Taller de graficación

Ecuaciones diferenciales de primer orden

Agenda

- Python, Maxima y Matlab (Librerías y sintaxis).
- Graficación de funciones.
- Campo de pendientes (o direcciones)
- Solución simbólica de E.D.O de primer orden:
  -Usando Matlab
  -Usando Máxima

Introducción

Existen muchas herramientas (basadas en diferentes lenguajes de programación) que nos permiten solucionar de forma analítica o numérica ecuaciones diferenciales ordinarias(es decir de una sola variable dependiente) y hacer una representación grafica de las mismas.

En el curso nos centraremos en 3 de ellas; Maxima, Matlab y Python.

En este tutorial se explicara y se darán ejemplos de como realizar los siguientes tipos de graficas:

-Funciones explicitas (Cuando se puede despejar la variable dependiente)

-Funciones implícitas (Cuando es imposible despejar la variable dependiente)

-Campo de pendientes

Librerías y comandos con Python

Librerías:

Para representar gráficamente ecuaciones, ya sean implícitas o explicitas, o para graficar el campo de pendientes, es necesario importar las siguientes librerías:

- matplotlib.pyplot #Necesaria para ejecutar los comandos de graficación.
- Numpy #Facilita el uso de trabajo con arreglos y funciones trigonométricas (aunque tiene muchas mas funciones)

Comandos:

- Linspace(*inicio, fin,n*) #Genera un vector de n puntos
- plot(*var_independiente, var_dependiente*) #Para graficar funciones explicitas
- X,Y=Meshgrid (*x1,x2)* *#Región de graficación (malla o cuadricula)*
- *Contour(X,Y, f(x,y))* *#Para grafica de funciones implícitas*
- quiver(X,Y,U,V) #Campo de pendientes (se explica mejor mas adelante)

Librerías y comandos con Maxima

Librerías para máxima:

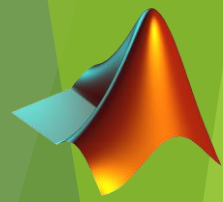
- Implicit_plot #se usa para hacer graficas de funciones implícitas
- Drawdf #Necesaria para realizar el campo e pendientes

Para cargar una librería en máxima se usa a instrucción *load(LIBRERIA)*

Comandos:

- plot2d($f(x)$,[x,x_i,x_f]) # Para graficar funciones explicitas.
- Implicit_plot(Igualdad,[x, x_i, x_f],[y, y_i, y_f]) #Para graficar funciones explicitas
- drawdf/plotdf(EDO) # Se usa para graficar el campo de pendientes (en algunas versiones de máxima y el IDE online esta instrucción no funciona, así que se recomienda usar la aplicación para Android).
- ode2(edo,y,x) #solución simbólica y general de una ecuación diferencial

Comandos con Matlab



En el caso de Matlab no tendremos que importar o cargar ninguna librería para resolver ecuaciones diferenciales o para graficar ecuaciones, el único inconveniente es que a diferencia de Python y máxima no hay ningún compilador online que nos permita ejecutar código en Matlab, así que es necesario tenerlo instalado en un pc.

Comandos:

- ezplot('ec',[xi,xf]) # Se usa para graficar expresiones en las cuales la variable dependiente se define como simbólica.
- plot(x,y) #Se usa para graficar expresiones en las que la variable independiente se define como un conjunto de puntos.
- [X,Y]=meshgrid(x,y) #Cuadricula necesaria para graficar una función implícita "x" y "y" son el conjunto de puntos que delimitan la cuadricula.
- contour($X, Y, f(X,Y), [c_1, c_2$ ...]) #Realiza la grafica de la función implícita (c_1, c_2 ..., son constantes que indican que curva se grafica den entre toda la familia de curvas posibles. Habrán tantas gráficas como constantes)
- dsolve(EDO,codiciones iniciales(opcional),x) $

Python

Para trabajar con Python tenemos varias opciones, entre esas las que recomiendo son:

-Anaconda, que posee 2 interfaces que son Spyder y Jupyter (requiere instalación)

-Cocalc, que permite programar en la nube con una interfaz similar a jupyter (no requiere instalación).

Este tutorial usara Cocalc para realizar algunos de ejemplos de graficación mediante el uso del lenguaje Python.

Como usar Cocalc:

- 1.Ingresar al sitio web https://cocalc.com/
- 2.Hacer clic en crear cuenta gratis.
- 3.Llenar el formulario o ingresar con cuenta Gmail (recomendado)
- 4. Crear un nuevo proyecto ->Darle un nombre -> Dar clic sobre él.
- 5.Crear un nuevo fichero -> seleccionar Jypyter notebook

Graficación de funciones explicitas con Python

Ejemplo 1. Graficar la siguientes funciones:

$$1.\ y = e^{-x^2} \qquad 2.\ y = e^{-x} sen(2\pi x) \qquad 3.\ y = \frac{xe^{-x}}{1-x^2}$$

Graficación de funciones explicitas con Python

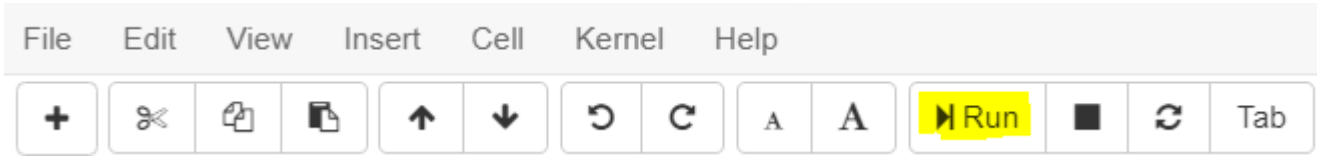
Ejemplo 1. Graficar la siguientes funciones:

$\mathbf{1.}\ y = e^{-x^2}$ $\quad 2.\ y = e^{-x}sen(2\pi x)$ $\quad 3.\ y = \frac{xe^{-x}}{1-x^2}$

1.Solución:

```
import numpy as np #importa la libreria numpy y la renombra para invocarla como np
import matplotlib.pyplot as plt #importa la libreria matplotlib y la renombra para invocarla como plt

x=np.linspace(-5,5,500) #se genera un vector de -5 hasta 5 con 500 puntos
y=np.exp(-x**2) #se define la función a graficar

fig=plt.figure(figsize=(6,4)) #se genera una figura de 8x6
plt.grid(True) #se activan las cuariculas
plt.plot(x,y,'b-',linewidth=2)  # Se grafica la y
plt.xlabel("x",size=18) #titulo del eje x
plt.ylabel("y",size=18) #titulo del eje y
plt.title(r"$e^{-x^2}$",size=18) #titulo del gráfico
plt.show() # esta instruccion es necesaria o no se mostrara la figura
```

Para ejecutar le damos clic a el botón Run, que se encuentra la parte superior



Graficación de funciones explicitas con Python

Ejemplo 1. Graficar la siguientes funciones:

$\mathbf{1.}\ y = e^{-x^2}$ $\quad 2.\ y = e^{-x}sen(2\pi x)$ $\quad 3.\ y = \frac{xe^{-x}}{1-x^2}$

1.Solución:

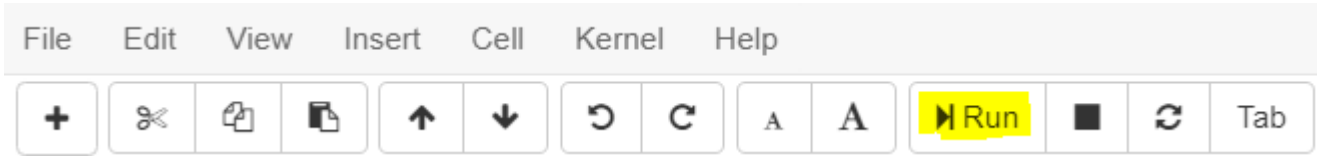
```
import numpy as np #importa la libreria numpy y la renombra para invocarla como np
import matplotlib.pyplot as plt #importa la libreria

x=np.linspace(-5,5,500) #se genera un vector de -5 h
y=np.exp(-x**2) #se define la función a graficar

fig=plt.figure(figsize=(6,4)) #se genera una figura
plt.grid(True) #se activan las cuariculas
plt.plot(x,y,'b-',linewidth=2)  # Se grafica la y
plt.xlabel("x",size=18) #titulo del eje x
plt.ylabel("y",size=18) #titulo del eje y
plt.title(r"$e^{-x^2}$",size=18) #titulo del gráfico
plt.show() # esta instruccion es necesaria o no se mostrara la figura
```

Se pueden importar las librerías sin renombrarlas, por ejemplo se puede importar numpy como:

from numpy import *

De esta forma no es necesario poner x=np.linspace(a,b,n), solo se colocaría x=linspace(a,b,n)

Para ejecutar le damos clic a el botón Run, que se encuentra la parte superior



Graficación de funciones explicitas con Python

Ejemplo 1. Graficar la siguientes funciones:

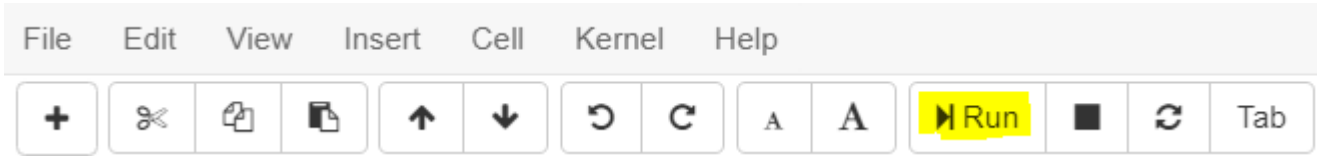
$\mathbf{1.}\ y = e^{-x^2}$ $\quad 2.\ y = e^{-x}sen(2\pi x)$ $\quad 3.\ y = \frac{xe^{-x}}{1-x^2}$

1.Solución:

```
import numpy as np #importa la libreria numpy y la renombra para invocarla como np
import matplotlib.pyplot as plt #importa la libreria m

x=np.linspace(-5,5,500) #se genera un vector de -5 has
y=np.exp(-x**2) #se define la función a graficar

fig=plt.figure(figsize=(6,4)) #se genera una figura de
plt.grid(True) #se activan las cuariculas
plt.plot(x,y,'b-',linewidth=2)  # Se grafica la y
plt.xlabel("x",size=18) #titulo del eje x
plt.ylabel("y",size=18) #titulo del eje y
plt.title(r"$e^{-x^2}$",size=18) #titulo del gráfico
plt.show() # esta instruccion es necesaria o no se mostrara la figura
```

La razón por la cual se importan las librerías de esta manera por que alguna veces diferentes librerías tienen funciones que tienen el mismo nombre.
Por ejemplos la math y numpy tienen una función que se llama exp()

Para ejecutar le damos clic a el botón Run, que se encuentra la parte superior

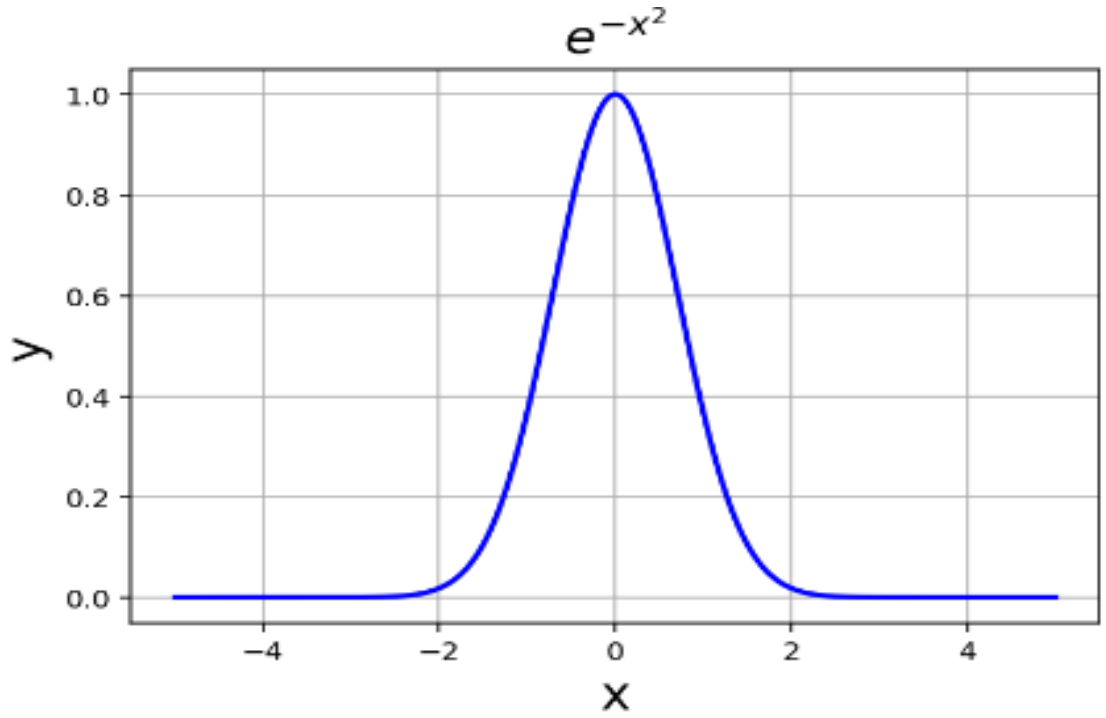


Graficación de funciones explicitas con Python

Ejemplo 1. Graficar la siguientes funciones:

$\mathbf{1.}\ \boldsymbol{y = e^{-x^2}}$ $\qquad 2.\ y = e^{-x}sen(2\pi x)$ $\qquad 3.\ y = \frac{xe^{-x}}{1-x^2}$

Nos debe mostrar la siguiente figura



Graficación de funciones explicitas con Python

Ejemplo 1. Graficar la siguientes funciones:

$$1.\ y = e^{-x^2} \qquad \mathbf{2.\ y = e^{-x}sen(2\pi x)} \qquad 3.\ y = \frac{xe^{-x}}{1-x^2}$$

2. Solución

```
import numpy as np #importa la libreria numpy y la renombra para invocarla como np
import matplotlib.pyplot as plt #importa la libreria matplotlib y la renombra para invocarla como plt

x=np.linspace(0,5,500) #se genera un vector de -5 hasta 5 con 500 puntos
y=np.exp(-x)*np.sin(2*np.pi*x) #se define la función a graficar

fig=plt.figure(figsize=(8,4)) #se genera una figura de 8x6
plt.grid(True) #se activan las cuariculas
plt.plot(x,y,'b-',linewidth=2)  # Se grafica la y
plt.xlabel("x",size=18) #titulo del eje x
plt.ylabel("y",size=18) #titulo del eje y
plt.title(r"$e^{-x}sin(2 \pi x)$",size=18) #titulo del gráfico
plt.show() # esta instruccion es necesaria o no se mostrara la figura
```

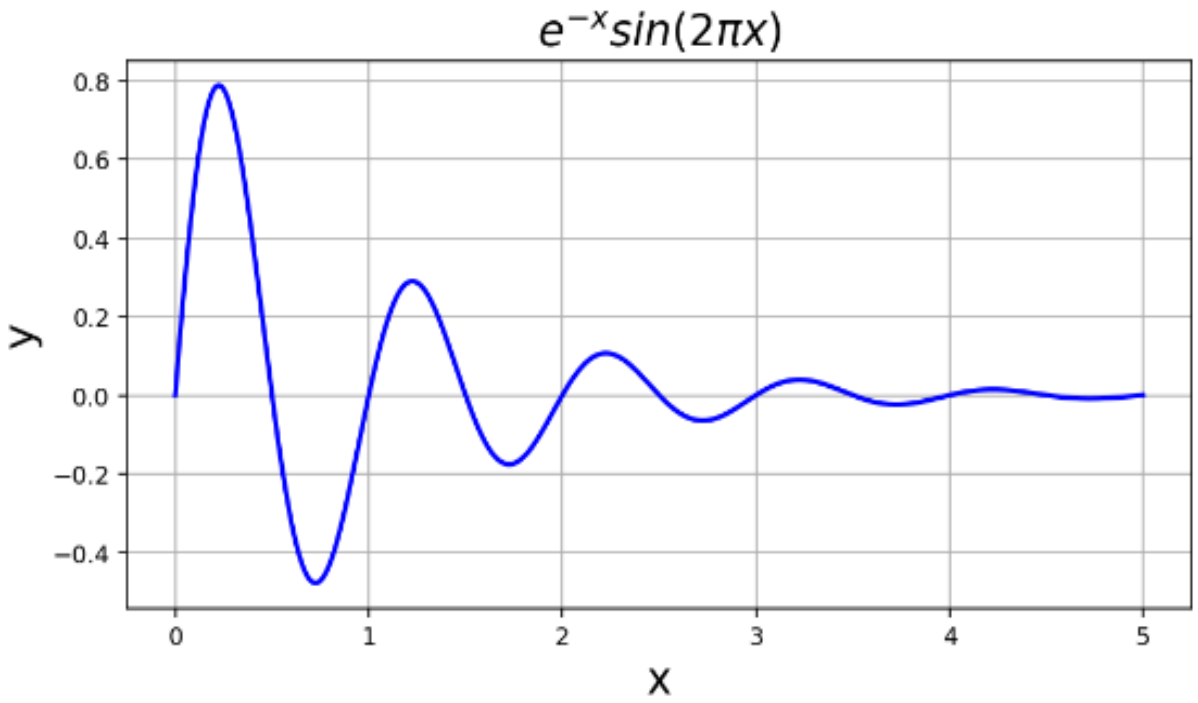
Jypiter permite ejecutar el código por bloques, por lo cual si en un bloque anterior ya se importaron las librerías necesarias, no es necesario volver a importarla en el siguiente bloque.

Graficación de funciones explicitas con Python

Ejemplo 1. Graficar la siguientes funciones:

$1. y = e^{-x^2}$ $\quad$ **$2. y = e^{-x} sen(2\pi x)$** $\quad$ $3. y = \frac{xe^{-x}}{1-x^2}$

Nos debe mostrar la siguiente figura:



Graficación de funciones explicitas con Python

Ejemplo 1. Graficar la siguientes funciones:

$$1.\ y = e^{-x^2} \qquad 2.\ y = e^{-x} sen(2\pi x) \qquad \mathbf{3.}\ \boldsymbol{y = \frac{xe^{-x}}{1-x^2}}$$

3. Solución:

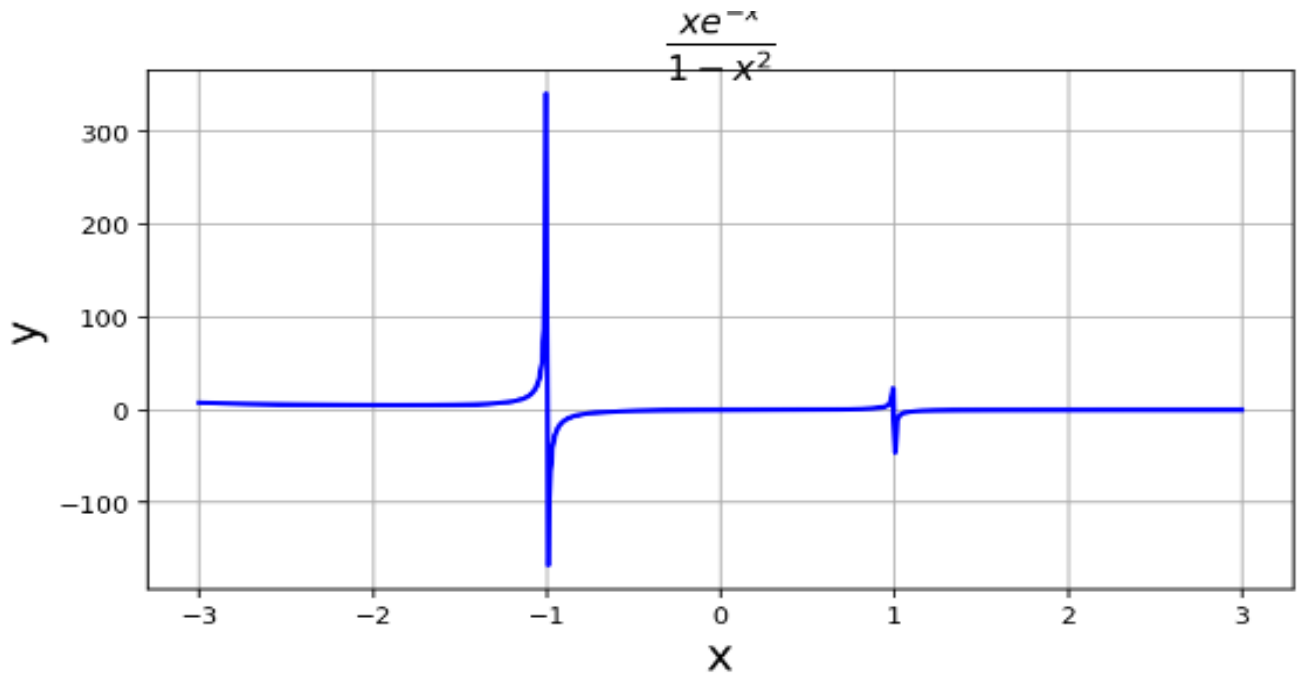
```
import numpy as np #importa la libreria numpy y la renombra para invocarla como np
import matplotlib.pyplot as plt #importa la libreria matplotlib y la renombra para invocarla como plt

x=np.linspace(-3,3,500) #se genera un vector de -3 hasta 3 con 500 puntos
y=x*np.exp(-x)/(1-x**2) #se define la función a graficar

fig=plt.figure(figsize=(8,4)) #se genera una figura de 8x6
plt.grid(True) #se activan las cuariculas
plt.plot(x,y,'b-',linewidth=2)  # Se grafica la y
plt.xlabel("x",size=18) #titulo del eje x
plt.ylabel("y",size=18) #titulo del eje y
plt.title(r"$\frac{xe^{-x}}{1-x^2}$",size=20) #titulo del gráfico
plt.show() # esta instruccion es necesaria o no se mostrara la figura
```

Graficación de funciones explicitas con Python

Ejemplo 1. Graficar la siguientes funciones:

$1. y = e^{-x^2}$ $\quad$ $2. y = e^{-x} sen(2\pi x)$ $\quad$ $\mathbf{3.}\ \boldsymbol{y = \frac{xe^{-x}}{1-x^2}}$

Nos debe mostrar la siguiente figura

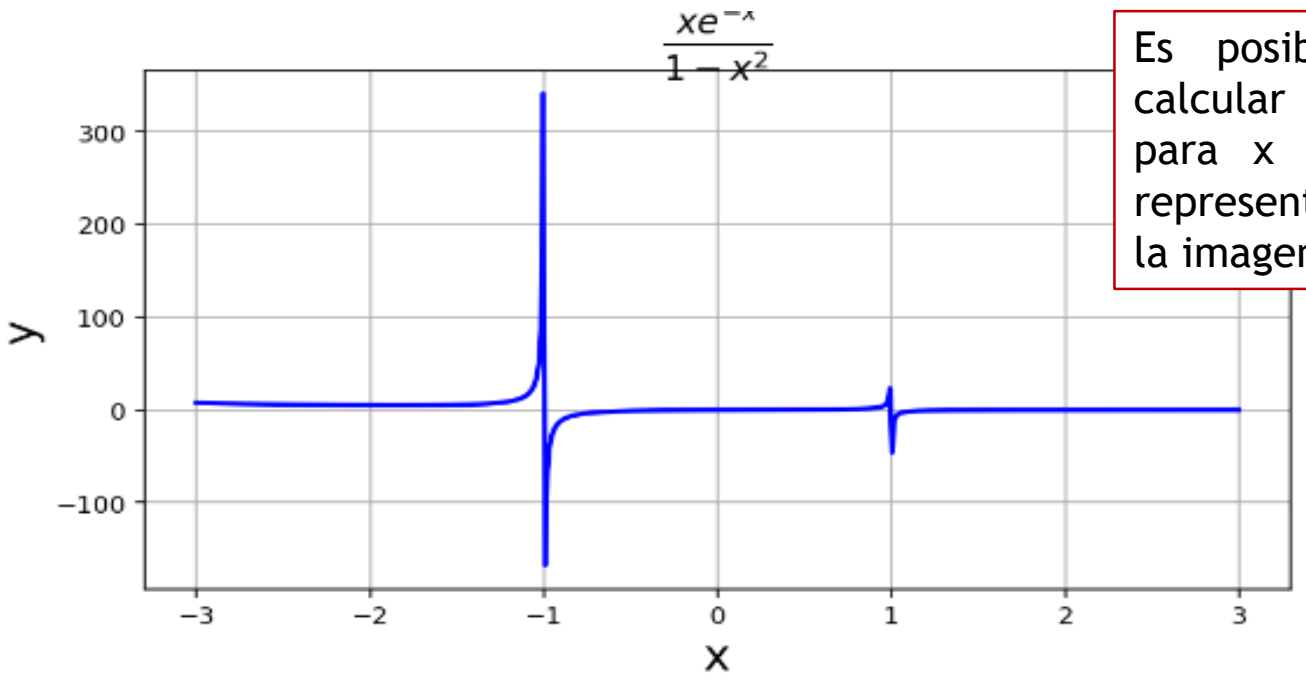


Graficación de funciones explicitas con Python

Ejemplo 1. Graficar la siguientes funciones:

$1. y = e^{-x^2}$ $\quad$ $2. y = e^{-x} sen(2\pi x)$ $\quad$ $\mathbf{3.}\ \boldsymbol{y = \frac{xe^{-x}}{1-x^2}}$

Nos debe mostrar la siguiente figura



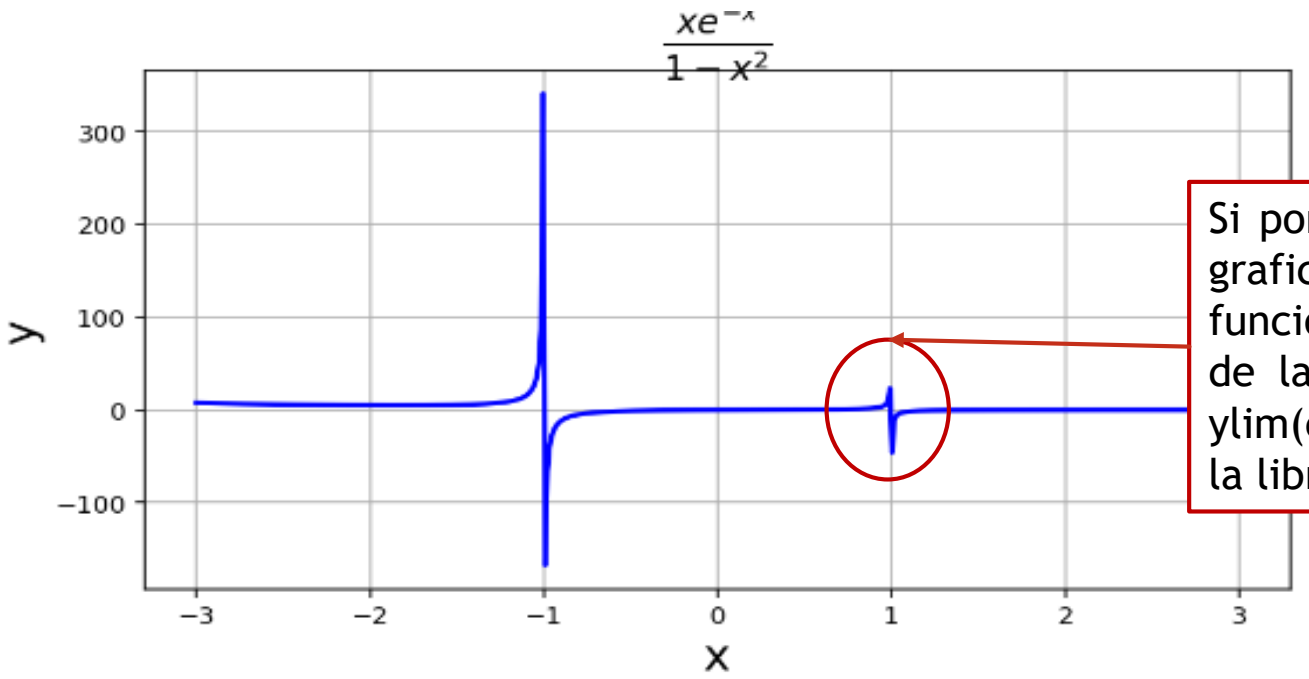
Es posible que aunque necesitemos calcular todos los valores de la función para x entre -3 y 3 solo se desee representar gráficamente una porción de la imagen.

Graficación de funciones explicitas con Python

Ejemplo 1. Graficar la siguientes funciones:

$1. y = e^{-x^2}$ $\quad$ $2. y = e^{-x} sen(2\pi x)$ $\quad$ $\mathbf{3.}\ \boldsymbol{y = \frac{xe^{-x}}{1-x^2}}$

Nos debe mostrar la siguiente figura



Si por ejemplo solo se desea graficar esta porción de la función, se puede hacer uso de las funciones xlim(a,b) y ylim(c,d) que hacen parte de la librería matplotlib

Graficación de funciones explicitas con Python

Ejemplo 1. Graficar la siguientes funciones:

$$1.\ y = e^{-x^2} \qquad 2.\ y = e^{-x} sen(2\pi x) \qquad \mathbf{3.}\ \boldsymbol{y = \frac{xe^{-x}}{1-x^2}}$$

Vemos:

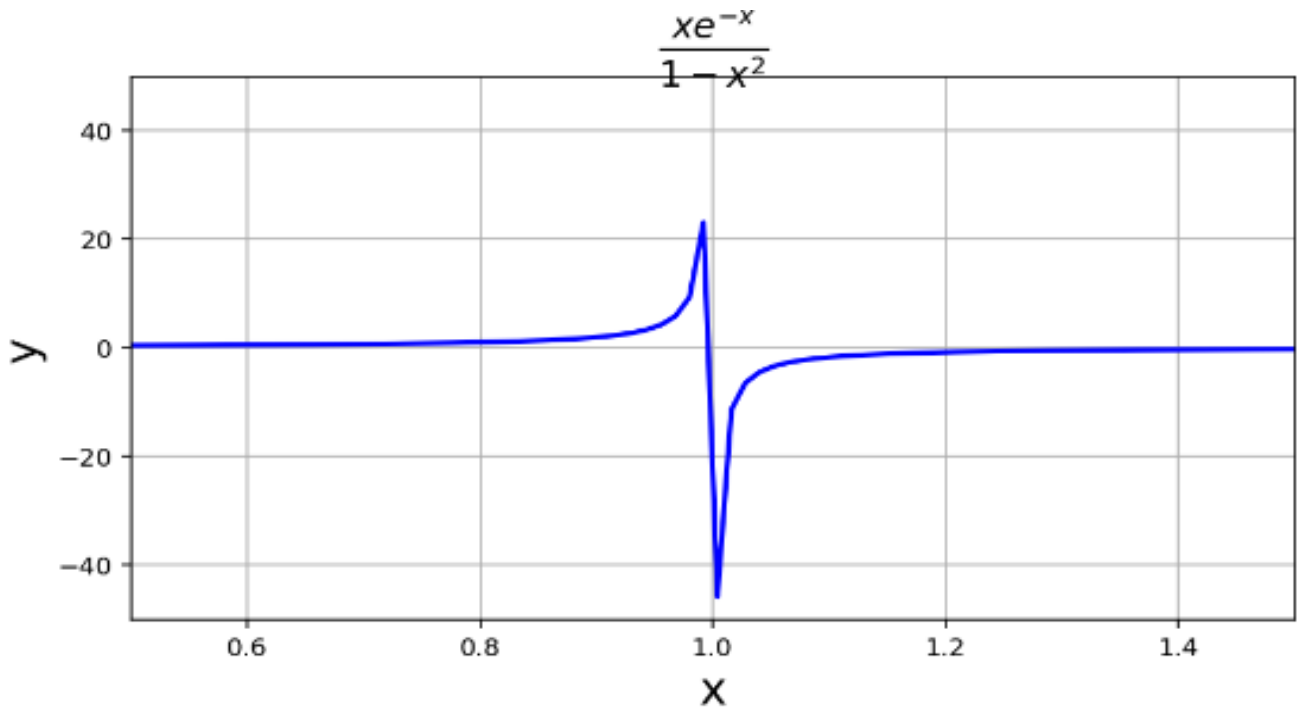
```
x=np.linspace(-3,3,500) #se genera un vector de -3 hasta 3 con 500 puntos
y=x*np.exp(-x)/(1-x**2) #se define la función a graficar

fig=plt.figure(figsize=(8,4)) #se genera una figura de 8x6
plt.grid(True) #se activan las cuariculas
plt.plot(x,y,'b-',linewidth=2)  # Se grafica la y
plt.xlabel("x",size=18) #titulo del eje x
plt.ylabel("y",size=18) #titulo del eje y
plt.xlim(0.5,1.5) #Se grafica para x entre 0.5 y 1.5
plt.ylim(-50,50) #Se grafica para y entre -50 y 50
plt.title(r"$\frac{xe^{-x}}{1-x^2}$",size=20) #titulo del gráfico
plt.show() # esta instruccion es necesaria o no se mostrara la figura
```

Graficación de funciones explicitas con Python

Ejemplo 1. Graficar la siguientes funciones:

$1. y = e^{-x^2}$ $\quad$ $2. y = e^{-x} sen(2\pi x)$ $\quad$ $\mathbf{3.}\ \boldsymbol{y = \frac{xe^{-x}}{1-x^2}}$

Nos debe mostrar la siguiente figura



Como graficar una familia de curvas

Veamos como seria el código para graficar algunas curvas pertenecientes a la familia de curvas:

$$y = Ce^{-x^2}$$

Como graficar una familia de curvas

Veamos como seria el código para graficar algunas curvas pertenecientes a la familia de curvas:

$$y = Ce^{-x^2}$$

Solución:

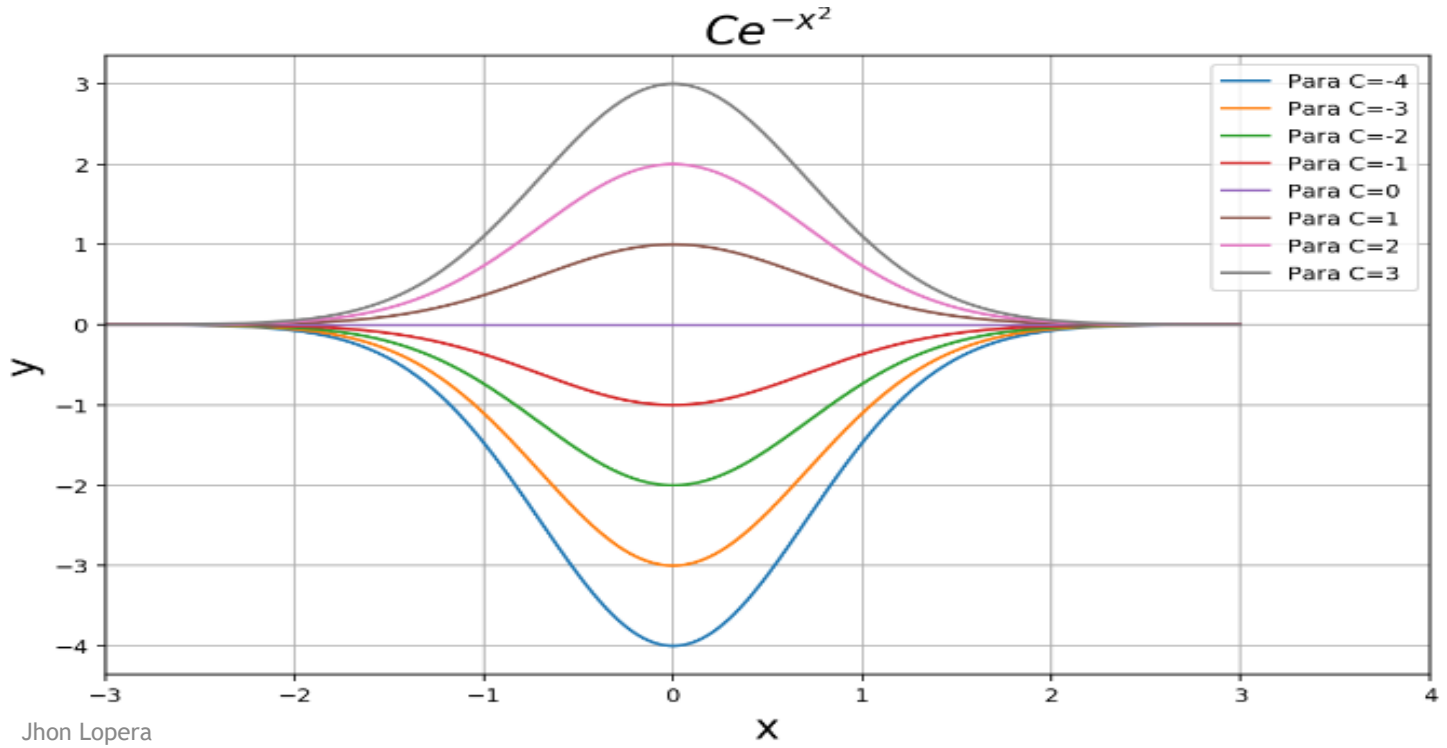
```
import numpy as np #importa la libreria numpy y la renombra para invocarla como np
import matplotlib.pyplot as plt #importa la libreria matplotlib y la renombra para invocarla como plt

x=np.linspace(-3,3,500)
fig=plt.figure(figsize=(8,4))
for C in range(-4,4,1): #Ciclo para graficar varias veces sobre la misma figura
    y=C*np.exp(-x**2)
    plt.plot(x,y,label="Para C=%d" %C)
    plt.xlim(-3,4) # limites para el eje x
    plt.grid(True) # Activa la cuadrícula
    plt.legend(loc=1) #Esta instrucción activa las leyendas
plt.show()
```

Como graficar una familia de curvas

Veamos como seria el código para graficar algunas curvas pertenecientes a la familia de curvas:

$$y = Ce^{-x^2}$$

Resultado:



Maxima

Antes de comenzar con máxima es necesario aclarar algunas cosas:

1.Para asignar un valor o un conjunto de valores a una variable se hace con ":" y no con "=",ya que este representa una igualdad lo cual es en este caso es diferente. Así, sí se tiene la función $y = sin(x)$ tendrá que escribirse como:

y: sin(x)

Maxima

Antes de comenzar con máxima es necesario aclarar algunas cosas:

1. Para asignar un valor o un conjunto de valores a una variable se hace con ":" y no con "=",ya que este representa una igualdad lo cual es en este caso es diferente. Así, sí se tiene la función $y = sin(x)$ tendrá que escribirse como:

y: sin(x)

2. Existen básicamente 2 opciones para trabajar con Maxima; Descargando el instalador, ya sea para pc o dispositivo Android o trabajando con la plataforma online http://maxima-online.org.

Maxima

Antes de comenzar con máxima es necesario aclarar algunas cosas:

1. Para asignar un valor o un conjunto de valores a una variable se hace con ":" y no con "=",ya que este representa una igualdad lo cual es en este caso es diferente. Así, sí se tiene la función $y = sin(x)$ tendrá que escribirse como:

y: sin(x)

2. Existen básicamente 2 opciones para trabajar con Maxima; Descargando el instalador, ya sea para pc o dispositivo Android o trabajando con la plataforma online http://maxima-online.org.

Si no le es posible instalar el pc se recomienda la aplicación para Android, ya que la plataforma online tiene algunas limitaciones.(Este tutorial se basara en lo posible en la plataforma online)

Graficación de funciones explicitas con Maxima

Ejemplo 2. Graficar la siguientes funciones:

$1. y = e^{-x^2}$ $2. y = e^{-x} sen(2\pi x)$ $3.\ y = \cos(4\pi) + 2x^2$

Graficación de funciones explicitas con Maxima

Ejemplo 2. Graficar la siguientes funciones:

$\mathbf{1.}\ y = e^{-x^2}$ $\quad 2.\ y = e^{-x}sen(2\pi x)$ $\quad 3.\ y = \cos(4\pi) + 2x^2$

1.Solucion:

y:%e^(-x^2); Para usar el numero Euler se antepone "%" a la "e"

plot2d (y, [x, -5, 5]); Para definir el rango de graficación primero va la variable dependiente(x) y luego el intervalo deseado

Para hacer varias graficas en una misma figura se usa la siguiente sentencia:

plot2d ([y_1, y_2 ...], [x, x_i, x_f]); y_1 y y_2 son funciones definidas en x

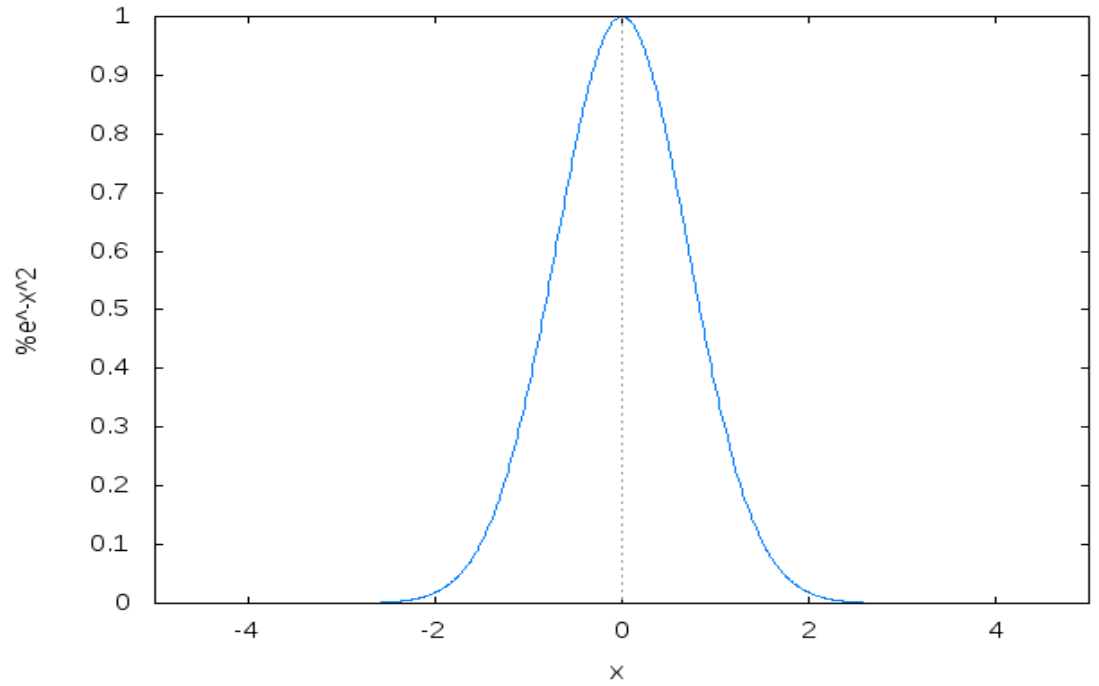
Para graficar le damos clic a "Calculate" que se encuentra debajo del cuadro de comandos

Graficación de funciones explicitas con Maxima

Ejemplo 2. Graficar la siguientes funciones:

1. $\boldsymbol{y = e^{-x^2}}$ 2. $y = e^{-x} sen(2\pi x)$ 3. $y = \cos(4\pi) + 2x^2$

1.Solucion:



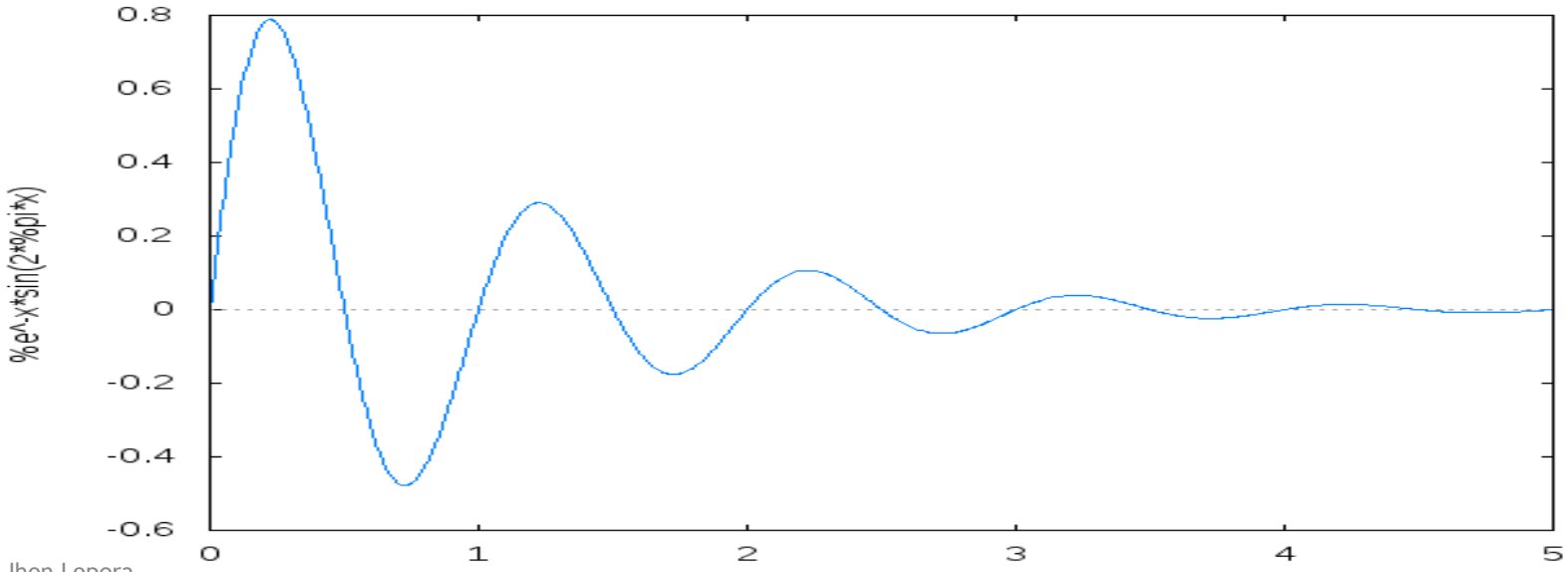
Graficación de funciones explicitas con Maxima

Ejemplo 2. Graficar la siguientes funciones:

$1. y = e^{-x^2}$ **$2. y = e^{-x} sen(2\pi x)$** $3.\ y = \cos(4\pi) + 2x^2$

2.Solucion:

```
y:%e^(-x)*sin(2*%pi*x);
plot2d (y, [x, 0, 5]);
```



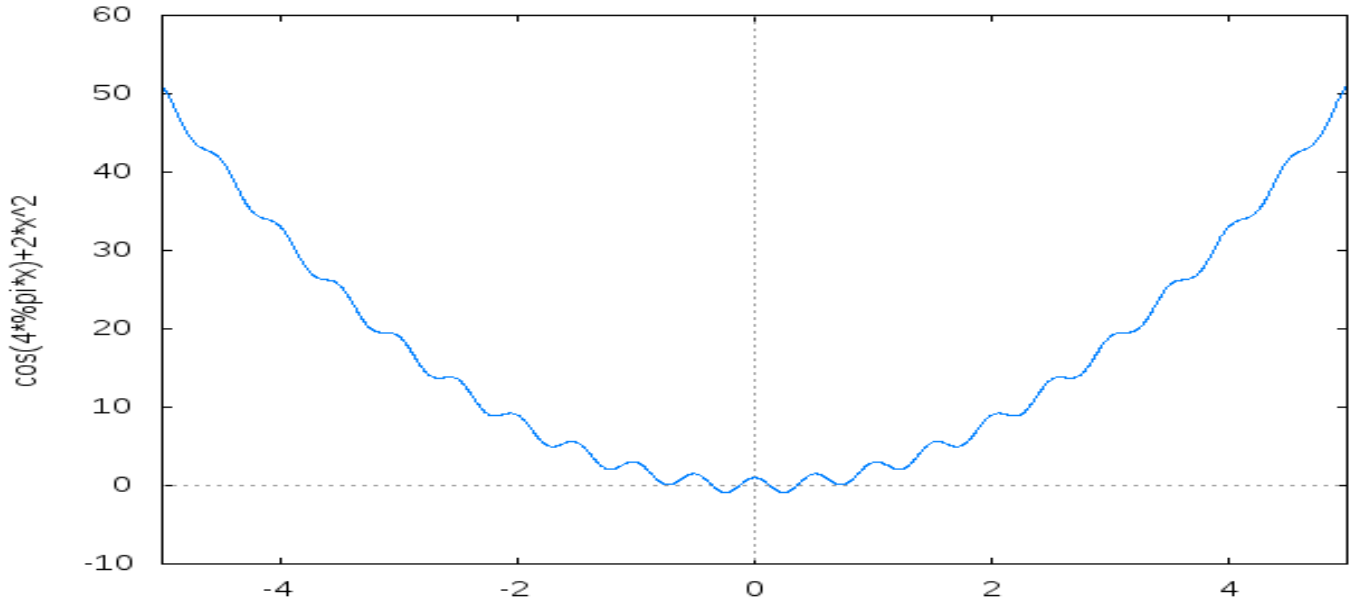
Graficación de funciones explicitas con Maxima

Ejemplo 2. Graficar la siguientes funciones:

$1. y = e^{-x^2}$ $2. y = e^{-x} sen(2\pi x)$ **$3. y = cos(4\pi x) + 2x^2$**

3.Solucion:

```
y:cos(4*%pi*x)+2*x^2;
plot2d (y, [x, -5, 5]);
```



Matlab

Ejemplo 3. Graficar la siguientes funciones:

$1. y = x * e^{-x^2} + 2$ $2. y = e^{-x} sen(2\pi x)$ $3.\ y = \cos(4\pi x) + 2x^2$

Matlab

Ejemplo 3. Graficar la siguientes funciones:

$\mathbf{1.}\ \boldsymbol{y = x * e^{-x^2} + 2}$ $\quad 2.\ y = e^{-x} sen(2\pi x)$ $\quad 3.\ y = \cos(4\pi x) + 2x^2$

1.Solucion:

Solución para variable simbólica

```
syms x                  % se define una variable simbólica
y=exp(-x^2)+x+2;        %se define la función a graficar
ezplot(y,[-5,5])        % se usa ezplot para graficar variable simbólica
grid on                 % se activa la cuadricula
```

Matlab

Ejemplo 3. Graficar la siguientes funciones:

$\mathbf{1.}\ y = x * e^{-x^2} + 2$ $\quad 2.\ y = e^{-x} sen(2\pi x)$ $\quad 3.\ y = \cos(4\pi x) + 2x^2$

1.Solucion:

Solución para variable simbólica

```
syms x                  % se define una variable simbólica
y=exp(-x^2)+x+2;        %se define la función a graficar
ezplot(y,[-5,5]) % se usa ezplot para graficar variable simbólica
grid on                 % se activa la cuadricula
```

Solución para un conjunto de puntos(vector):

```
x=linspace(-5,5,500); %se define la función a graficar
y=exp(-x.^2).*x+2;
plot(t,y)               % se usa plot para graficar una función evaluada en
                        un conjunto de puntos
grid on                 % se activa la cuadricula
```

Note que cuando se trabaja con vectores se debe usar ".*" para hacer una multiplicación entre funciones.

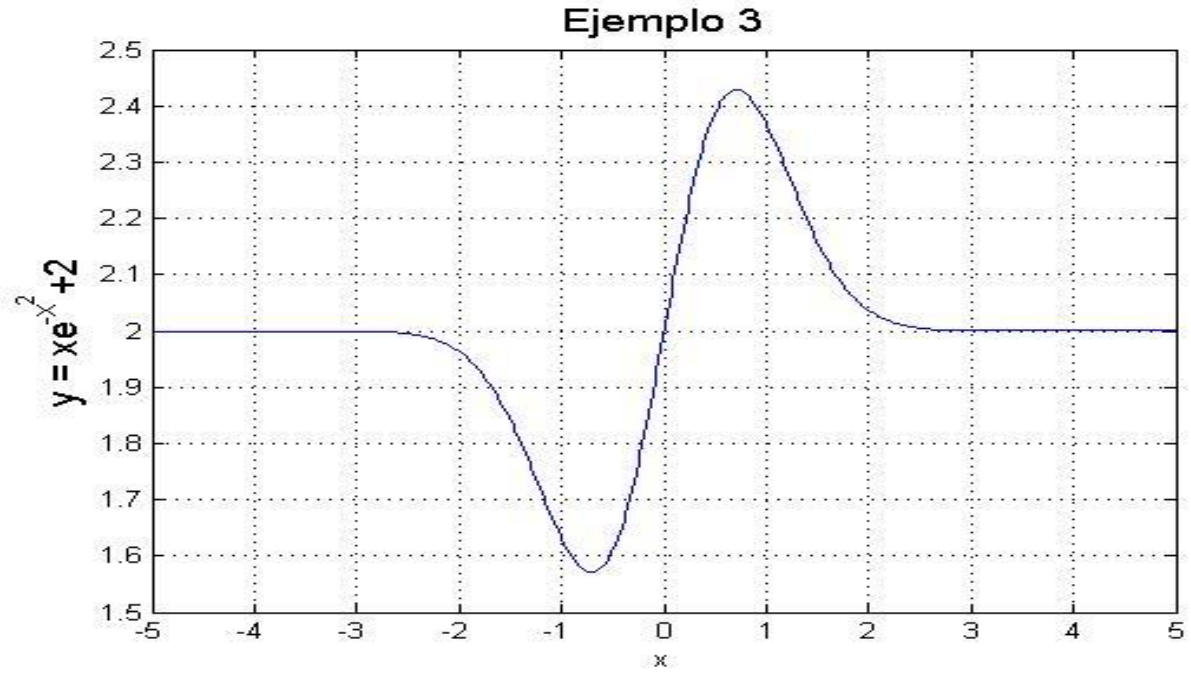
Matlab

Ejemplo 3. Graficar la siguientes funciones:

$\mathbf{1.}\ \boldsymbol{y = x * e^{-x^2} + 2}$ $\quad 2.\ y = e^{-x} sen(2\pi x)$ $\quad 3.\ y = \cos(4\pi x) + 2x^2$

1.Solucion:

Resultado



Familia de curvas con Matlab

Para graficar una familia de curvas con Matlab se realiza un procedimiento similar al que se hizo en Python (salvo por cambios en la sintaxis).

Ejemplo 4. Grafique la siguiente familia de curvas:

$$y = \frac{1}{c}x^2 + 2$$

Familia de curvas con Matlab

Para graficar una familia de curvas con Matlab se realiza un procedimiento similar al que se hizo en Python (salvo por cambios en la sintaxis).

Ejemplo 4. Grafique la siguiente familia de curvas:

$$y = \frac{1}{c}x^2 + 2$$

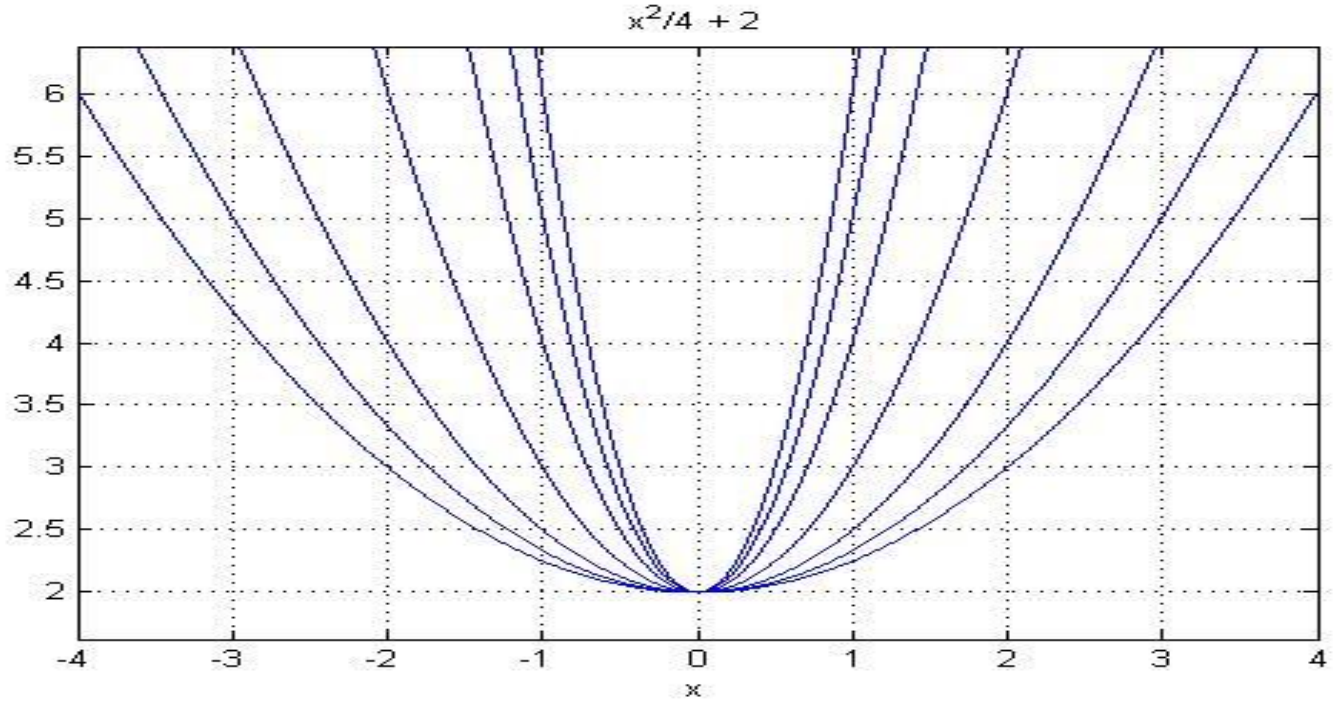
Solución:

```
syms x                  %Se define la variable simbólica x
for c=1:1:4             % Se hace un ciclo con 4 iteraciones
    y=1/c*x^2+2;        % Se define la función a graficar
    ezplot(y,[-4,4])
    hold on
    grid on
end
```

Familia de curvas con Matlab

Para graficar una familia de curvas con Matlab se realiza un procedimiento similar al que se hizo en Python (salvo por cambios en la sintaxis).

Resultado



Graficación implícita con Python

En algunos casos (como con las cónicas hiperbólicas por ejemplo) no es posible despejar la variable dependiente de forma explicita. Aun así existe una alternativa que permite representar de forma gráfica dicha función.

Ejemplo 5. Graficar la siguiente expresión:

$$\frac{X^2}{2} - \mathrm{y}^2 - \mathrm{e}^{-\mathrm{xy}} = \mathrm{C}$$

Graficación implícita con Python

En algunos casos (como con las cónicas hiperbólicas por ejemplo) no es posible despejar la variable dependiente de forma explicita. Aun así existe una alternativa que permite representar de forma gráfica dicha función.

Solución:

```
import numpy as np #importa la libreria numpy y la renombra para invocarla como np
import matplotlib.pyplot as plt #importa la libreria matplotlib y la renombra para invocarla como plt

def f(x,y): # Es necesario definir la ecuacion a graficar como una función
    return ((x**2)/2)- (y**2)-np.exp(-x*y)

x=np.linspace(-10,10,500)               # Rango y puntos en x
y=np.linspace(-10,10,500)               # Rango y puntos en y
C=[0,1,2,3]
X, Y = np.meshgrid(x, y)                # construccion de malla
cs=plt.contour(X, Y, f(X, Y),C)     # gráfica
plt.xlabel("x",size=18)                 #titulo del eje x
plt.ylabel("y",size=18)                 #titulo del eje y
plt.title(r'$\frac{x^{2}}{2}-y^2 -e^{-xy}=C$',size=20) #titulo del gráfico
plt.grid(True)
```

Graficación implícita con Python

En algunos casos (como con las cónicas hiperbólicas por ejemplo) no es posible despejar la variable dependiente de forma explicita. Aun así existe una alternativa que permite representar de forma gráfica dicha función.

Solución:

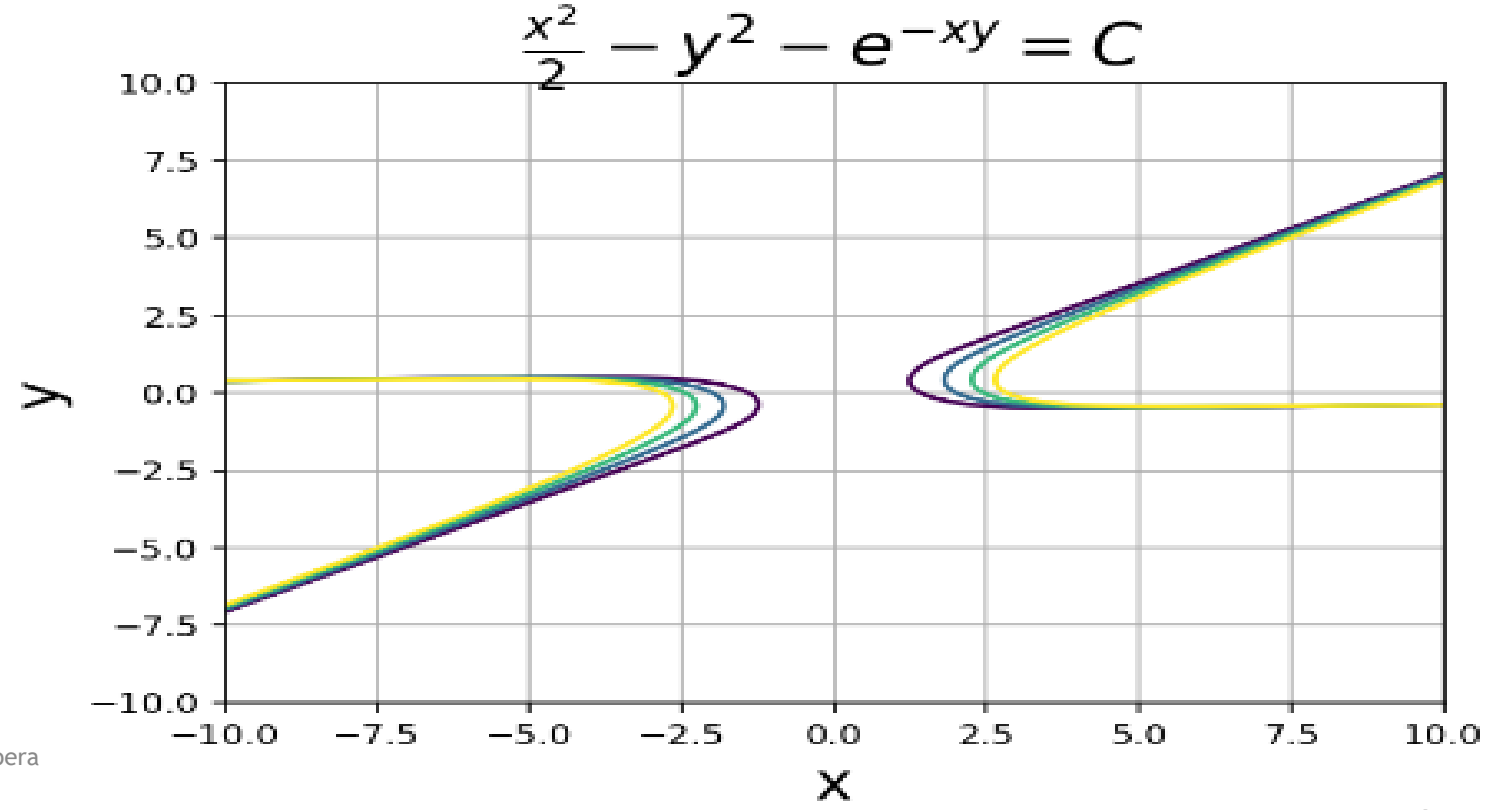
```
import numpy as np #importa la libreria numpy y la renombra para invocarla como np
import matplotlib.pyplot as plt #importa la libreria matplotlib y la renombra para invocarla como plt

def f(x,y): # Es necesario definir la ecuacion a graficar como una función
    return ((x**2)/2)- (y**2)-np.exp(-x*y)

x=np.linspace(-10,10,500)                # Rango y puntos en x
y=np.linspace(-10,10,500)                # Rango y puntos en y
C=[0,1,2,3]
X, Y = np.meshgrid(x, y)                 # construccion de malla
cs=plt.contour(X, Y, f(X, Y),C)     # gráfica
plt.xlabel("x",size=18)                  #titulo del eje x
plt.ylabel("y",size=18)                  #titulo del eje y
plt.title(r'$\frac{x^{2}}{2}-y^2 -e^{-xy}=C$',size=20) #titulo del gráfico
plt.grid(True)
```

Se define una función del tipo $f(x,y) = C$. Donde la expresión a retornar es el lado izquierdo de la igualdad. El lado derecho se entrega en forma de lista, como argumento del comando contour.

Graficación implícita con Python

En algunos casos (como con las cónicas hiperbólicas por ejemplo) no es posible despejar la variable dependiente de forma explicita. Aun así existe una alternativa que permite representar de forma gráfica dicha función.

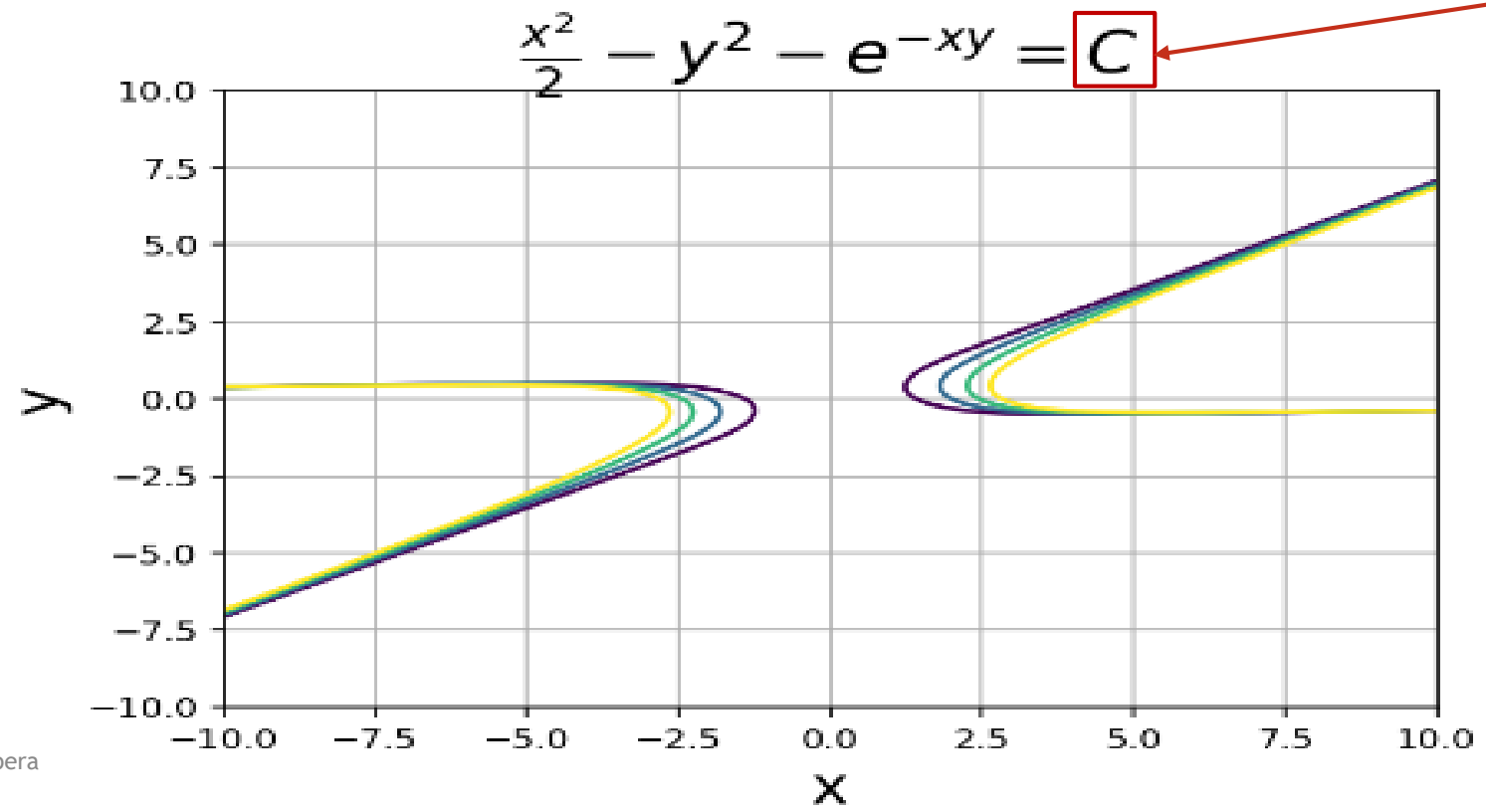
Resultado:



Graficación implícita con Python

En algunos casos (como con las cónicas hiperbólicas por ejemplo) no es posible despejar la variable dependiente de forma explicita. Aun así existe una alternativa que permite representar de forma gráfica dicha función.

Resultado:



Una de las ventajas de trabajar con contour es que podemos graficar la familia de curvas. Solo es necesario entregarle los valores de C en forma de lista

Graficación implícita con Maxima

Ejemplo 6. Represente gráficamente l siguiente expresión:

$$x^2 = y^2 + 1$$

En el caso de Maxima se usa la función *implicit_plot()* para graficar expresiones de este tipo. Dicha función recibe como argumento una igualdad, es decir, recibe la ecuación tal y como esta escrita en el ejemplo(o en una forma similar).

Graficación implícita con Maxima

Ejemplo 6. Represente gráficamente la siguiente expresión

$$x^2 = y^2 + 1$$

Solución:

```
load(implicit_plot);  #Es necesario cargar primero la función implicit_plot
implicit_plot (x^2 = y^2 + 1, [x, -4, 4], [y, -4, 4]); # Se grafica entre -4 y 4
                                                         tanto para x como para y
```

Graficación implícita con Maxima

Ejemplo 6. Represente gráficamente la siguiente expresión:

$$x^2 = y^2 + 1$$

Solución:

```
load(implicit_plot);  #Es necesario cargar primero la función implicit_plot
implicit_plot (x^2 = y^2 + 1, [x, -4, 4], [y, -4, 4]); # Se grafica entre -4 y 4
                                                        tanto para x como para y
```

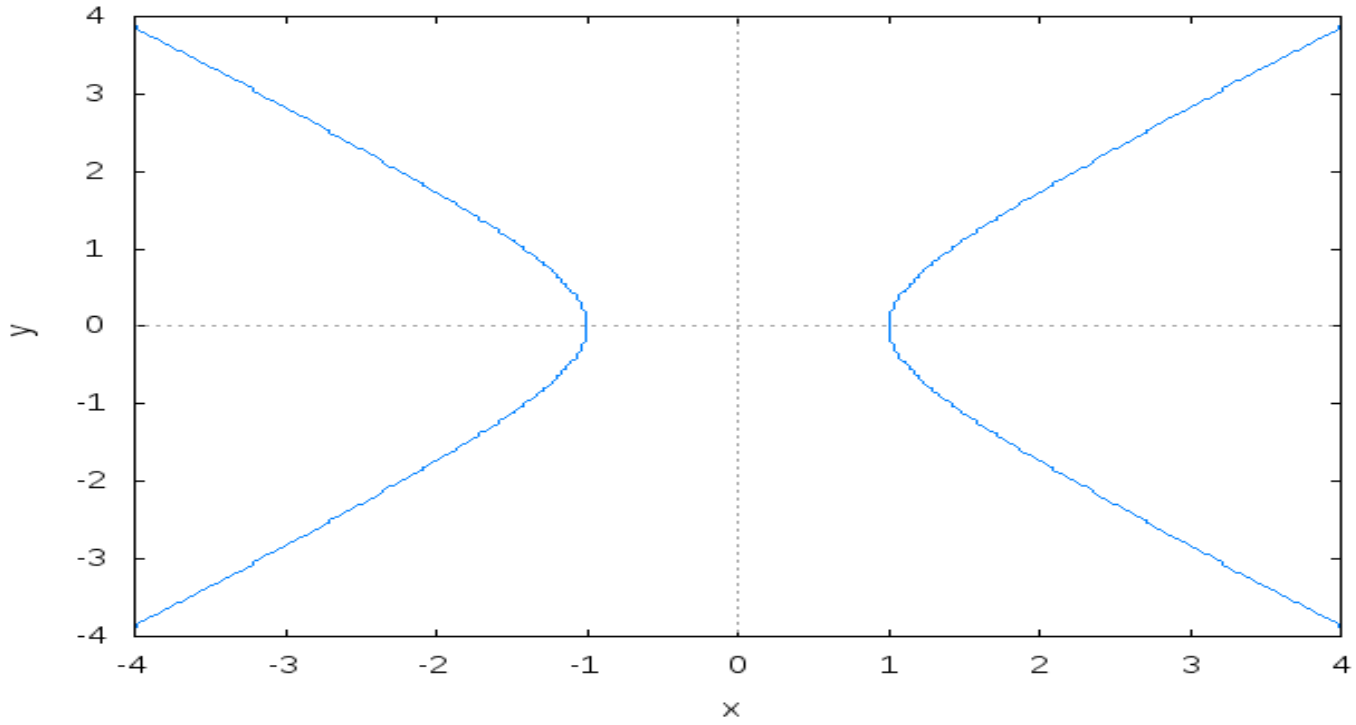
Como se menciono antes, máxima no usa "=" para asignar valores a una variable, si no para representar igualdades que se usan en este tipo de funciones.

Graficación implícita con Maxima

Ejemplo 6. Represente gráficamente la siguiente expresión:

$$x^2 = y^2 + 1$$

Resultado:



Graficación implícita con Matlab

Con MatLab veremos 2 opciones para graficar una función; La función contour (la cual se usa de forma similar a la vista en Python) y la función ezplot.

Ejemplo 7:

$$x^2 = y^2 + C$$

Graficación implícita con Matlab

Con MatLab veremos 2 opciones para graficar una función; La función contour (la cual se usa de forma similar a la vista en Python) y la función ezplot.

Ejemplo 7:

$$x^2 = y^2 + C$$

Solucion 1 (contour)

```
x = linspace(-10,10,500);
y = linspace(-10,10,500);
[X,Y] = meshgrid(x,y); %se genera l cuadricula
Z = X.^2 - Y.^2 ; % se define una expresión de la forma f(x,y) = C
c=[0,1,2,3]; %se define un vector de valores para C
contour(X,Y,Z,c)
grid on
```

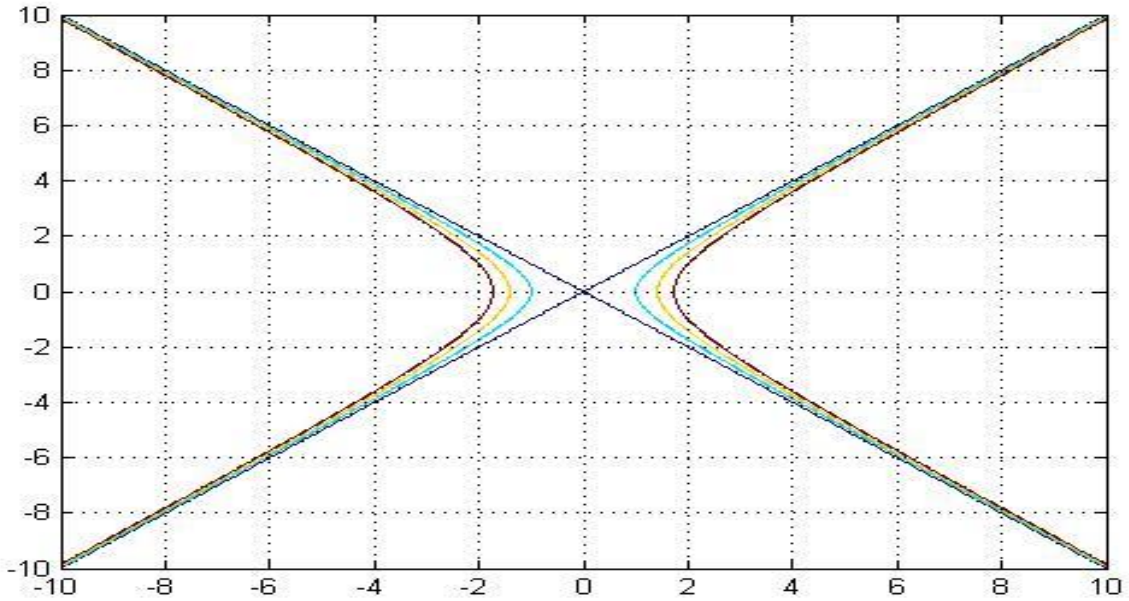
Graficación implícita con Matlab

Con MatLab veremos 2 opciones para graficar una función; La función contour (la cual se usa de forma similar a la vista en Python) y la función ezplot.

Ejemplo 7:

$$x^2 = y^2 + C$$

Resultado:



Graficación implícita con Matlab

Con MatLab veremos 2 opciones para graficar una función; La función contour (la cual se usa de forma similar a la vista en Python) y la función ezplot.

Ejemplo 7:

$$x^2 = y^2 + C$$

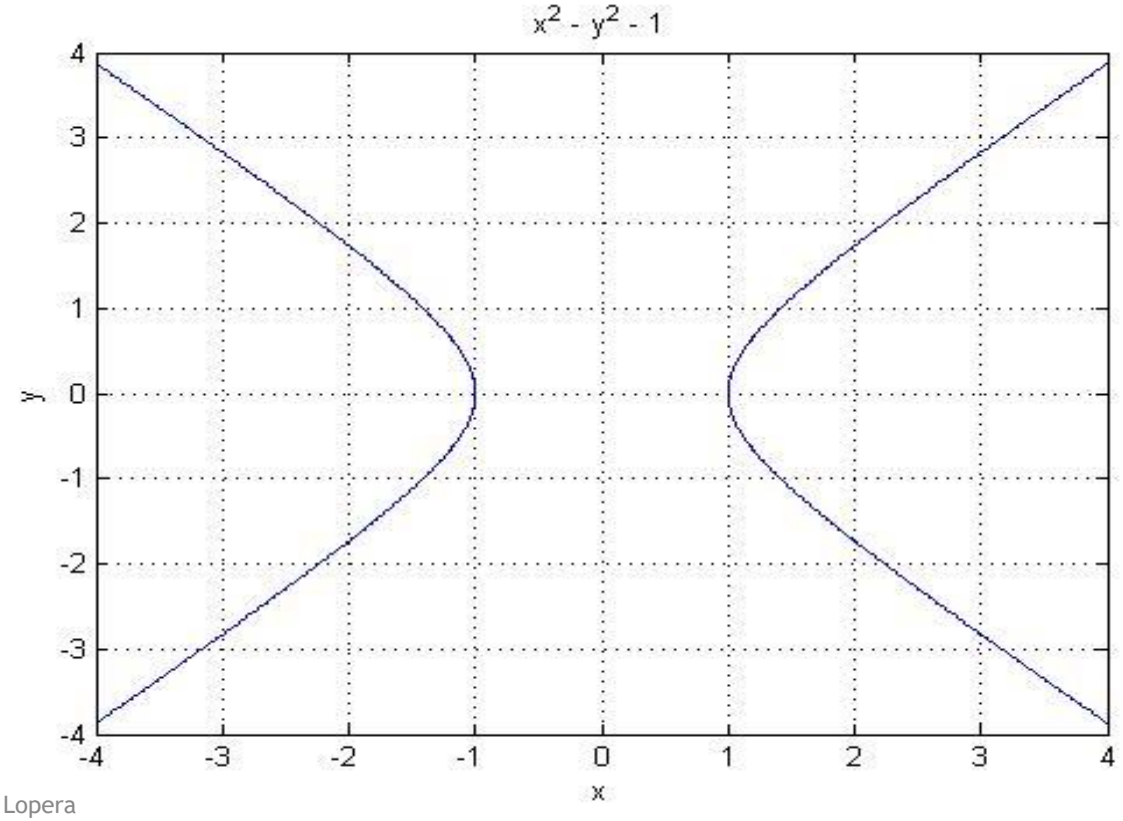
Solucion 2 (ezplot)

```
syms x y % Se deben definir x e y como simbólicas
z=x^2-y^2-1; % se define la expresión a graficar (c=1)
g=ezplot(z,[-4,4]); % se grafica entre -4 y 4
set(g,'color','b') % con esta instrucción se cambia el color de la
                   grafica hecha con ezplot
grid on %cuadricula
```

Graficación implícita con Matlab

Con MatLab veremos 2 opciones para graficar una función; La función contour (la cual se usa de forma similar a la vista en Python) y la función ezplot.

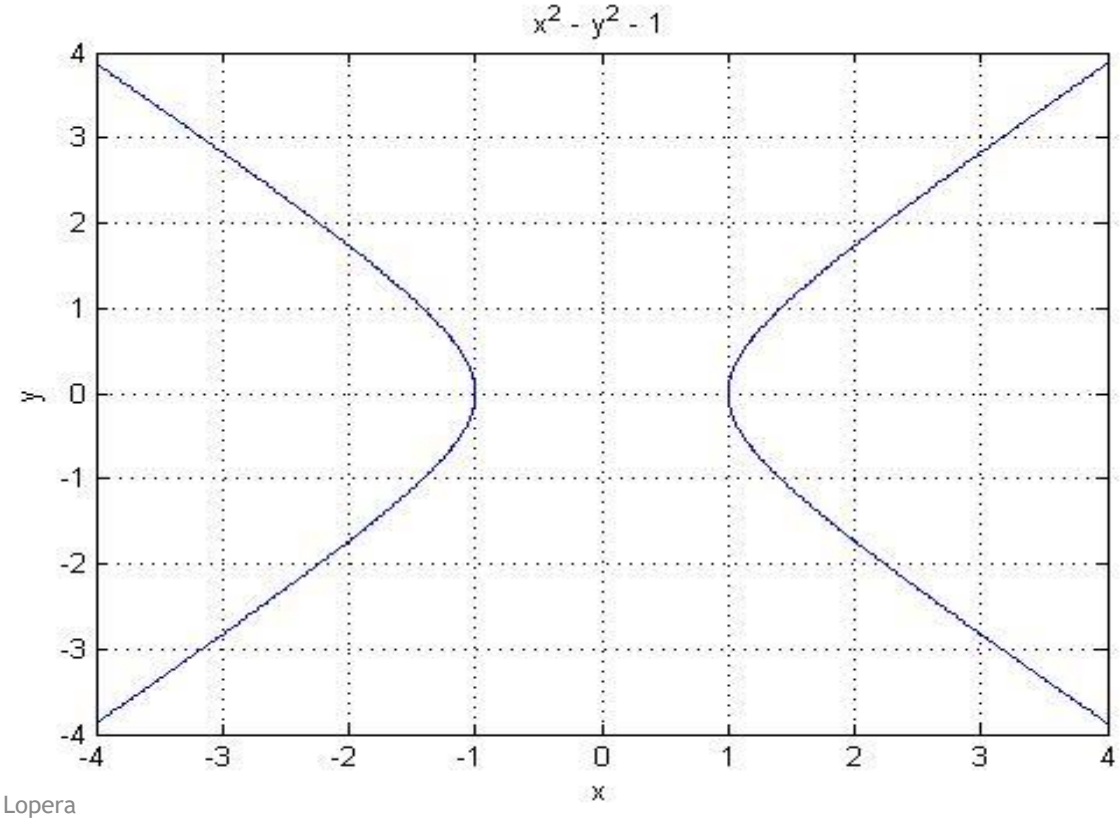
Resultado:



Graficación implícita con Matlab

Con MatLab veremos 2 opciones para graficar una función; La función contour (la cual se usa de forma similar a la vista en Python) y la función ezplot.

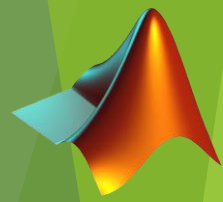
Resultado:



El inconveniente con usar ezplot(), es que solo se hace una curva a la vez (Para C=1 en este caso).

Si se quieren graficar varias ecuaciones pertenecientes a la misma familia de curvas usando ezplot, es necesario realizar un proceso que itere sobre la variable c, tal como se mostro en ejemplos anteriores.

Graficación implícita con Matlab



Con MatLab veremos 2 opciones para graficar una función; La función contour (la cual se usa de forma similar a la vista en Python) y la función ezplot.

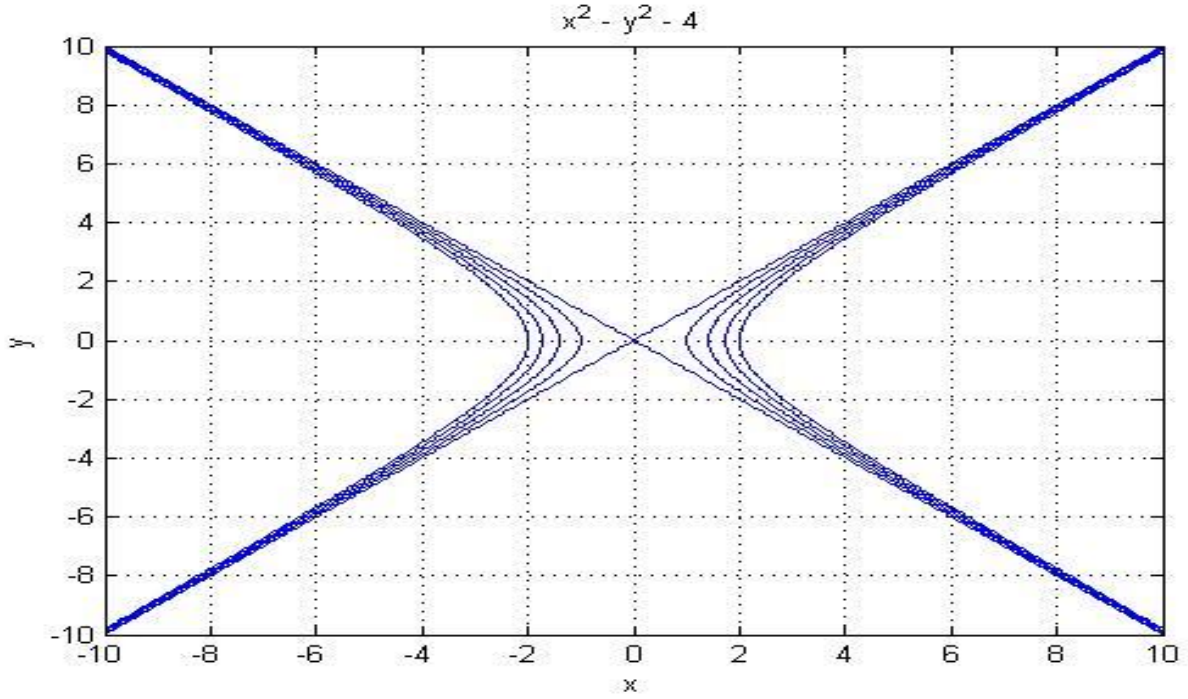
Veamos:

```
syms x y % Se deben definir x e y como simbolicas
        for c=0:1:4
            z=x^2-y^2-c;
            g=ezplot(z,[-10,10]); % se grafica entre -4 y 4
            set(g,'color','b') % con esta instrucción se cambia el color de la
                               grafica hecha con ezplot
            hold on
            grid on
        end
```

Graficación implícita con Matlab

Con MatLab veremos 2 opciones para graficar una función; La función contour (la cual se usa de forma similar a la vista en Python) y la función ezplot.

Resultado:



Campo de pendientes con Python

Ejemplo 8. Represente el campo de pendientes de la siguiente ecuación diferencial:

$$\frac{dy}{dx} = 2\frac{y}{x}$$

Para realizar el campo de pendientes de una ecuación diferencial (lo cual es bastante útil cuando la ecuación no es resoluble analíticamente) se hará uso de la función quiver, que hace parte de la librería matplotlib.pyplot.

Campo de pendientes con Python

Ejemplo 8. Represente el campo de pendientes de la siguiente ecuación diferencial:

$$\frac{dy}{dx} = 2\frac{y}{x}$$

Solución:

```
import numpy as np #importa la libreria numpy y la renombra para invocarla como np
import matplotlib.pyplot as plt #importa la libreria matplotlib y la renombra para invocarla como plt

# Ecuacion Diferencial: dy/dx=f(x,y)
def f(x,y):
    return 2*y/x
N=20
x=np.linspace(-10,10,N)  # Si se hace N muy grande no se alcanzan a apreciar las pendientes
y=np.linspace(-10,10,N)  #Esto es porque habria muy poco espacio entre puntos concecutivos
X,Y = np.meshgrid( x,y ) # Se genera una malla de forma similar a las gráficas de funciones implícitas
U = 1; V = f(X,Y);
U2, V2 = U/np.sqrt(U**2+V**2), V/np.sqrt(U**2+V**2)
fig=plt.figure(figsize=(10,6))
plt.quiver( X,Y,U2, V2,color='k') # se grafica el campo de pendientes
plt.grid(True)
plt.title(r'Campo de pendientes: $\frac{dy}{dx}=2 \frac{y}{x}$')
```

Campo de pendientes con Python

Ejemplo 8. Represente el campo de pendientes de la siguiente ecuación diferencial:

$$\frac{dy}{dx} = 2\frac{y}{x}$$

Solución:

```
import numpy as np #importa la libreria numpy y la renombra para invocarla como np
import matplotlib.pyplot as plt #importa la libreria matplotlib y la renombra para invocarla como plt

# Ecuacion Diferencial: dy/dx=f(x,y)
def f(x,y):
    return 2*y/x
N=20
x=np.linspace(-10,10,N)  # Si se hace N muy grande no se alcanza
y=np.linspace(-10,10,N)  #Esto es porque habria muy poco espacio
X,Y = np.meshgrid( x,y ) # Se genera una malla de forma similar
U = 1; V = f(X,Y);
U2, V2 = U/np.sqrt(U**2+V**2), V/np.sqrt(U**2+V**2)
fig=plt.figure(figsize=(10,6))
plt.quiver( X,Y,U2, V2,color='k') # se grafica el campo de pendi
plt.grid(True)
plt.title(r'Campo de pendientes: $\frac{dy}{dx}=2 \frac{y}{x}$')
```

Esta es la parte que realmente cambiaria si se tuviera otra EDO.
Si la ED fuera por ejemplo:

$$\frac{dy}{dx} = xe^{-x^2}$$

la línea de código seria :

return x*np.exp(x**(-2))

El resto del código podría conservarse igual salvo que se quiera graficar una región mas grande

Campo de pendientes con Python

Ejemplo 8. Represente el campo de pendientes de la siguiente ecuación diferencial:

$$\frac{dy}{dx} = 2\frac{y}{x}$$

Solución:

```
import numpy as np #importa la libreria numpy y la renombra para invocarla como np
import matplotlib.pyplot as plt #importa la libreria matplotlib y la renombra para invocarla como plt

# Ecuacion Diferencial: dy/dx=f(x,y)
def f(x,y):
    return 2*y/x
N=20
x=np.linspace(-10,10,N)  # Si se hace N muy grande no se alcanzan a apreciar las pendientes
y=np.linspace(-10,10,N)  #Esto es porque habria muy poco espacio entre puntos concecutivos
X,Y = np.meshgrid( x,y ) # Se genera una malla de forma similar a las gráficas de funciones implícitas
U = 1; V = f(X,Y);
U2, V2 = U/np.sqrt(U**2+V**2), V/np.sqrt(U**2+V**2)
fig=plt.figure(figsize=(10,6))
plt.quiver( X,Y,U2, V2,color='k') # se grafica el campo de pendientes
plt.grid(True)
plt.title(r'Campo de pendientes: $\frac{dy}{dx}=2 \frac{y}{x}$')
```

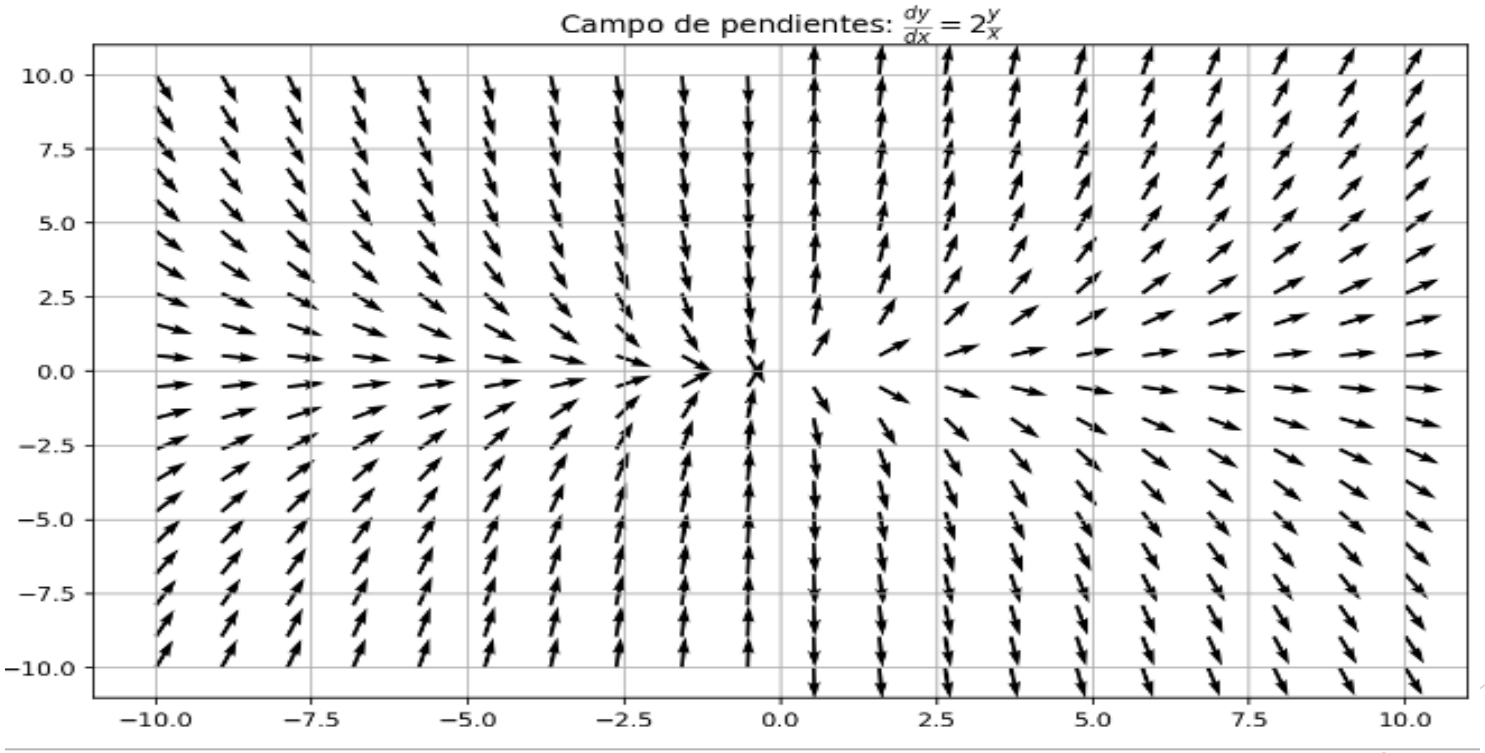
Esta línea se realiza una normalización para que todas la flechas queden con el mismo tamaño, lo cual quiere decir que no es estrictamente necesaria, solo se hace con motivos de mejorar la representación gráfica

Campo de pendientes con Python

Ejemplo 8. Represente el campo de pendientes de la siguiente ecuación diferencial:

$$\frac{dy}{dx} = 2\frac{y}{x}$$

Resultado:



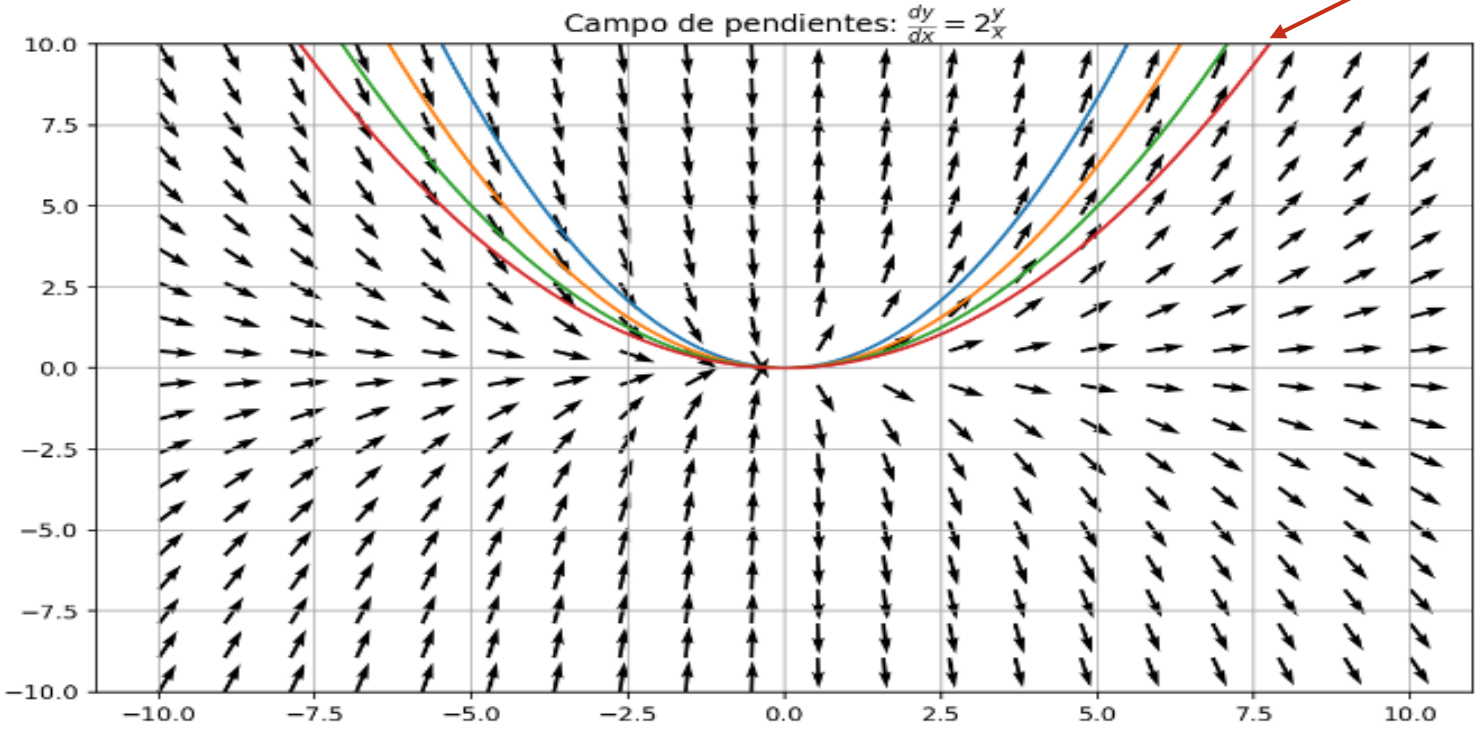
Campo de pendientes con Python

Ejemplo 8. Represente el campo de pendientes de la siguiente ecuación diferencial:

$$\frac{dy}{dx} = 2\frac{y}{x}$$

La solución general de la edo es: $y = Cx^2$

Se puede observar que la soluciones son de la forma:



Campo de pendientes con Maxima

Ejemplo 9. Represente el campo de pendientes de la siguiente ecuación diferencial:

$$\frac{dy}{dx} = -\frac{y}{x}$$

Para graficar el campo de pendientes de una edo en Maxima se puede hacer uso de las instrucciones plotdf o drawdf (esto depende de la versión de Maxima).

Esta función no esta implementada en el compilador online, por lo cual se recomienda descargar el programa para pc o la aplicación para Android

Campo de pendientes con Maxima

Ejemplo 9. Represente el campo de pendientes de la siguiente ecuación diferencial:

$$\frac{dy}{dx} = -\frac{y}{x}$$

Solución:

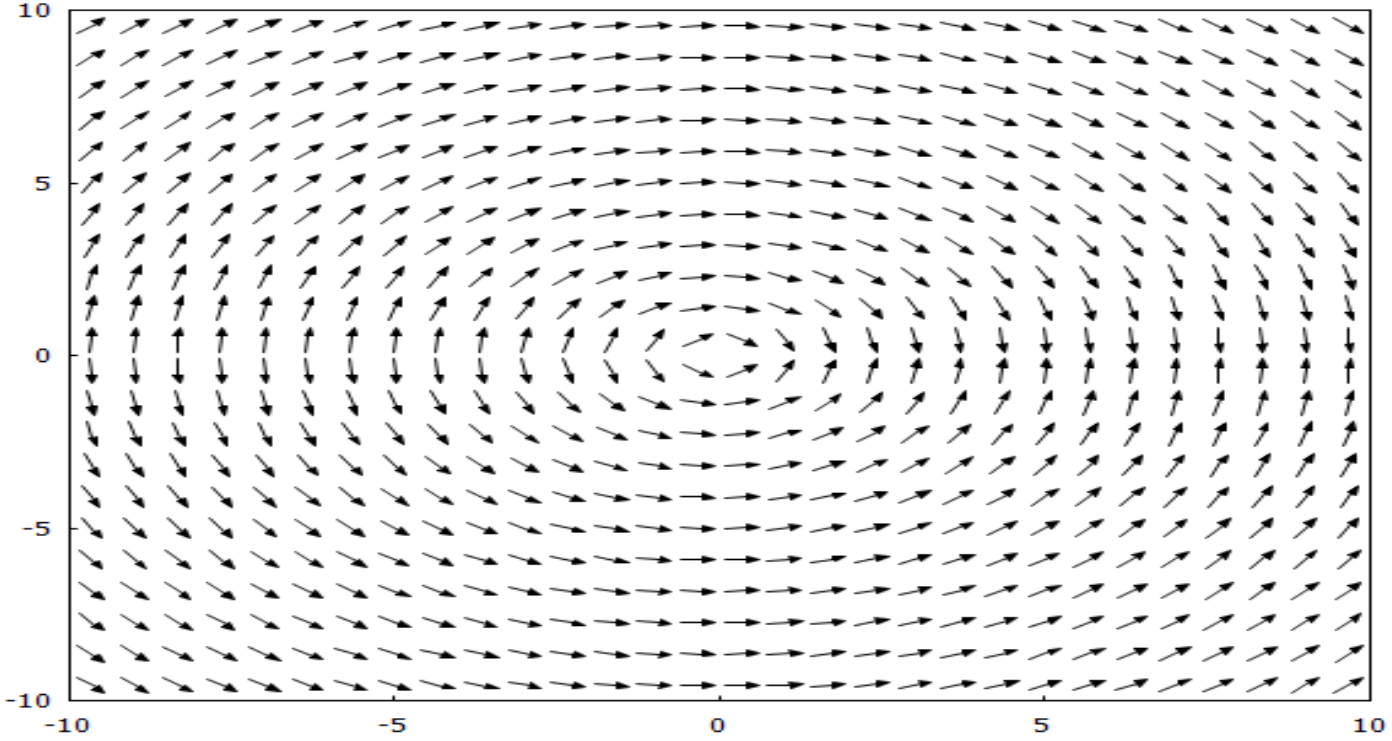
```
load(drawdf)      # Se debe "importar" la función drawdf
drawdf(-y/x)      # Se le entrega el lado derecho de una
                    expresión de la forma dy/dx = f(x,y)
```

Campo de pendientes con Maxima

Ejemplo 9. Represente el campo de pendientes de la siguiente ecuación diferencial:

$$\frac{dy}{dx} = -\frac{y}{x}$$

Resultado:



Solución de una EDO con MatLab

Para resolver analíticamente una EDO en MatLab se hace uso del comando dsolve(), que es una función que recibe como argumento la ecuación diferencial expresada como una igualdad (con una determinada notación). De forma opcional recibe las condiciones iniciales del problema.

La expresión que se ingresa como argumento es una expresión de la forma

```
grid on
```

$$Dy + ay + bx = c$$

Donde:

$$\mathrm{D} = \frac{dy}{dx}$$

La sintaxis será

dsolve('Dy+ay+bx=c','$y(x_0) = y_0$','x')

Solución de una EDO con MatLab

Para resolver analíticamente una EDO en MatLab se hace uso del comando dsolve(), que es una función que recibe como argumento la ecuación diferencial expresada como una igualdad (con una determinada notación). De forma opcional recibe las condiciones iniciales del problema.

La expresión que se ingresa como argumento es una expresión de la forma

$$Dy + ay + bx = c$$

Donde:

$$\mathrm{D} = \frac{dy}{dx}$$

La sintaxis será

dsolve('Dy+ay+bx=c','$y(x_0) = y_0$','x')

Cuando no se indica el valor inicial en el comando dsolve este retorna la solución general de la EDO.

Solución de una EDO con MatLab

Para resolver analíticamente una EDO en MatLab se hace uso del comando dsolve(), que es una función que recibe como argumento la ecuación diferencial expresada como una igualdad (con una determinada notación). De forma opcional recibe las condiciones iniciales del problema.

La expresión que se ingresa como argumento es una expresión de la forma

$$Dy + ay + bx = c$$

Donde:

$$\mathrm{D} = \frac{dy}{dx}$$

Esto indica que la ecuación diferencial se resuelve para la variable x

La sintaxis será

dsolve('Dy+ay+bx=c','$y(x_0) = y_0$','x')

Solución de una EDO con MatLab

Ejemplo 10. Resuelva el siguiente problema de valor inicial:

$$\frac{dy}{dx} = 2\frac{y}{x}\ con\ y(1) = 0$$

Solución de una EDO con MatLab

Ejemplo 10. Resuelva el siguiente problema de valor inicial.

$$\frac{dy}{dx} = 2\frac{y}{x} \; con \; y(1) = 0$$

Solución:

```
y=dsolve('Dy-2*y/x=4*x^2','y(1)=0','x'); % Resuelve la EDO
ys=simplify(y); % Simplifica la ecuación resultante
pretty(ys) %cambia la notación de impresión en pantalla
ezplot(ys,[-2,2])
grid on
```

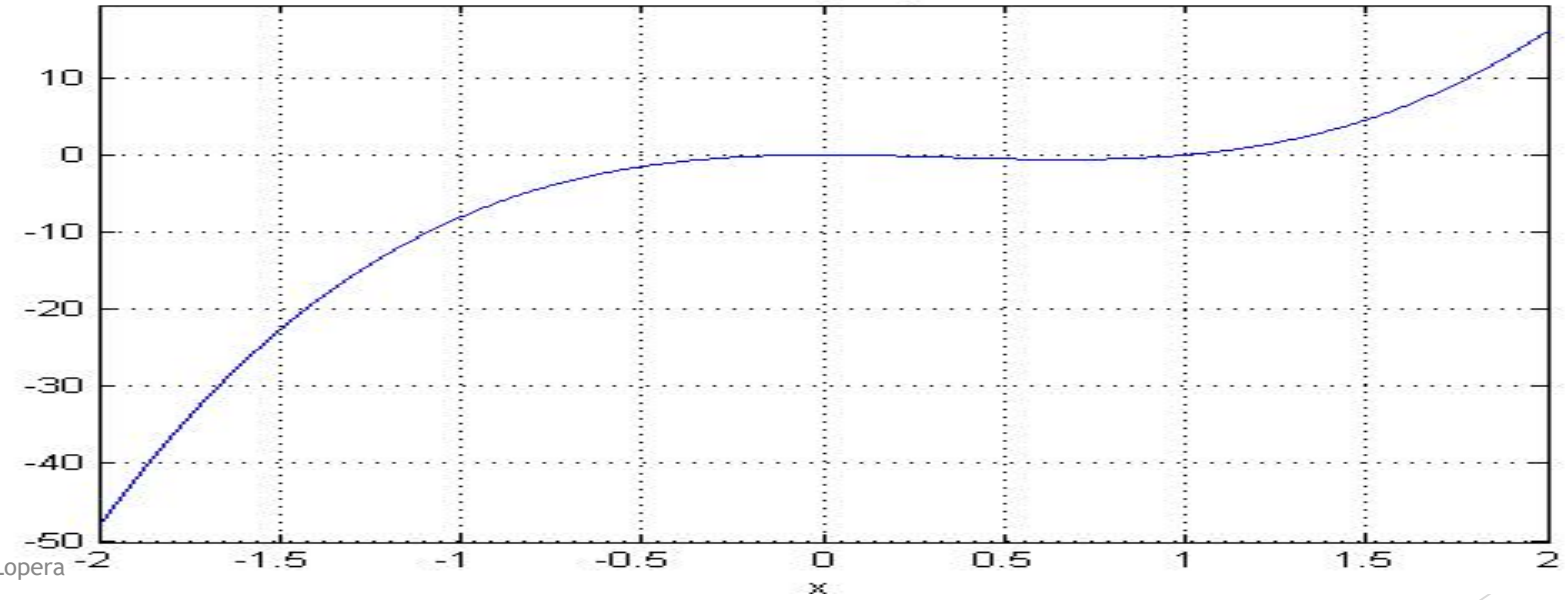
Solución de una EDO con MatLab

Ejemplo 10. Resuelva el siguiente problema de valor inicial:

$$\frac{dy}{dx} = 2\frac{y}{x} \; con \; y(1) = 0$$

Resultado:

```
                2
>> 4 x    (x - 1)
```



Solución de una EDO de primer orden con MatLab

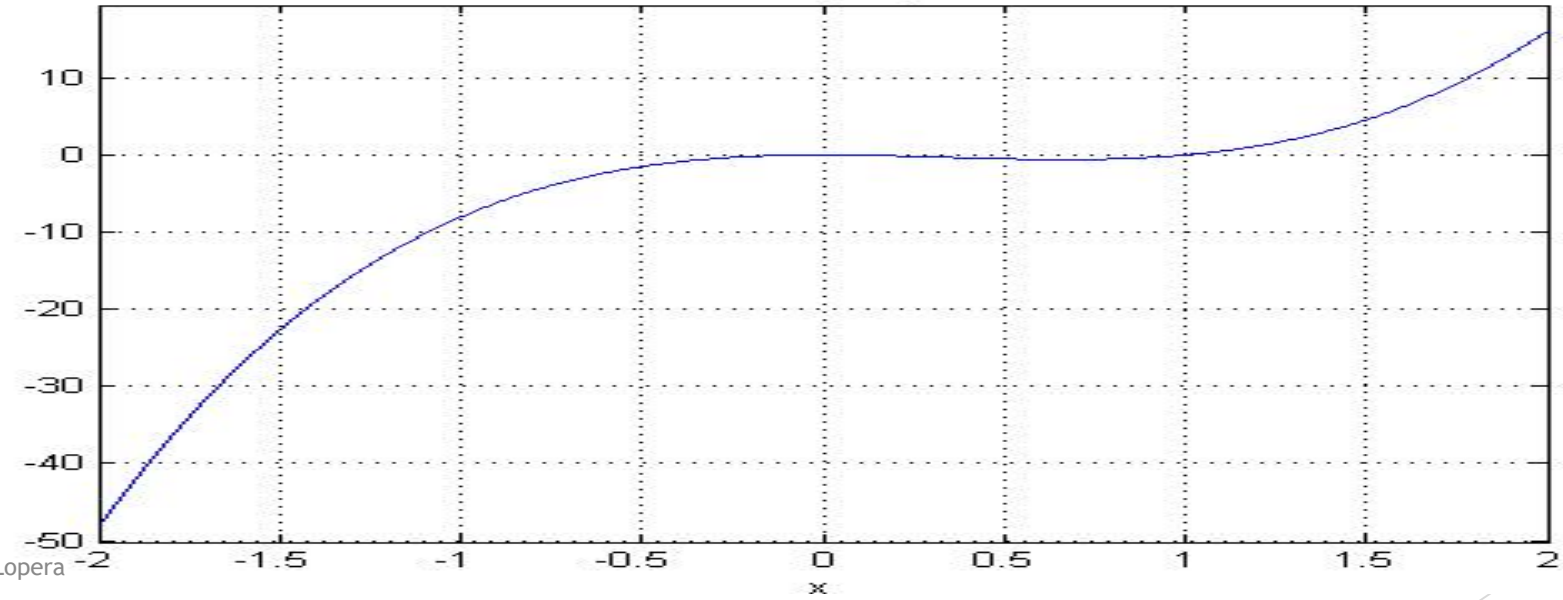
Ejemplo 10. Resuelva el siguiente problema de valor inicial:

$$\frac{dy}{dx} = 2\frac{y}{x} \; con \; y(1) = 0$$

Al usar la instrucción pretty(), matlab muestra el resultado en este formato

Resultado:

```
                2
>> 4 x   (x - 1)
```



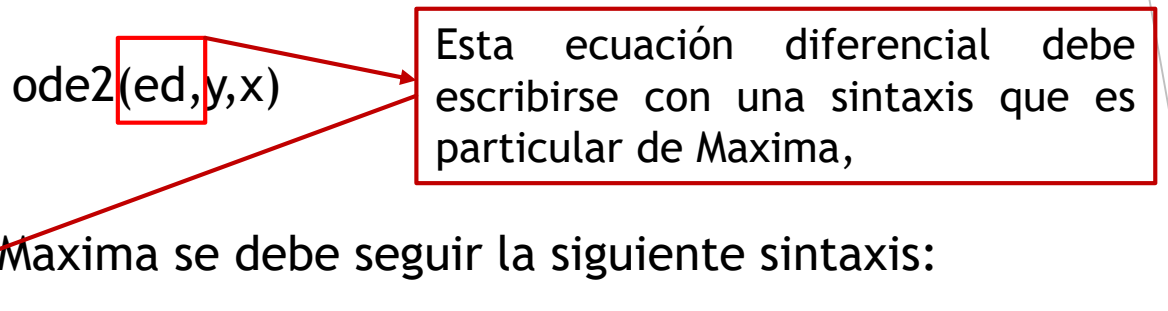
Solución de una EDO de primer orden con Maxima

Para resolver una ecuación diferencial ordinaria con Maxima se puede hacer uso de la función ode2(), la cual tiene la siguiente sintaxis:

ode2(ed,y,x)

Solución de una EDO de primer orden con Maxima

Para resolver una ecuación diferencial ordinaria con Maxima se puede hacer uso de la función ode2(), la cual tiene la siguiente sintaxis:



Para representar una derivada en Maxima se debe seguir la siguiente sintaxis:

```
'diff(y,x,n)    #La comilla (') es necesaria
```

El primer y segundo argumento son la variable dependiente e independiente respectivamente.

El tercer argumento representa el orden de la derivada.

El comando ode2() entrega la solución general de la EDO, por lo cual para resolver un PVI debe evaluarse la solución general con el comando ic1().

Solución de una EDO de primer orden con Maxima

Ejemplo 11. Resuelva el siguiente PVI:

$$\frac{dy}{dx} + 2y = e^{-x} + x^2 + \sin(x) \qquad con\ y(0) = 0$$

Solución de una EDO de primer orden con Maxima

Ejemplo 11. Resuelva el siguiente PVI:

$$\frac{dy}{dx} + 2y = e^{-x} + x^2 + \sin(x) \qquad con\ y(0) = 0$$

Solución:

```
ed:'diff(y,x,1)+2*y=%e-x+x^2+sin(x); # La ecuación diferencial se
                                       expresa como una igualdad

sln:ode2(ed,y,x);

y:ic1(sln,x=0,y=0);      # ic1 se usa para calcular el valor de la
                           constante C cuando la ecuación diferencial es
                           de orden 1
```

Solución de una EDO de primer orden con Maxima

Ejemplo 11. Resuelva el siguiente PVI:

$$\frac{dy}{dx} + 2y = e^{-x} + x^2 + \sin(x) \qquad con\ y(0) = 0$$

Resultado:

$$y = \frac{\%e^{-2x}\,(4\,\%e^{2x}\sin(x) - 2\,\%e^{2x}\cos(x) + (5x^2 - 10x + 5\,\%e + 5)\,\%e^{2x} - 5\,\%e - 3)}{10}$$