Gráficas de listas

La función **listplot**, nos permite graficar listas de números. Su sintaxis es:

listplot(L)

Donde **L** es una lista de datos numéricos o puntos. Por ejemplo:

```
> listplot([1, 2.5, 3, -4, 5, 3.5, 7, 8.2, 9, 1.5], color=red);
```



En este ejemplo puede notarse que la función **listplot** toma una lista de la forma:

[a, b, c, d, e, f, g, h, ...]

y a partir de ella crea un conjunto de puntos:

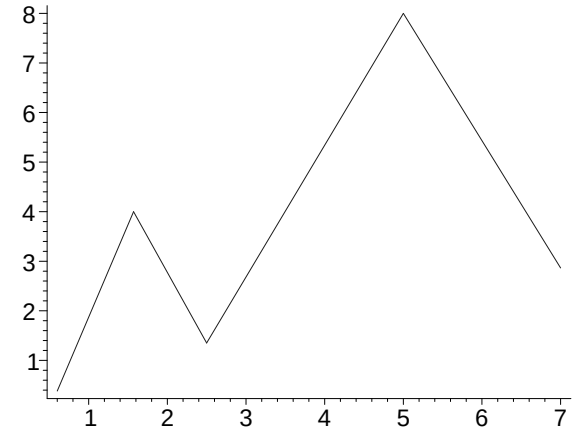
(1, a), (2, b), (3, c), (4, d), (5, e) ...

los cuales une con una linea en el orden que fueron colocados en la lista. Esta última también puede darse en la forma:

[[x1, y1], [x2, y2], [x3, y3], [x4, y4], ...]

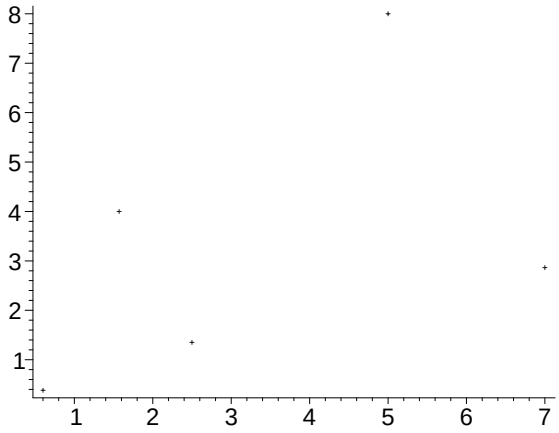
Veamos un ejemplo:

```
> listplot([[.6, sin(Pi/8)], [Pi/2, 4], [2.5, exp(0.3)], [5, 8],
> [7, 9/Pi]], color=blue);
```



Si deseamos que los puntos sean graficados como tales, debemos incluir la opción **style=point**.

```
> listplot([[.6, sin(Pi/8)], [Pi/2, 4], [2.5, exp(0.3)], [5, 8],
> [7, 9/Pi]], style=point, symbolsize=14, color=blue);
```



7.9. Gráficas de soluciones de ecuaciones diferenciales

La función **odeplot** del paquete **plots**, nos permite graficar una solución particular para una ecuación diferencial o un sistema de ecuaciones diferenciales, obtenida mediante **dsolve**; siempre que esta solución se encuentre en forma numérica. Su sintaxis es:

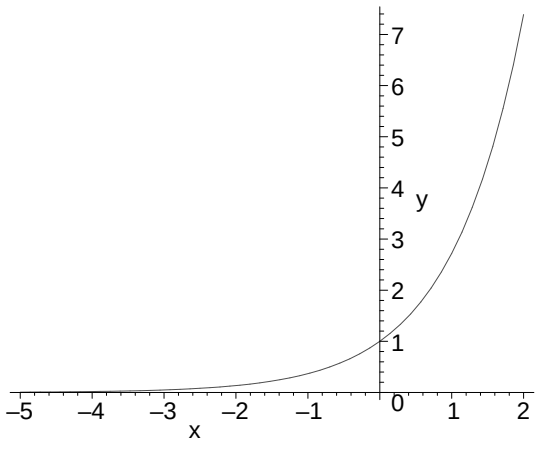
odeplot(sol)

Donde **sol** es una solución particular para una ecuación diferencial (o un sistema de ecuaciones diferenciales), obtenida de manera numérica; es decir, mediante la instrucción: **dsolve(..., numeric)**.

Por ejemplo, graficaremos la solución de la ecuación: $\frac{d}{dx}\,\mathrm{y}(x) = \mathrm{y}(x)$, para $y(0) = 1$.

```
> s := dsolve({D(y)(x) = y(x), y(0)=1}, type=numeric, range=-5..2);
```

```
> odeplot(s);
```



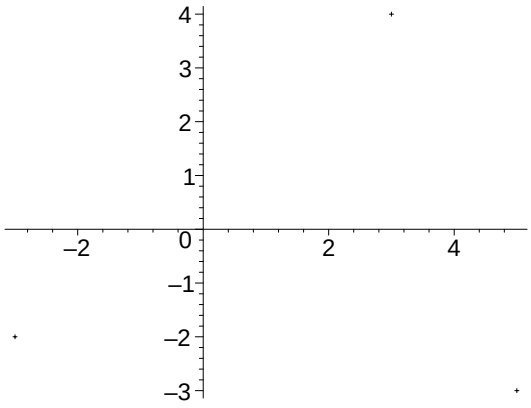
7.10. Gráficas de puntos

Otra de las funciones contenidas dentro de este paquete es **pointplot**, la cual nos permite generar gráficas de puntos. Su sintaxis es:

pointplot(pts, opciones)

Donde **pts** es una lista o conjunto de puntos. Por ejemplo:

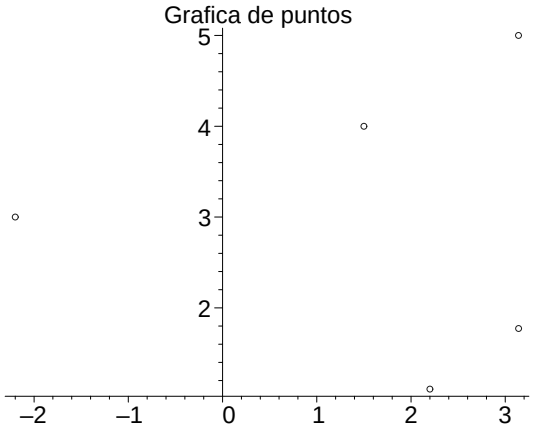
```
> pointplot({[3, 4], [5, -3], [-3, -2]}, symbolsize=14);
```



pointplot es similar a **listplot**, pero a diferencia de ésta última, **pointplot** no une los puntos recibidos con una linea de manera predeterminada, para que lo haga debe incluirse la opción **style**=**line**.

Esta función también es similar a **plot**, invocada con la opción **style**=**point**. Veamos otro ejemplo:

```
> pointplot({[-2.2, 3], [1.5, 4], [3.14, 5], [Pi, sqrt(Pi)],
> [2.2, exp(0.1)]}, symbol=circle, symbolsize=15, color=blue,
> title=‘Grafica de puntos‘);
```



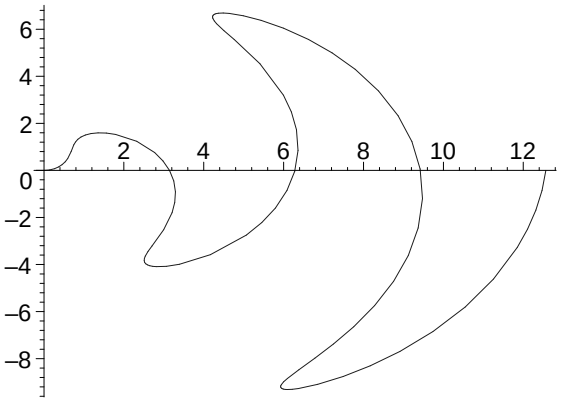
7.11. Gráficas en coordenadas polares

Las funciones dadas en coordenadas polares pueden ser graficadas por medio de la instrucción **polarplot**. La sintaxis es:

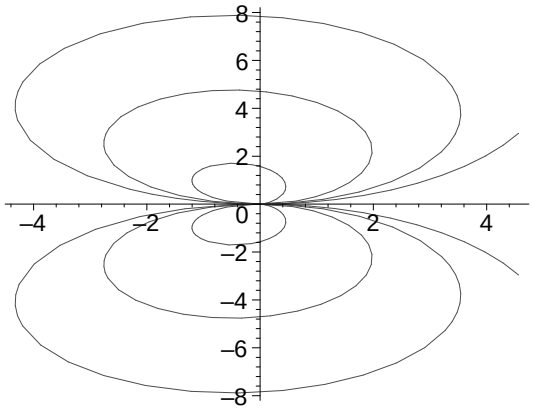
polarplot(expr, opciones)

Donde **expr** es una curva o un conjunto de curvas dadas en coordenadas polares. Las opciones aplicables a esta función son las mismas de **plot**. Por ejemplo:

```
> polarplot([t, sin(t), t=0..4*Pi], color=blue);
```




```
> polarplot(x*sin(x), x=-10..10);
```

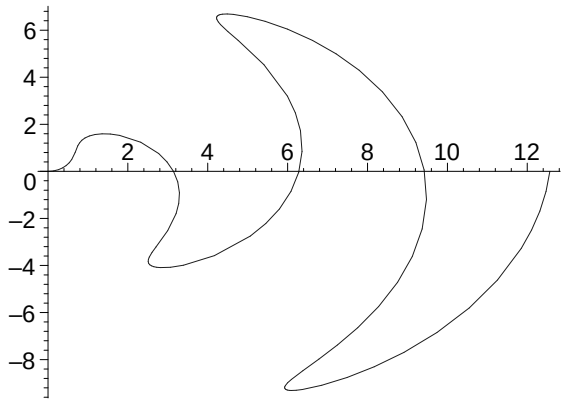


En estos ejemplos, cuando la función a graficar está dada en la forma **"[f(x), g(x)]"**, la primera componente es considerada el radio y la segunda el ángulo. En cambio, si la función está dada en la forma **"h(x)"** (como en la segunda gráfica), la variable es considerada el ángulo y la función el radio.

Este tipo de gráficas también pueden desplegarse por medio de **plot**, incluyendo la opción **coords=polar**.

Veamos el siguiente ejemplo:

```
> plot([t, sin(t), t=0..4*Pi], coords=polar, color=blue);
```



7.12. Gráficas de polígonos

La función **polygonplot** del paquete **plots**, nos permite desplegar gráficas de poligonos. La sintaxis es:

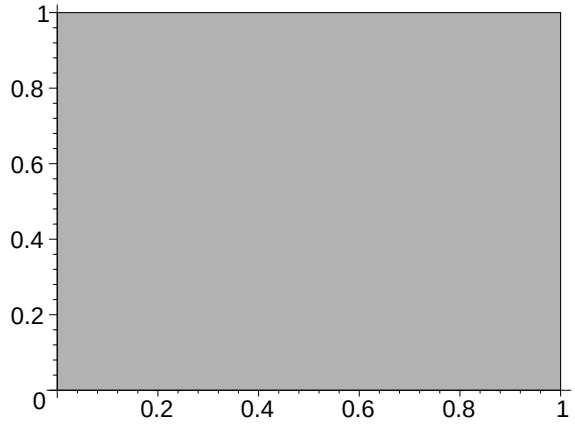
polygonplot(pts, opciones)

Donde **pts** son los puntos de un polígono o de un conjunto de polígonos (cada uno debe ser dado como una lista de puntos). Las opciones aplicables son las mismas de **plot**. Veamos algunos ejemplos:

```
> pol1 := [[0, 0], [1, 0], [1, 1], [0, 1]];
```

$$pol1 := [[0, 0], [1, 0], [1, 1], [0, 1]]$$

```
> polygonplot(pol1, color=cyan);
```



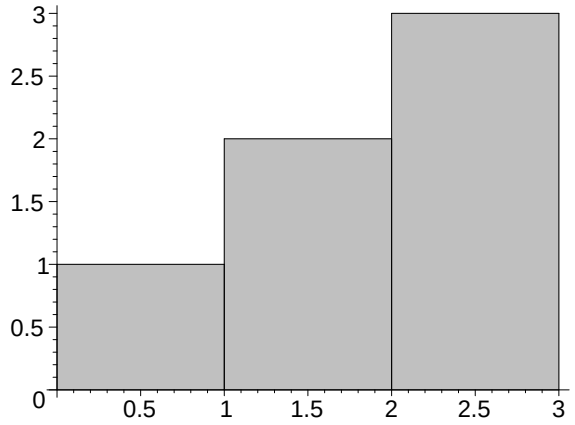

```
> pol2 := [[1, 0], [2, 0], [2, 2], [1, 2]];
```

$$pol2 := [[1, 0], [2, 0], [2, 2], [1, 2]]$$

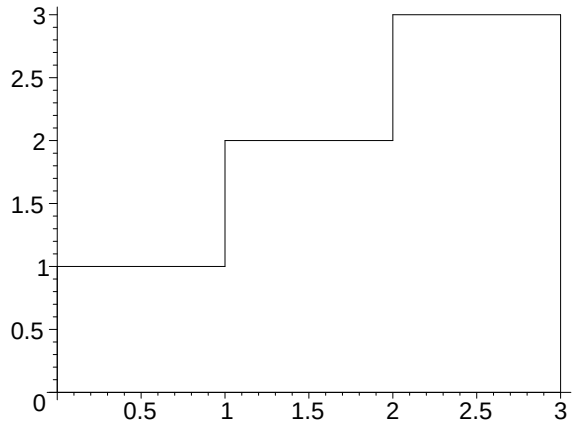
```
> pol3 := [[2, 0], [3, 0], [3, 3], [2, 3]];
```

$$pol3 := [[2, 0], [3, 0], [3, 3], [2, 3]]$$

```
> polygonplot([pol1, pol2, pol3], color=gray);
```




```
> polygonplot([[0, 0], [0, 1], [1, 1], [1, 2], [2, 2], [2, 3],
> [3, 3], [3, 0]]);
```



7.13. Gráficas de texto

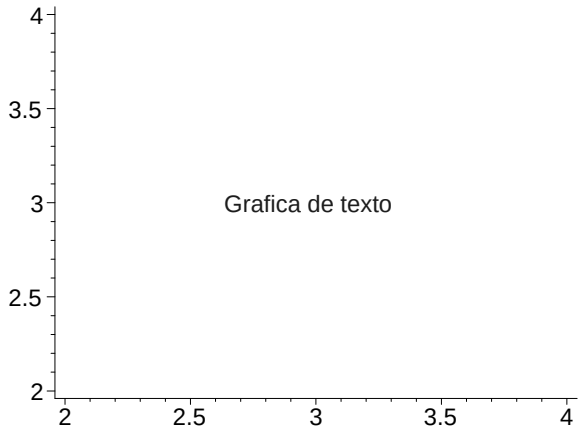
Otra función útil es **textplot**, por medio de la cual se pueden crear gráficas de cadenas de texto en dos dimensiones. Su sintaxis es:

textplot(expr, opciones)

Donde **expr** es una lista que contiene las coordenadas de un punto en dos dimensiones, así como la cadena de texto que será colocada en este punto.

Las opciones aplicables a esta función son basicamente las mismas de **plot**. Por ejemplo:

```
> textplot([3, 3, 'Grafica de texto'], color=blue);
```



En una misma gráfica se pueden colocar varias cadenas de texto, expresandolas como una lista de listas o un conjunto de listas. Por ejemplo:

```
> txt1 := [-2, 2, 'Texto 1'];
```

$$txt1 := [-2,\ 2,\ Texto\ 1]$$

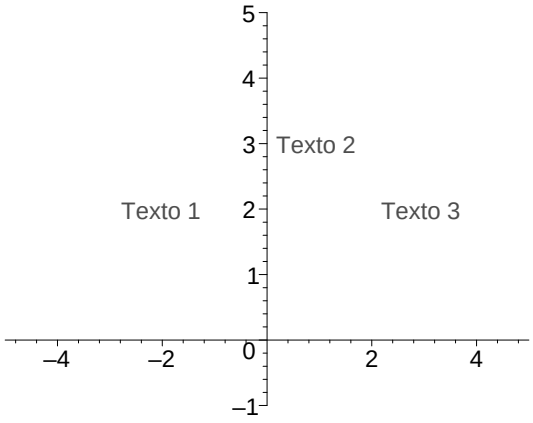
```
> txt2 := [1, 3, 'Texto 2'];
```

$$txt2 := [1,\ 3,\ Texto\ 2]$$

```
> txt3 := [3, 2, 'Texto 3'];
```

$$txt3 := [3,\ 2,\ Texto\ 3]$$

```
> textplot({txt1, txt2, txt3}, view=[-5..5,-1..5], color=red);
```

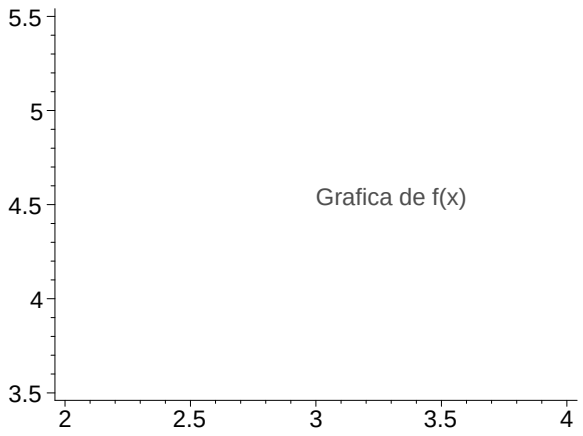


De manera predeterminada, **textplot** coloca la cadena de texto centrada con respecto al punto recibido como argumento (tanto horizontal como verticalmente). Esto puede modificarse por medio de la opción:

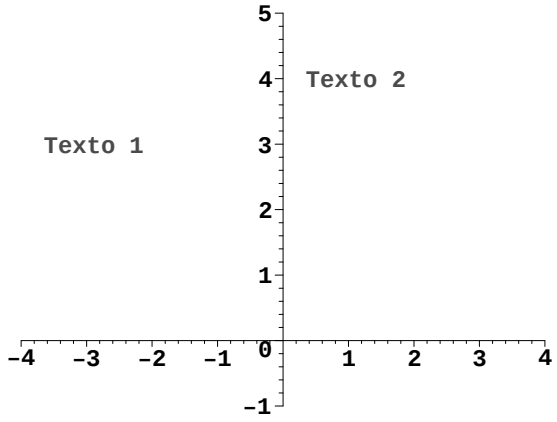
align=opciones de alineación

Las opciones de alineación pueden ser **ABOVE**, **BELOW**, **RIGHT** o **LEFT**, las cuales pueden usarse individualmente o en combinación. Por ejemplo:

```
> textplot([3, 4.5, 'Grafica de f(x)'], align={ABOVE, RIGHT}, color=brown);
```




```
> textplot([[-2, 3, 'Texto 1'], [2, 4, 'Texto 2']], align=LEFT,
> color=maroon, font=[COURIER, BOLD, 10], view=[-4..4, -1..5]);
```



Este tipo de despliegues pueden ser combinados con gráficas de funciones por medio de la instrucción **display**. Veamos un ejemplo.

Primero definimos las siguientes funciones:

```
> f1 := x -> sin(x);
```

$$f1 := \sin$$

```
> f2 := x -> x^3/(20*Pi^2);
```

$$f2 := x \rightarrow \frac{1}{20}\,\frac{x^3}{\pi^2}$$

```
> f3 := x -> -x^3*sin(x)/(20*Pi^2);
```

$$f3 := x \rightarrow -\frac{1}{20}\,\frac{x^3 \sin(x)}{\pi^2}$$

A continuación generamos sus gráficas:

```
> g1 := plot(f1(x), x=-10..10, color=blue):

> g2 := plot(f2(x), x=-10..10, color=brown):

> g3 := plot(f3(x), x=-10..10, color=magenta):
```

Despues generamos una gráfica de texto para cada función:

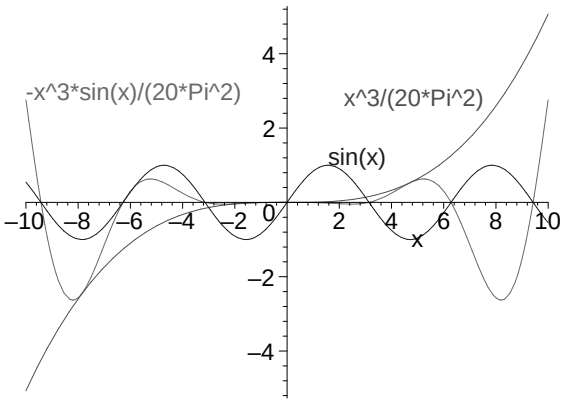
```
> tx1 := textplot([Pi/2, f1(Pi/2), `sin(x)`], color=blue,
> align={ABOVE, RIGHT}):

> tx2 := textplot([8, f2(8), `x^3/(20*Pi^2)`], color=brown,
> align={ABOVE, LEFT}):

> tx3 := textplot([-10, f3(-10), `-x^3*sin(x)/(20*Pi^2)`],
> color=magenta,
> align={ABOVE,RIGHT}):
```

Finalmente desplegamos todos estos elementos con **display**:

```
> display([g1, g2, g3, tx1, tx2, tx3]);
```



Existen otras funciones para despliegue de gráficas en dos dimensiones, incluidas en el paquete **plots**. Consulte la ***página de ayuda*** mediante la instrucción: **?plots** para obtener información sobre ellas.

7.14. Manipulación de la región gráfica

En las secciones anteriores se describio la utilidad de algunos de los botones que aparecen en la barra contextual cuando se selecciona una región gráfica. Otros de los elementos que aparecen en esta barra son:

- 2.08, 2.24. En esta región se despliegan un par de coordenadas, cada vez que se seleccciona un punto en la gráfica por medio del ratón.

-  . Estos dos botones permiten modificar el despliegue de la gráfica seleccionada, utilizando una linea continua o un conjunto de puntos, respectivamente. Estos botones tienen el mismo efecto sobre la gráfica que las opciones **style=LINE** y **style=POINT**.

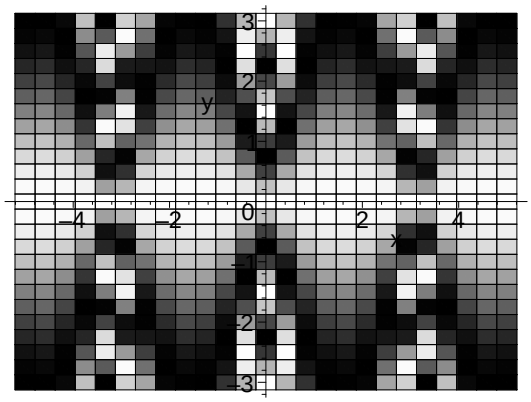
-  . Estos dos botones solo son aplicables a gráficas en las que se despliegan superficies. El primero de estos permite desplegar la gráfica con una malla, mientras que el segundo elimina esta malla.

  La acción de estos botones es equivalente a utilizar las opciones:

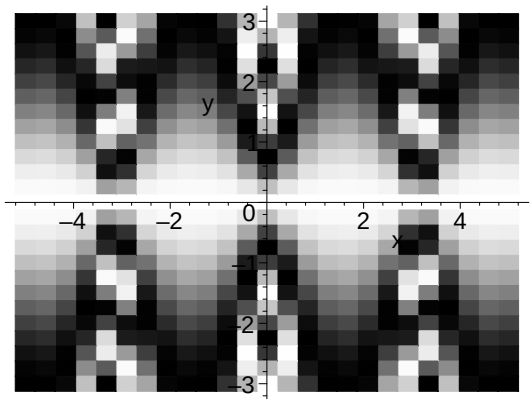
  style=PATCH y **style=PATCHNOGRID**

  Un caso en el que se pueden aplicar estos botones en gráficas de dos dimensiones es en los mapas de densidades. Para ejemplificar estas opciones comparense las siguientes gráficas:

```
> plots[densityplot](cos(y/sin(x)), x=-5..5, y=-3..3, style=PATCH);
```




```
> plots[densityplot](cos(y/sin(x)), x=-5..5, y=-3..3, style=PATCHNOGRID);
```



Muchas de las opciones descritas en este capítulo pueden ser aplicadas directamente a las gráficas por medio del menú contextual, para ello colocamos el apuntador del ratón sobre una gráfica y oprimimos el botón derecho, esto desplegrará el menú contextual que se muestra en la Figura 2.

Figura 2: Menú contextual para gráficas

Por medio de éste podemos modificar en la gráfica, por ejemplo, el estilo de las lineas, el tipo de simbólos usados (por ejemplo para gráficas de puntos) y el tipo de ejes, entre otras cosas. Además, la opción **Export As**, nos permite exportar la gráfica como imagen a un archivo, en formatos tales como: GIF, JPEG, EPS (Postscript encapsulado), entre otros.

7.15. Creación de gráficas de forma interactiva

Esta versión de Maple nos ofrece una opción para la creación de gráficas en dos dimensiones de manera interactiva, haciendo uso de una nueva herramienta conocida como **Interactive Plot Builder**. Ésta puede ser invocada por medio de la instrucción **interactive** del paquete **plots**.

Para ejemplificar esto generaremos la gráfica de la función **sin(sqrt(abs(x))**; para ello invocamos la función **interactive** como se muestra a continuación:

```
> plots[interactive](sin(sqrt(abs(x))));
```

esta instrucción desplegará la ventana del **Interactive Plot Builder** que se muestra en la Figura 3.

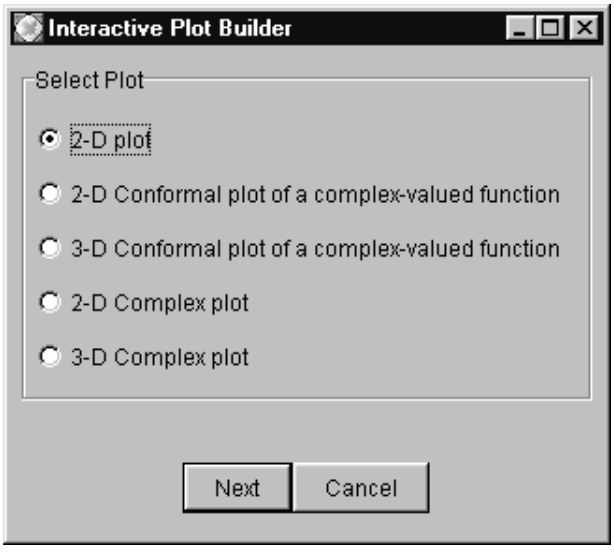


Figura 3: Ventana inicial del Interactive Plot Builder

Seleccionamos **2-D plot** en esta ventana y oprimimos el botón **Next**. A continuación aparecerá la ventana de la Figura 4.

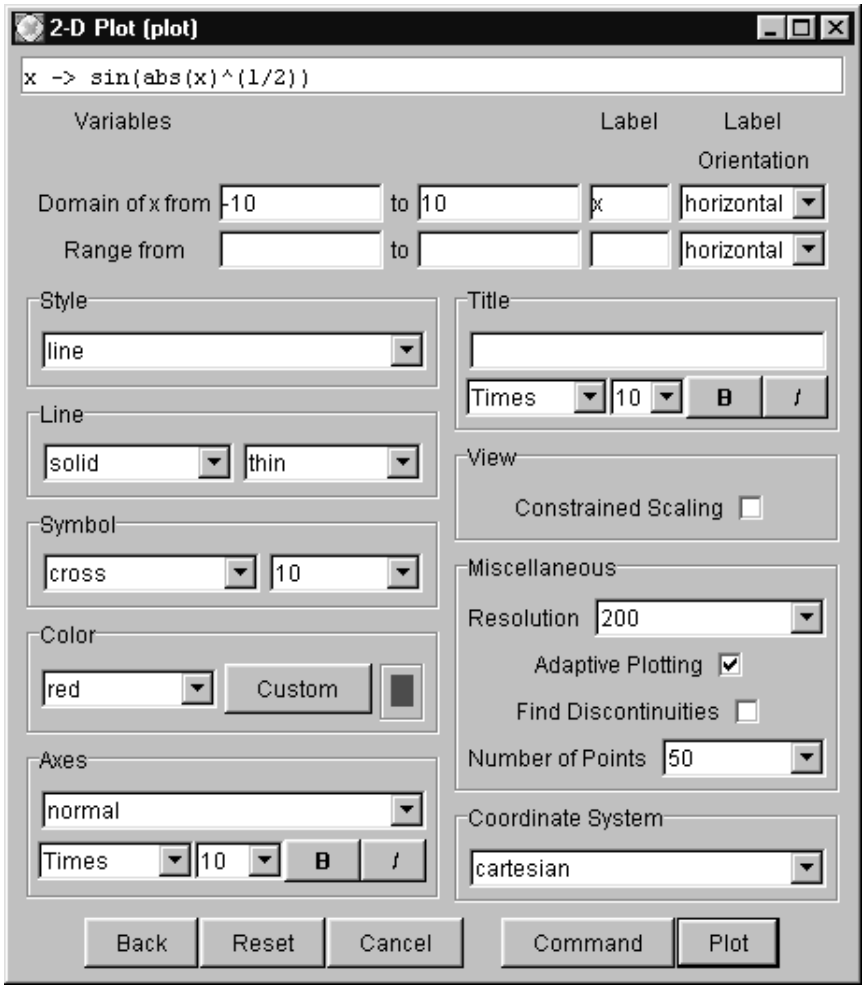


Figura 4: Interactive Plot Builder

En ésta última podemos especificar todas las opciones que deberán aplicarse a nuestra gráfica, por ejemplo: el rango para el eje y (en la sección **Range from .. to**), el color de la gráfica, el sistema de coordenadas a usar, el tipo de linea, el tipo de ejes, etcétera. De hecho, usando esta herramienta podemos aplicar todas las opciones que se describieron en este capítulo. Al final, oprimimos el botón **Plot**, con lo cual se desplegará la gráfica.

Acerca de la ventana del **Interactive Plot Builder**, ésta nos permite generar otro tipo de gráficas, por ejemplo gráficas de complejos en dos y tres dimensiones, como puede verse en la Figura 3.

7.16. Creación de gráficas a partir de una expresión de salida

Otra opción más que podemos usar para generar una gráfica es seleccionando, en una región de salida, una expresión o parte de ésta y oprimiendo el botón derecho del ratón; en el menú contextual que aparece seleccionamos el submenú **Plots** y en éste podemos seleccionar las opciones **Plot Builder** o **2D-Plot** (también nos proporciona la opción **3D-Plot** para despliegue de gráficas en tres dimensiones), como puede verse en la Figura 5.

La primera opción desplegará la ventana descrita en la sección anterior, mientras que la segunda opción generará automaticamente una gráfica de la expresión seleccionada (este despliegue se genera invocando a

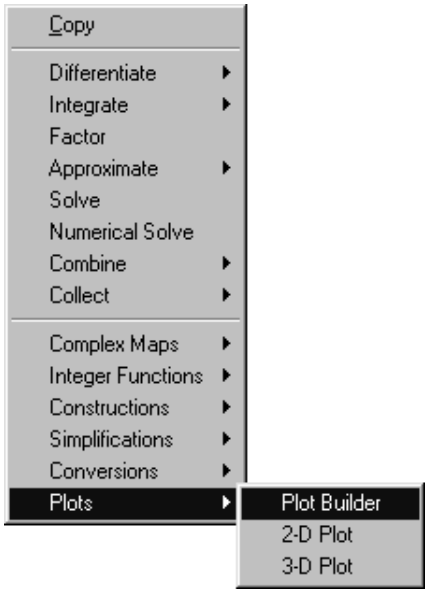


Figura 5: Menú contextual para generación de gráficas

la función **smartplot**).

Además de la forma anterior, esta versión de Maple nos proporciona una manera más para el despliegue de gráficas en dos dimensiones a partir de una expresión de salida (este mismo método puede ser usado para despliegue de gráficas en tres dimensiones). Seleccionamos, en el submenú **Plot** del menú **Insert** la opción **2-D**; esto insertará una región gráfica vacia en nuestra hoja de trabajo. Para poder desplegar la gráfica de una función en esta región insertada, seleccionamos, de una región de salida, una expresión o una parte de ésta y, por medio de las opciones del menú **Edit**, copiamos y pegamos esta expresión dentro de la región gráfica vacia; Maple automáticamente generará una gráfica de la expresión seleccionada. De esta misma forma, en el mismo despliegue podemos copiar y pegar varias expresiones, con lo cual obtendremos una gráfica que incluya todas éstas.