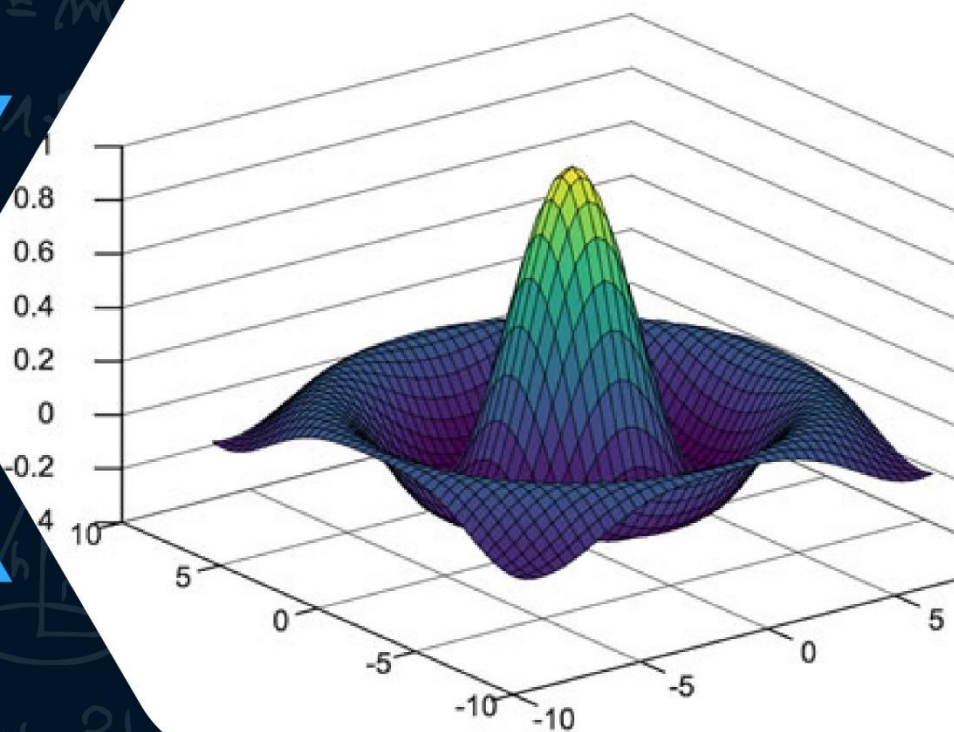








Programación de Métodos Numéricos con

Octave


Eidéc
EDITORIAL



6370 km $U = \varepsilon \cdot d$ $v = h \cdot m$ $\sigma = h \cdot \pi$ $\vec{G} = \frac{\Delta \vec{r}}{\Delta t}$ $f = \frac{1}{T}$
 $\frac{2\pi}{T}$ $J = I \cdot \omega$ $v = abh$ $C_p = C_v + R$ $p = \frac{m}{V}$ $\lambda = v \cdot T$
 $v = \frac{4}{3} \pi r^3$ $W = -p \cdot \Delta V$ $\vec{F}_s = -k \cdot \vec{x}$ $\omega = \frac{\Delta \theta}{\Delta t}$
 $\frac{\Delta W}{\Delta t}$ $U = \varepsilon \cdot d$ $v = P \cdot h$ $\Delta U = Q + W$  $R_2 = 6370$
 $p + q = 1$ $\lambda = \frac{h}{m \cdot v}$  $\sigma = \omega \cdot r$
 $v = h \cdot m$ $v = \pi r^2 \cdot h$ $W = -p \cdot \Delta V$ $v = abh$ $a^2 + b^2 = c^2$
 $\sqrt{\frac{m}{k}}$ $T = 2\pi \sqrt{\frac{l}{g}}$ $J = I \cdot \omega$ $\frac{T_1^2}{R_1^3} = \frac{T_2^2}{R_2^3} = \text{const}$ $\sigma_1 = \sqrt{\frac{G \cdot M_2}{R_2}}$
 $v = h \cdot m$ $v = abh$ $E = m \cdot c^2$ $p =$
 $= m \cdot c^2$ $U = \frac{W}{q}$ $g = 9.81 \frac{m}{s^2}$ $R_2 = 6370 \text{ km}$ $I = \sum_{i=1}^3 m_i \cdot r_i^2$
 $s = v_0 \cdot t + \frac{1}{2} a \cdot t^2$ $f = \frac{R}{2}$ $v = \frac{4}{3} \pi r^3$ $f = \frac{R}{2}$ $\sin \alpha = \frac{a}{c}$
 $R_2 = 6370 \text{ km}$ $W = -p \cdot \Delta V$ $g = 9.81 \frac{m}{s^2}$
 $\frac{2\pi}{T}$ $J = I \cdot \omega$   $C_p = C_v + R$ $\vec{G} = \frac{\Delta \vec{r}}{\Delta t}$
 $v = \frac{4}{3} \pi r^3$ $\vec{F}_s = -k \cdot \vec{x}$ $p = \frac{m}{V}$
 $E = \frac{\Delta W}{\Delta t}$ $U = \varepsilon \cdot d$ $v = P \cdot h$ $\Delta U = Q + W$  ω
 $9 \frac{\text{km}}{s}$ $p + q = 1$ $\lambda = \frac{h}{m \cdot v}$  $\sigma = \omega \cdot r$
 $v = h \cdot m$ $v = \pi r^2 \cdot h$ $W = -p \cdot \Delta V$ $v = abh$ $a^2 + b^2 = c^2$
 $W = \frac{1}{2} Q \cdot U$

PROGRAMACIÓN DE MÉTODOS NUMÉRICOS CON OCTAVE

COLECCIÓN RESULTADO DE INVESTIGACIÓN

Primera Edición 2024 Vol. 1

Editorial EIDEC

Sello Editorial EIDEC (978-958-53018)

NIT 900583173-1

ISBN: 978-628-96378-9-2

Formato: Digital PDF (Portable Document Format)

DOI: doi.org/10.34893/z9449-8029-0760-d

Publicación: Colombia

Fecha Publicación: 31/10/2024

Coordinación Editorial

Escuela Internacional de Negocios y Desarrollo Empresarial de Colombia – EIDEC

Centro de Investigación Científica, Empresarial y Tecnológica de Colombia – CEINCET

Red de Investigación en Educación, Empresa y Sociedad – REDIEES

Revisión y pares evaluadores

Centro de Investigación Científica, Empresarial y Tecnológica de Colombia – CEINCET

Red de Investigación en Educación, Empresa y Sociedad – REDIEES

**Eidéc**
EDITORIAL

Coordinadores editoriales

Paula Alejandra Noguera Zambrano

Editorial EIDEC

Dr. Cesar Augusto Silva Giraldo

Centro de Investigación Científica, Empresarial y Tecnológica de Colombia – CEINCET – Colombia.







Dr. David Andrés Suarez Suarez

Red de Investigación en Educación, Empresa y Sociedad – REDIEES – Colombia.

El libro **PROGRAMACIÓN DE MÉTODOS NÚMERICOS CON OCTAVE**, está publicado bajo la licencia de Creative Commons Atribución-NoComercial 4.0 Internacional (CC BY-NC 4.0) Internacional (<https://creativecommons.org/licenses/by-nc/4.0/deed.es>). Esta licencia permite copiar, adaptar, redistribuir y reproducir el material en cualquier medio o formato, con fines no comerciales, dando crédito al autor y fuente original, proporcionando un enlace de la licencia de Creative Commons e indicando si se han realizado cambios.

Licencia: CC BY-NC 4.0.

NOTA EDITORIAL: Las opiniones y los contenidos publicados en el libro **PROGRAMACIÓN DE MÉTODOS NÚMERICOS CON OCTAVE**, son de responsabilidad exclusiva de los autores; así mismo, éstos se responsabilizarán de obtener el permiso correspondiente para incluir material publicado por parte de la **Editorial EIDEC**.

6370 km $U = \varepsilon \cdot d$ $v = h \cdot m$ $\sigma = h \cdot \pi$ $\vec{G} = \frac{\Delta \vec{r}}{\Delta t}$ $f = \frac{1}{T}$
 $\frac{2\pi}{T}$ $J = I \cdot \omega$ $v = abh$ $C_p = C_v + R$ $p = \frac{m}{V}$ $\lambda = v \cdot T$
 $v = \frac{4}{3} \pi r^3$ $W = -p \cdot \Delta V$ $\vec{F}_s = -k \cdot \vec{x}$ $\omega = \frac{\Delta \theta}{\Delta t}$
 $\frac{\Delta W}{\Delta t}$ $U = \varepsilon \cdot d$ $v = P \cdot h$ $\Delta U = Q + W$  $R_2 = 6370$
 $p + q = 1$ $\lambda = \frac{h}{m \cdot v}$  $\sigma = \omega \cdot r$
 $v = h \cdot m$ $v = \pi r^2 \cdot h$ $W = -p \cdot \Delta V$ $v = abh$ $a^2 + b^2 = c^2$
 $\sqrt{\frac{m}{k}}$ $T = 2\pi \sqrt{\frac{l}{g}}$ $J = I \cdot \omega$ $\frac{T_1^2}{R_1^3} = \frac{T_2^2}{R_2^3} = \text{const}$ $\sigma_1 = \sqrt{\frac{G \cdot M_2}{R_2}}$
 $v = h \cdot m$ $v = abh$ $E = m \cdot c^2$ $p =$
 $= m \cdot c^2$ $U = \frac{W}{q}$ $g = 9.81 \frac{m}{s^2}$ $R_2 = 6370 \text{ km}$ $I = \sum_{i=1}^3 m_i \cdot r_i^2$
 $s = v_0 \cdot t + \frac{1}{2} a \cdot t^2$ $f = \frac{R}{2}$ $v = \frac{4}{3} \pi r^3$ $f = \frac{R}{2} \sin \alpha = \frac{a}{c}$
 $R_2 = 6370 \text{ km}$ $W = -p \cdot \Delta V$ $g = 9.81 \frac{m}{s^2}$
 $\frac{2\pi}{T}$ $J = I \cdot \omega$   $C_p = C_v + R$ $\vec{G} = \frac{\Delta \vec{r}}{\Delta t}$
 $v = \frac{4}{3} \pi r^3$ $\vec{F}_s = -k \cdot \vec{x}$ $p = \frac{m}{V}$
 $E = \frac{\Delta W}{\Delta t}$ $U = \varepsilon \cdot d$ $v = P \cdot h$ $\Delta U = Q + W$  ω
 $9 \frac{\text{km}}{s}$ $p + q = 1$ $\lambda = \frac{h}{m \cdot v}$  $\sigma = \omega \cdot r$
 $v = h \cdot m$ $v = \pi r^2 \cdot h$ $W = -p \cdot \Delta V$ $v = abh$ $a^2 + b^2 = c^2$
 $W = \frac{1}{2} Q \cdot U$

PROGRAMACIÓN DE MÉTODOS NÚMERICOS CON OCTAVE PROGRAMMING NUMERICAL METHODS WITH OCTAVE

Rolando Elias Ipanaque Silva¹

Moisés David Saavedra Arango²

Cesar Silva More³

Eder Escobar Gómez⁴

Hoover Augusto Puicón Zapata⁵

Luis Armando Saavedra Yarleque⁶

Karen Vanessa Arias Abramonte⁷

Pares evaluadores: Red de Investigación en Educación, Empresa y Sociedad – REDIEES⁸

¹ Universidad Nacional de Piura, ripanaques@unp.edu.pe

² Universidad Nacional de Piura.

³ Universidad Nacional de Piura.

⁴ Universidad Nacional de Piura.

⁵ Universidad Nacional de Piura.

⁶ Universidad Nacional de Piura.

⁷ Universidad tecnológica del Perú.

⁸ Red de Investigación en Educación, Empresa y Sociedad – REDIEES. www.rediees.org

INDICE GENERAL

CAPITULO I. GRÁFICAS EN OCTAVE.....	14
2.1 Gráficas en el plano bidimensional.....	14
1.2 Gráfico de múltiples curvas	19
1.3 Etiquetas en los ejes coordenados.....	20
1.4 Gráficas en múltiples ejes	20
1.5 Gráficos en el plano	21
1.6 Gráficos en el sistema cartesiano tridimensional.....	23
1.7 Gráficos especiales	28
1.8 Gráficas en coordenadas polares.....	30
1.9 Gráfica de un campo vectorial 2D.....	32
Taller N° 01 gráficos en 2D y 3D	32
Solución del taller N° 01 gráficos en 2D y 3D	34
CAPITULO II. CALCULO DIFERENCIAL E INTEGRAL	37
2.1 Limite de una función de variable real.	37
2.2 Limite de una función a trozos.	39
2.3 Los Límites al infinito.....	42
Taller N° 02 de Limites	43
Solución del taller N° 02 con octave	44
2.4 Derivada de una función real en un punto	46
2.5 Derivadas laterales.....	47
2.6 Derivadas de orden superior	48
Taller N° 03 derivadas	50
Solución del taller N° 03 derivadas con octave	51
2.7 Integral indefinida.....	54
2.8 Integral definida.....	54
2.8 Integrales impropias.....	55
2.8.1 Comandos similares a quad.	56
Taller N° 04 Integrales.....	56

Solución del taller N° 04 Integrales con octave.... 558

CAPITULO III. ECUACIONES DIFERENCIALES LINEALES 62

3.1 Ecuaciones diferenciales ordinarias..... 62

3.2 El comando Isode de Octave 62

3.3 Método de Euler..... 71

3.4 Método de Taylor de orden n..... 73

3.5 Método de Runge Kutta..... 75

3.5.1 Método del punto medio 75

3.5.2 Método modificado de Euler..... 75

3.5.3 Método de Heun 75

3.5.4 Método de RK de orden cuatro. 75

3.6 Ecuación diferencial ordinaria de segundo orden76

3.7 Ecuación diferencial ordinaria de tercer orden . 77

3.8 Ecuación diferencial ordinaria de cuarto orden 77

Taller N° 05 EDO con octave 80

CAPITULO IV. ECUACIONES DIFERENCIALES NO LINEALES. 81

4.1 Modelo de proyección de Velezmoro. 81

4.1.1 Modelo de proyección 4D..... 81

4.2. Sistema de Lorenz 4D..... 82

4.3. Algoritmo para graficar diagramas de fase de un sistema no lineal en 4D. 83

4.4. Programa SystemLorenz4D. 83

4.5 Aplicación del algoritmo al nuevo sistema de Lorenz 4D. 84

4.6 Análisis de estabilidad en el origen..... 85

REFERENCIAS BIBLIOGRAFICAS 89

Prólogo

Este libro titulado “programación de métodos numéricos con Octave” se elaboró con el propósito de colaborar en el aprendizaje de nuevos conocimientos en los estudiantes del nivel de pregrado de las diferentes carreras de Ciencias e Ingeniería. Los lectores en el capítulo I aprenderán a graficar diferentes funciones en el plano y el espacio con los comandos que posee Octave. En el capítulo II aprenderán a calcular el valor del límite, derivada, integral indefinida y la integral definida de funciones reales, mediante definiciones y propiedades, para luego constatar los resultados con las funciones incorporadas que posee Octave. En el capítulo III se enseña como aproximar numéricamente una ecuación diferencial ordinaria lineal con los diferentes comandos que posee Octave. El capítulo IV enseña como aproximar soluciones numéricamente del sistema no lineal de Lorenz en la cuarta dimensión y la visualización de diagramas de fase, haciendo uso de los comandos que posee el software Octave.

Al termino de cada capítulo se presenta un listado de ejercicios con su respectiva solución con ayuda del software Octave.

PROGRAMACIÓN DE MÉTODOS NUMÉRICOS CON OCTAVE

PROGRAMMING NUMERICAL METHODS WITH OCTAVE

RESUMEN

En este libro se han considerado cuatro capítulos que le serán muy útiles a los estudiantes de pregrado de las diferentes carreras de ciencias e ingeniería. En el primer y segundo ciclo de estudios universitarios a los estudiantes se les enseña como graficar funciones univariadas y también como calcular el límite, la derivada e integral de funciones reales, hoy en día existen diferentes softwares que ayudan a graficar y calcular el límite, la derivada e integral de dichas funciones y una de ellas es Octave. Este software es gratuito para todos los amantes de la programación y está disponible en google, se puede descargar de la siguiente forma, escribir en google **octave.org**. El objetivo del primer capítulo es utilizar funciones de Octave para elaborar gráficos en dos y tres dimensiones. En el segundo capítulo se desarrollarán límites, derivadas e integrales de funciones reales con los comandos apropiados que posee Octave. El objetivo del tercer capítulo es aproximar numéricamente la solución de una EDO. El cuarto capítulo trata sobre aproximación numérica de la solución del sistema no lineales de Lorenz en la cuarta dimensión y su visualización de los diagramas de fase. Esperamos con esta monografía hecha, aportar en los conocimientos básicos a los lectores.

PALABRAS CLAVES: Métodos numéricos, Sistemas no lineales, Cuarta dimensión, Octave.

ABSTRACT

In this book we have considered four chapters that will be very useful to undergraduate students of different careers in science and engineering. In the first and second cycle of university studies students are taught how to graph univariate functions and also how to calculate the limit, derivative and integral of real functions, nowadays there are different software that help to graph and calculate the limit, derivative and integral of such functions and one of them is Octave. This software is free for all programming enthusiasts and is available on google, you can download it as follows, type in google octave.org. The objective of the first chapter is to use Octave functions to develop graphs in two and three dimensions. In the second chapter limits, derivatives and integrals of real functions will be developed with the appropriate Octave commands. The objective of the third chapter is to numerically approximate the solution of an ODE. The fourth chapter deals with numerical approximation of the solution of the nonlinear Lorentz system in the fourth dimension and its visualization of the phase diagrams. We hope that this monograph will provide basic knowledge to the readers.

KEYWORDS: Numerical methods, Nonlinear systems, Fourth dimension, Octave.

INTRODUCCIÓN

En la actualidad, la utilización de computadoras es común en el quehacer diario de las personas del siglo XXI, estas máquinas han reducido el uso del bolígrafo y papel, además se puede guardar y adquirir grandes cantidades de información en un lapso reducido.

Hoy en día existe una variedad de software libres que ayudan a solucionar problemas matemáticos, para apoyarnos de estos softwares debemos estar en una constante capacitación y actualización debido a los avances tecnológicos en las computadoras.

El software Octave es una herramienta fundamental para ciencias e ingeniería que desarrolla elementos del cálculo matricial, métodos numéricos, ecuaciones diferenciales y gráficas de funciones.







Todo estudiante de ciencias e ingeniería lleva cursos básicos: como calculo I del autor Espinoza ramos (2000), de este libro se han extraído ejercicios de límites, derivadas de una función. Calculo II del autor Mitacc & Toro (1992), de este libro se han extraído ejercicios de integrales. Análisis matemático IV del autor Espinoza Ramos (2002), de este libro se ha extraído definiciones y propiedades de ecuaciones diferenciales ordinarias y de la tesis titulada “comportamientos cualitativos de un sistema no lineal de Lorenz en 4D y 5D apoyados gráficamente con software octave” de Escobar Gómez (2023), se ha extraído la sección que trata del sistema no lineal de Lorenz en 4D. La presente monografía presenta 4 capítulos ejemplificando paso a paso el uso de los comandos que posee Octave.

Capítulo 1. Gráficas en Octave.

Capitulo II. Calculo diferencial e integral.

Capitulo III. Ecuaciones diferenciales ordinarias lineales

Capitulo IV. Sistema de ecuaciones diferenciales ordinarias no lineal.

6370 km $U = \epsilon \cdot d$ $v = h \cdot m$ $\sigma = h \cdot \pi$ $\vec{G} = \frac{\Delta \vec{r}}{\Delta t}$ $f = \frac{1}{T}$
 $\frac{2\pi}{T}$ $J = I \cdot \omega$ $v = abh$ $\lambda = v \cdot T$
 $v = \frac{4}{3} \pi r^3$ $W = -p \cdot \Delta V$ $\vec{F}_s = -k \cdot \vec{x}$ $p = \frac{F}{A}$ $\omega = \frac{\Delta \theta}{\Delta t}$
 $\frac{\Delta W}{\Delta t}$ $U = \epsilon \cdot d$ $v = P \cdot h$ $\Delta U = Q + W$  $R_2 = 6370$
 $p + q = 1$ $\lambda = \frac{h}{m \cdot v}$  $\sigma = \omega \cdot r$
 $v = h \cdot m$ $v = \pi r^2 \cdot h$ $W = -p \cdot \Delta V$ $v = abh$ $a^2 + b^2 = c^2$
 $\sqrt{\frac{m}{k}}$ $T = 2\pi \sqrt{\frac{l}{g}}$ $J = I \cdot \omega$ $\frac{T_1^2}{R_1^3} = \frac{T_2^2}{R_2^3} = \text{const}$ $\sigma_1 = \sqrt{\frac{G \cdot M_2}{R_2}}$
 $v = h \cdot m$ $v = abh$ $E = m \cdot c^2$ $p = \frac{F}{A}$
 $= m \cdot c^2$ $U = \frac{W}{q}$ $g = 9.81 \frac{m}{s^2}$ $R_2 = 6370 \text{ km}$ $I = \sum_{i=1}^n m_i \cdot r_i^2$
 $s = v_0 \cdot t + \frac{1}{2} a \cdot t^2$ $f = \frac{R}{2}$ $v = \frac{4}{3} \pi r^3$ $f = \frac{R}{2}$ $\sin \alpha = \frac{a}{c}$
 $R_2 = 6370 \text{ km}$ $W = -p \cdot \Delta V$ $g = 9.81 \frac{m}{s^2}$
 $\frac{2\pi}{T}$ $J = I \cdot \omega$   $C_p = C_v + R$ $\vec{G} = \frac{\Delta \vec{r}}{\Delta t}$
 km $v = \frac{4}{3} \pi r^3$ $\vec{F}_s = -k \cdot \vec{x}$ $p = \frac{F}{A}$
 $E = \frac{\Delta W}{\Delta t}$ $U = \epsilon \cdot d$ $v = P \cdot h$ $\Delta U = Q + W$  ω
 $9 \frac{\text{km}}{s}$ $p + q = 1$ $\lambda = \frac{h}{m \cdot v}$  $\sigma = \omega \cdot r$
 $v = h \cdot m$ $v = \pi r^2 \cdot h$ $W = \frac{1}{2} Q \cdot U$ $v = abh$ $a^2 + b^2 = c^2$

CAPITULO I. GRÁFICAS EN OCTAVE

2.1 Gráficas en el plano bidimensional

A través de los siguientes comandos, Octave puede crear objetos gráficos en \mathbb{R}^2 y \mathbb{R}^3 e incluso artísticos utilizando expresiones matemáticas.

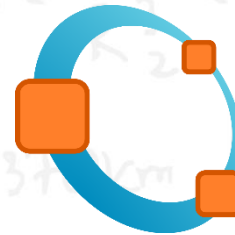
Comandos	Definición
<code>plot(x₁, x₂)</code>	La función plot dibuja un conjunto de puntos (x ₁ , x ₂) en un sistema bidimensional.
<code>bar(x₁, x₂)</code>	La función bar dibuja una secuencia de barras verticales, donde x ₂ representa las frecuencias y x ₁ define los espacios en el eje horizontal.

Objetivo:

Utilizar funciones de octave para elaborar gráficos en el plano y el espacio.

CONTENIDO:

- ✓ Gráfica de múltiples curvas.
- ✓ Gráficas especiales en el plano bidimensional.
- ✓ Gráfica de superficies.
- ✓ Gráfica en coordenadas polares.
- ✓ Gráfica de campos vectoriales 2D y 3D con flechas.



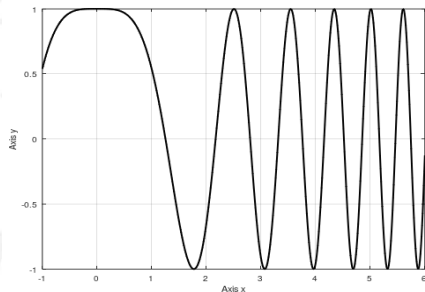
Capítulo

1

<code>barh(x1, x2)</code>	La función <code>barh</code> dibuja una secuencia de barras horizontales, donde x_2 son las frecuencias y x_1 son los espacios en el eje horizontal
<code>stem(x1, x2)</code>	La función <code>stem</code> dibuja una secuencia de bastones en forma vertical, donde x_2 son las frecuencias y x_1 son los espacios en el eje horizontal.
<code>stairs(x1, x2)</code>	La función <code>stairs</code> dibuja una curva de forma escalonada.
<code>polar(x1, x2)</code>	La función <code>polar</code> dibuja curvas en el sistema coordenado polar.
<code>pie(x1)</code>	La función <code>pie</code> grafica sectores en correspondencia al vector x_1 .
<code>rose(x1)</code>	La función <code>rose</code> dibuja histogramas en correspondencia al vector x_2 .

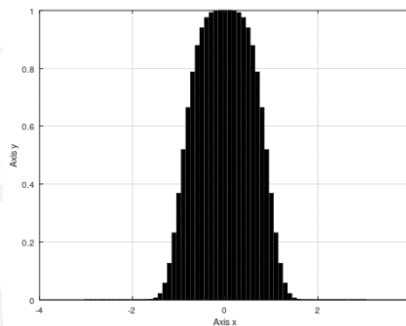
Ejemplo 1.1

```
% line:
x1=-1:0.01:6;
x2=cos(x1.^2);
plot(x1,x2,'k',
'linewidth',1)
grid
xlabel('Axis x')
ylabel('Axis y')
```



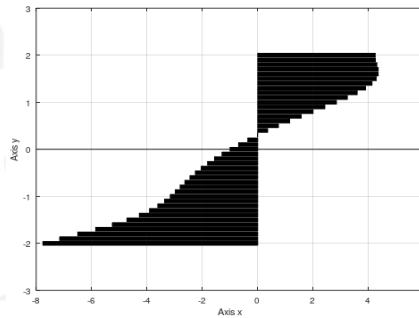
Ejemplo 1.2

```
% bar:
x1=-3:0.1:3;
x2=exp(-x1.^4);
bar(x1,x2,'k')
grid
xlabel('Axis x')
ylabel('Axis y')
```



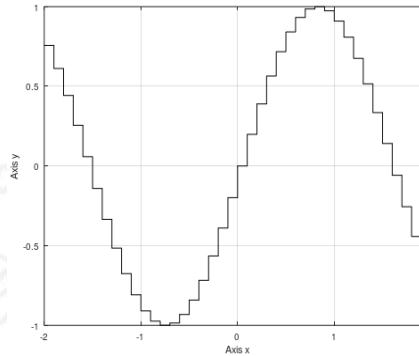
Ejemplo 1.3

```
% barh:  
x1=-2:0.1:2;  
x2=sin(x1.^2)  
+3*x1-1;  
barh(x1,x2,'k');  
grid  
xlabel('Axis x')  
ylabel('Axis y')
```



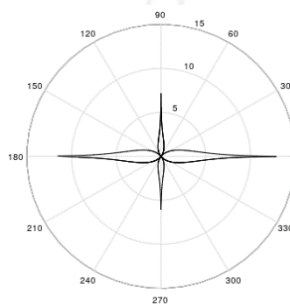
Ejemplo 1.4

```
% stairs:  
x1=0:0.1:15;  
x2=sin(2*x1);  
stairs(x1,x2,'k')  
grid  
xlabel('Axis x')  
ylabel('Axis y')
```



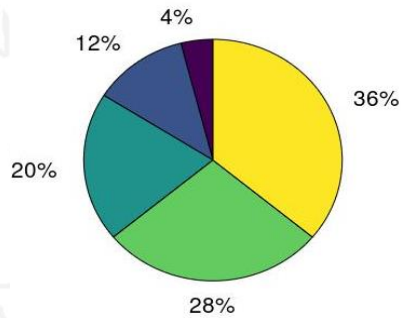
Ejemplo 1.5

```
% polar:  
t=0:0.01:3*pi;  
x1=log(sin(t).*tan(t));  
polar(t,x1,'k')
```



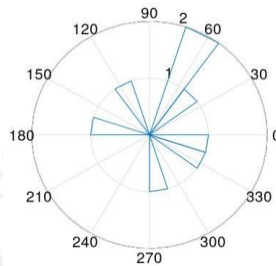
Ejemplo 1.6

% pie:
x1=1:2:9;
pie(x1);



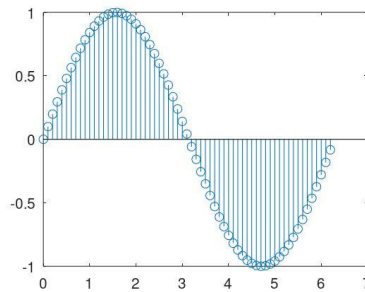
Ejemplo 1.7

% rose:
x1=[1 3 6 2 7 1 5 12];
rose(x1);



Ejemplo 1.8

% stem:
x1=0:0.1:2*pi;
x2=sin(x1);
stem(x1,x2);

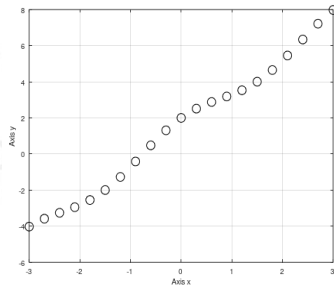


Octave gráfica expresiones matemáticas utilizando diferentes características, como se muestra a continuación.

Caracteres	Definición
+	Más
.	Punto
x	Equis
o	Circunferencia
*	Asterisco

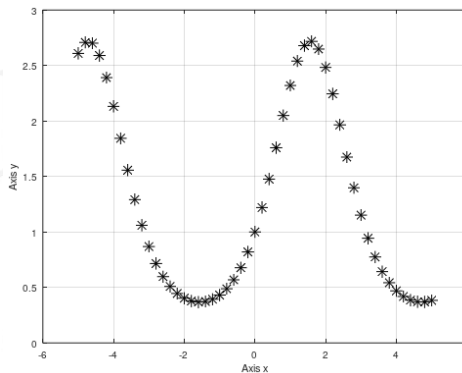
Ejemplo 1.9

```
x1=-3:0.3:3;
x2=cos(x1).^2+2*x1+1;
plot(x1,x2,'ok')
grid
xlabel('Axis x')
ylabel('Axis y')
```



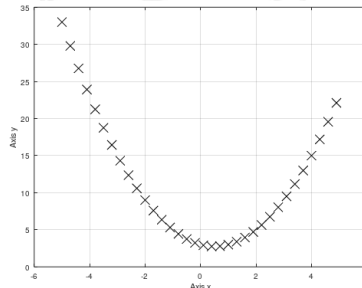
Ejemplo 1.10

```
x1=-5:0.2:5; x2=exp(sin(x1)); plot(x1,x2,'*k'); grid;
xlabel('Axis x'); ylabel('Axis y');
```



Ejemplo 1.11

```
x1=-5:0.3:5;
x2=x1.^2-x1+3;
plot(x1,x2,'xk')
grid
xlabel('Axis x')
ylabel('Axis y')
```



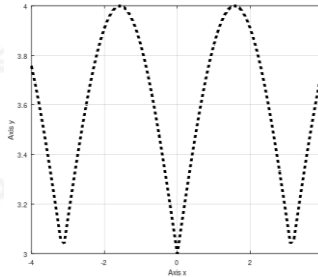
Octave posee muchas variedades en el diseño de gráficas como también una variedad de colores.

Líneas	Definición
--	Guiones
-	Continua
-.	Guiones y punto
:	Punteada

Color	Definición
y	Amarillo
r	Rojo
w	Blanco
b	Azul
g	Verde
k	Negro
c	Turquesa
m	Magenta

Ejemplo 1.12

```
x1=-4:0.1:4;
x2=abs(sin(x1))+3;
plot(x1,x2,'k','linewidth',2)
grid
xlabel('Axis x')
ylabel('Axis y')
```



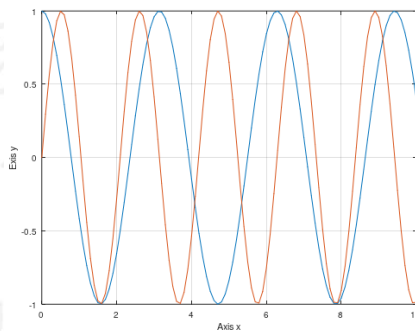
1.2 Gráfico de múltiples curvas

Con el comando **plot** de Octave se puede graficar una o varias curvas simultáneamente, estableciendo por defecto dos colores distintos como se muestra a continuación:

El comando **plot(x1, y1, x1, y2)**: esta función dibuja dos curvas con diferente color.

Ejemplo 1.13

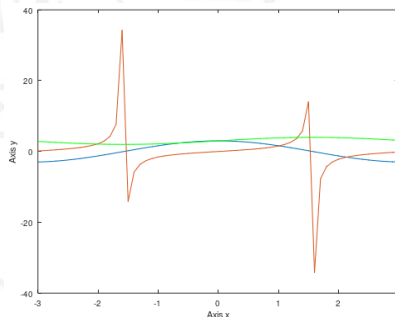
```
x1=0:0.1:10;
y1=cos(2*x1);
y2=sin(3*x1);
plot(x1,y1,x1,y2)
grid
xlabel('Axis x');
ylabel('Esis y')
```



Observación: Después de haber mostrado ejemplos con una variedad de gráficos, a continuación se mostrarán ejemplos donde se interceptan varias curvas en la misma figura, escribiendo la función **hold on**, en el penúltimo **plot** como se muestra a continuación.

Ejemplo 1.14

```
x1=-3:0.1:3;
x2=3*cos(x1);
x3=tan(x1);
plot(x1,x2,x1,x3)
hold on
w=sin(x1)+3;
plot(x1,w,'g')
xlabel('Axis x')
ylabel('Axis y')
```



1.3 Etiquetas en los ejes coordenados

Octave posee una variedad de comandos para colorear textos, etiquetar los ejes, colorear título, como se muestra a continuación:

xlabel: etiqueta al eje horizontal.

ylabel: etiqueta al eje vertical.

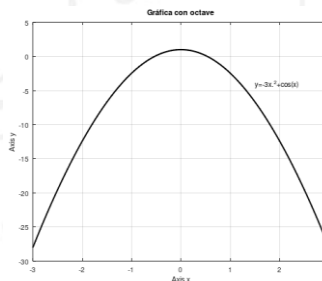
grid: Grafica rejillas (líneas horizontales y verticales) en la gráfica hecha con plot.

title: Coloca título a la gráfica.

text (x, y, 'cadena'); adiciona una cadena a la gráfica en el punto de coordenadas (x, y).

Ejemplo 1.15

```
x1=-3:0.05:3;
x2=-3*x1.^2+cos(x1);
plot(x1,x2,'k','linewidth',1);
grid
text(1.5,-4,'x2=-
3x1.^2+cos(x1)');
title('Gráfica con octave');
xlabel('Axis x')
ylabel('Axis y')
```



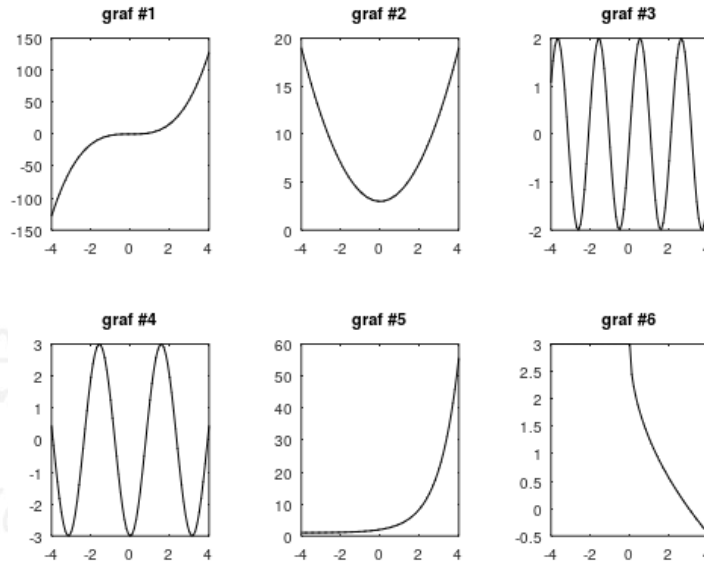
1.4 Gráficas en múltiples ejes

Octave permite la construcción de múltiples ejes en una sola pantalla que produce múltiples gráficos.

subplot (m_fil, n_col, k): Particiona la figura en m por n gráficas, siendo $k = m * n$, el número de gráficas.

Ejemplo 1.16

```
t=-4:0.1:4; x1=2*t.^3; x2=t.^2+3; x3=2*sin(3*t);
u1=-3*cos(2*t); u2=exp(t)+1; u3=3-sqrt(3*t);
subplot(2,3,1), plot(t,x1,'k'),title('graf #1');
subplot(2,3,2), plot(t,x2,'k'),title('graf #2');
subplot(2,3,3), plot(t,x3,'k'),title('graf #3');
subplot(2,3,4), plot(t,u1,'k'),title('graf #4');
subplot(2,3,5), plot(t,u2,'k'),title('graf #5');
subplot(2,3,6), plot(t,u3,'k'),title('graf #6');
```



1.5 Gráficos en el plano

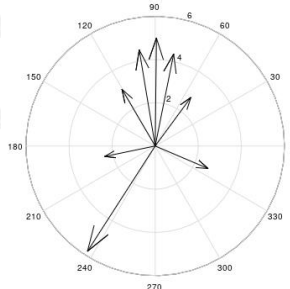
En el siguiente cuadro se mencionan los comandos que Octave tiene incorporados para dibujar funciones reales en el espacio bidimensional.

Comandos	Definición
<code>feather(x₁, x₂)</code>	La función feather dibuja vectores en el plano bidimensional con origen en el eje x_1 . Además, su longitud y dirección vienen dadas por el módulo de x_2 .
<code>loglog(x₁, x₂)</code>	La función loglog dibuja una curva en el plano bidimensional con escala logarítmica en los ejes x_1, x_2 .
<code>ezplot('f', [x₁, x₂])</code>	La función ezplot dibuja en el plano bidimensional la función real F en el intervalo [x, y] y el título lo pone por defecto.
<code>semilogy(x₁, x₂)</code>	La función semilogy dibuja una curva el plano bidimensional. Por defecto presenta escala logarítmica en la segunda componente y escala normal en la primera componente.
<code>semilogx(x₁, x₂)</code>	La función semilogx dibuja una curva en el plano bidimensional con escala logarítmica en la primera componente y escala normal en la segunda componente.
<code>fplot('F', [x₁, x₂])</code>	La función fplot dibuja la función F en el intervalo [x ₁ , x ₂].
<code>fill(x₁, x₂, h)</code>	La función fill grafica una región

	poligonal en el plano bidimensional, donde parámetro h contiene el color a graficar.
<code>compass(x1,x2)</code>	La función <code>compass</code> dibuja vectores tomando como punto de partida el origen de coordenadas.

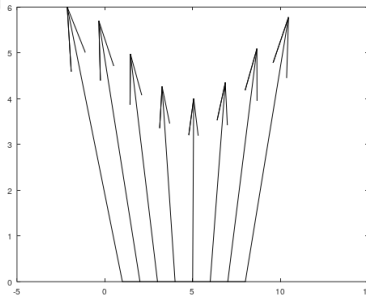
Ejemplo 1.17

```
x1=-pi:0.8:pi;
x2=5-x1.^2;
compass(x1,x2,'k')
```



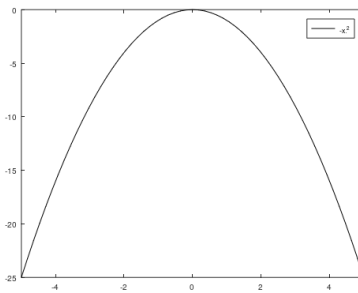
Ejemplo 1.18

```
x1=-pi:0.8:pi;
x2=5-cos(x1);
feather(x1,x2,'k')
```



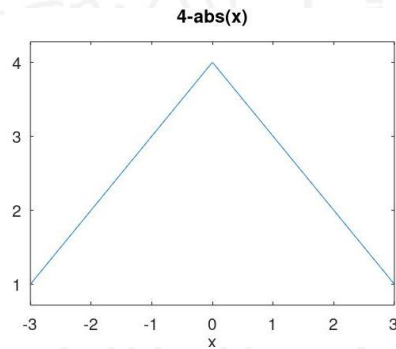
Ejemplo 1.19

```
fplot('-x.^2',[-5,5],'k')
```



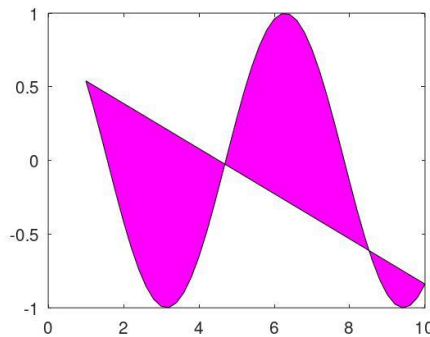
Ejemplo 1.20

```
ezplot('4-abs(x)',  
[-3,3])
```



Ejemplo 1.21

```
x=1:0.2:10;  
y=cos(x);  
fill(x,y,'m');
```



1.6 Gráficos en el sistema cartesiano tridimensional

Los dibujos en el sistema cartesiano tridimensional, de rectas y superficies, se pueden implementar en Octave como se muestra a continuación:

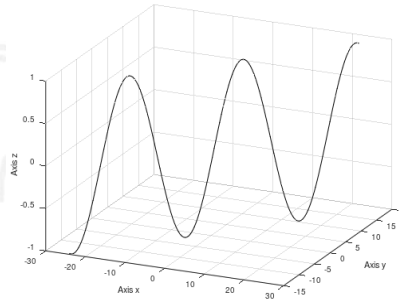
Comandos	Definición
<code>plot3(x₁, x₂, x₃)</code>	La función <code>plot3</code> dibuja los puntos (x_1, x_2, x_3) en el sistema cartesiano tridimensional.
<code>fill3(x₁, x₂, x₃, c)</code>	La función <code>fill3</code> dibuja la región poligonal cuyas coordenadas de los vértices son los elementos de los vectores columna x_1, x_2, x_3 . El parámetro <code>c</code> contiene el color a graficar.
<code>meshgrid(x₁, x₂)</code>	La función <code>meshgrid</code> construye matrices bidimensionales a partir de los vectores x_1 e x_2 , para poder construir la representación gráfica de una superficie explícita $z=f(x_1, x_2)$.

<code>mesh(x₁, x₂, x₃)</code>	La función mesh construye una representación gráfica de una superficie explícita $x_3 = f(x_1, x_2)$ con vectores x_1, x_2, x_3 .
<code>meshc(x₁, x₂, x₃)</code>	La función meshc construye una representación gráfica de una superficie explícita $x_3 = f(x_1, x_2)$ con los vectores x_1, x_2, x_3 ; proyectando las curvas de nivel en el plano x_1x_2 .
<code>meshz(x₁, x₂, x₃)</code>	La función meshz construye la representación gráfica de una superficie explícita $x_3 = f(x_1, x_2)$ con los arreglos x_1, x_2, x_3 ; cerrando la gráfica con las fronteras del dominio.
<code>surf(x₁, x₂, x₃)</code>	La función surf construye la representación gráfica de una superficie explícita $x_3 = f(x_1, x_2)$ con los vectores x_1, x_2, x_3 ; pintando cada una de las celdas.
<code>surfc(x₁, x₂, x₃)</code>	La función surfc construye la representación gráfica de una superficie explícita $x_3 = f(x_1, x_2)$ con los vectores x_1, x_2, x_3 ; proyectando las curvas de nivel en el plano xy.
<code>surfl(x₁, x₂, x₃)</code>	La función surfl construye la representación gráfica de una superficie explícita $x_3 = f(x_1, x_2)$ con los vectores x_1, x_2, x_3 ; considerando una iluminación en formato básico.
<code>waterfall(x₁, x₂, x₃)</code>	La función waterfall construye la representación gráfica de una superficie explícita $x_3 = f(x_1, x_2)$ en forma de cascada.
<code>contour(x₁, x₂, x₃)</code>	La función contour construye la representación gráfica de curvas de nivel de la superficie explícita $x_3 = f(x_1, x_2)$.

Los comandos mencionados anteriormente se utilizan para crear gráficos en tres dimensiones.

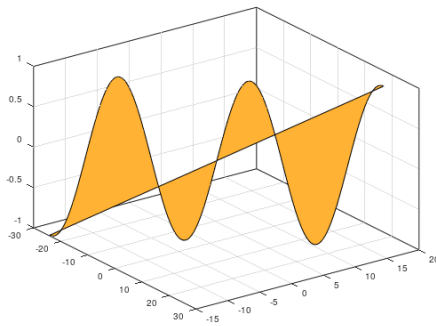
Ejemplo 1.22

```
t=-8:0.1:8;  
x1=3*t; x2=1+2*t;  
x3=sin(t);  
plot3(x1,x2,x3,'k');  
grid  
xlabel('Axis x')  
ylabel('Axis y')  
zlabel('Axis z')
```



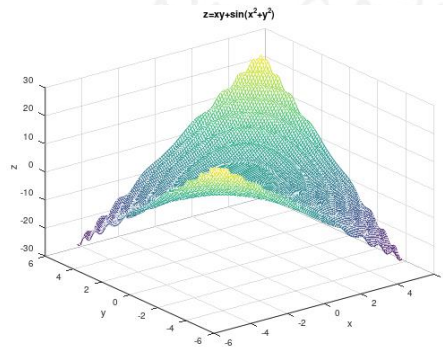
Ejemplo 1.23

```
t=-8:0.1:8;  
x=3*t;  
y=1+2*t;  
z=sin(t);  
k=[1 0.7 0.2];  
fill3(x,y,z,k);  
grid
```



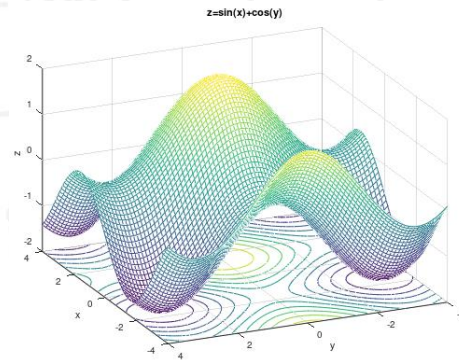
Ejemplo 1.24

```
[x1, x2]=meshgrid(-5:0.1:5);  
x3=x1.*x2+sin(x1.^2+x2.^2);  
mesh(x1,x2,x3); title('z=xy+sin(x^2+y^2)');  
xlabel('x');ylabel('y');zlabel('z');
```



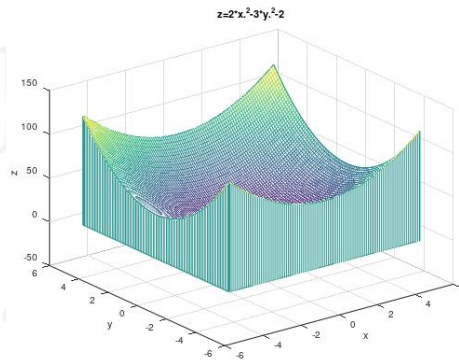
Ejemplo 1.25

```
[x1, x2]=meshgrid(-4:0.1:4); x3=sin(x1)+cos(x2); meshc(x1,x2,x3)  
title('z=sin(x)+cos(y)'); xlabel('x');ylabel('y');zlabel('z');
```



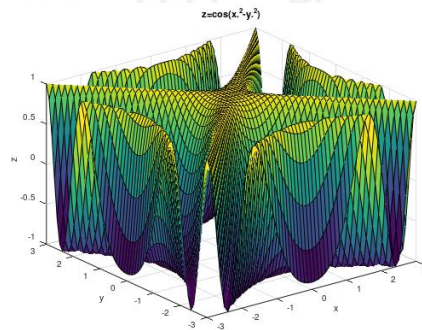
Ejemplo 1.26

```
[x1, x2]=meshgrid(-5:0.1:5); x3=2*x1.^2+3*x2.^2-2; meshz(x1,x2,x3); title('z=2*x.^2-3*y.^2-2'); xlabel('x');ylabel('y');zlabel('z');
```



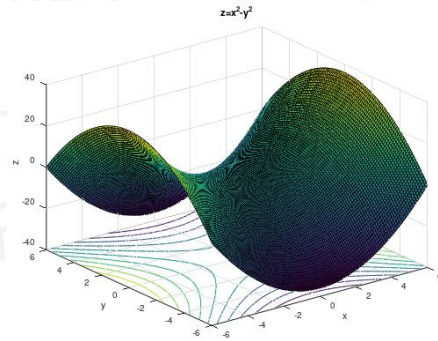
Ejemplo 1.27

```
[x1, x2]=meshgrid(-3:0.1:3); x3=cos(x1.^2-x2.^2); surf(x1,x2,x3); title('z=cos(x.^2-y.^2)'); xlabel('x');ylabel('y');zlabel('z');
```



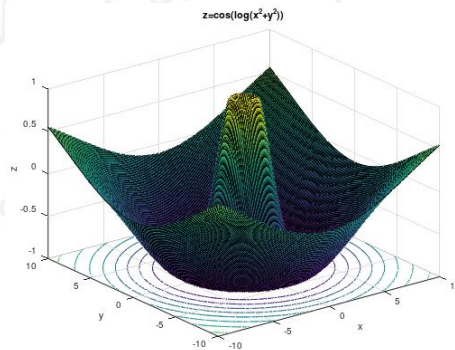
Ejemplo 1.28

```
[x1, x2]=meshgrid(-6:0.1:6); x3=x1.^2-x2.^2; surfc(x1,x2,x3); title('z=x^2-y^2'); xlabel('x');ylabel('y');zlabel('z');
```



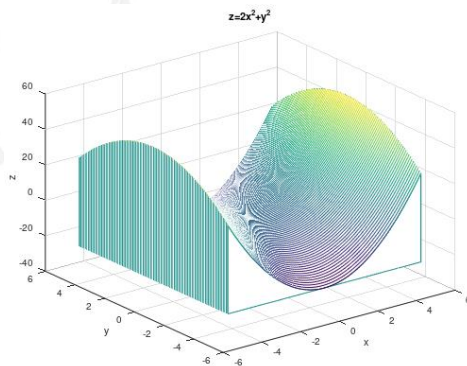
Ejemplo 1.29

```
[x1, x2]=meshgrid(-10:0.1:10); x3=cos(log(x1.^2+x2.^2));
surf(x1,x2,x3); title('z=cos(log(x^2+y^2))');
xlabel('x');ylabel('y');zlabel('z');
```



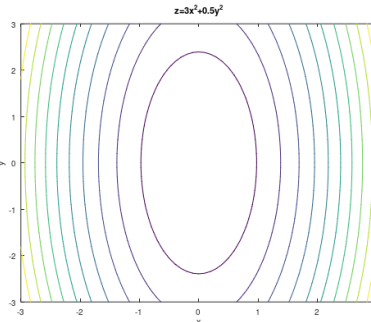
Ejemplo 1.30

```
[x1, x2]=meshgrid(-5:0.1:5); x3=2*x1.^2-x2.^2; waterfall(x1,x2,x3); title('z=2x^2+y^2');
xlabel('x');ylabel('y');zlabel('z');
```



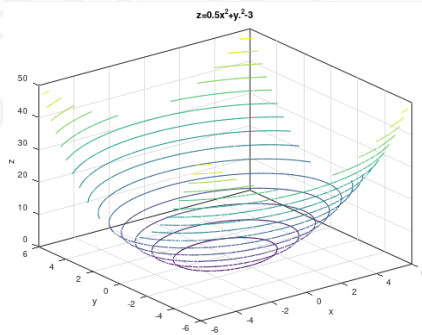
Ejemplo 1.31

```
[x1, x2]=meshgrid(-3:0.1:3); x3=3*x1.^2+0.5*x2.^2; contour(x1,x2,x3);  
title('z=3x^2+0.5y^2'); xlabel('x'); ylabel('y'); zlabel('z');
```



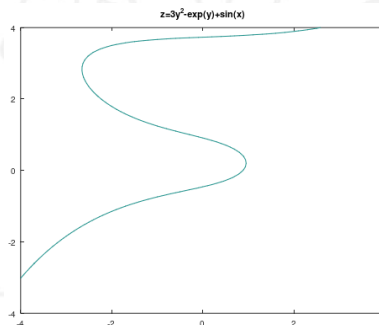
Ejemplo 1.32

```
[x1, x2]=meshgrid(-6:0.1:6); x3=0.5*x1.^2+x2.^2-3; contour3(x1,x2,x3,15);  
title('z=0.5x^2+y^2-3'); xlabel('x'); ylabel('y'); zlabel('z');
```



Ejemplo 1.33

```
[x1, x2]=meshgrid(-4:0.1:4);  
x3=3*x2.^2-exp(x2)+sinh(x1); contour(x1,x2,x3,[0,0]);title('z=3y^2exp(y)+sin(x)');
```



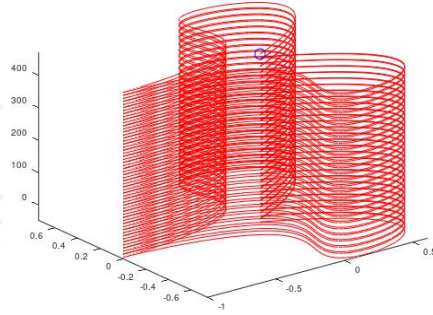
1.7 Gráficos especiales

La tabla muestra los comandos que Octave puede usar para representar superficies.

Comandos	Definición
cylinder(f)	La función cylinder construye la representación gráfica de una superficie de revolución.
sphere	Dibuja la esfera unitaria usando 20x20 caras.
ribbon(x, y, z, c)	La función ribbon construye la representación gráfica de una superficie explícita $z=f(x, y)$ en el espacio tridimensional. Además, c nos da el ancho de la superficie.
stem3(x, y, z)	La función stem3 construye la representación gráfica de puntos de una curva paramétrica $x=x(t)$, $y=y(t)$, $z=z(t)$, representándolo con líneas verticales tipo bastones.
pie3(x)	La función pie3 construye la representación gráfica de sectores en el espacio tridimensional.
quiver3(x, y, z, u, v, w, c)	La función quiver3 construye la representación gráfica de vectores de componentes (u, v, w) en los puntos (x, y, z), además c indica el tamaño de los vectores.
comet3(x, y, z)	La función comet construye la representación gráfica de una curva paramétrica, además genera un movimiento dicha curva.

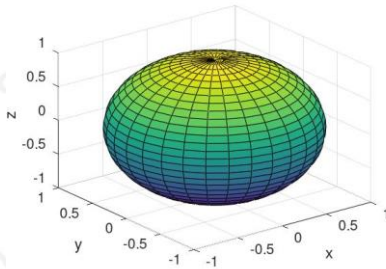
Ejemplo 1.34

```
t=-5*pi:pi/150:50*pi; x=(sin(3*t).^2).*cos(2*t); y=(sin(2*t).^2).*cos(t); z=3*t;
comet3(x,y,z)
```



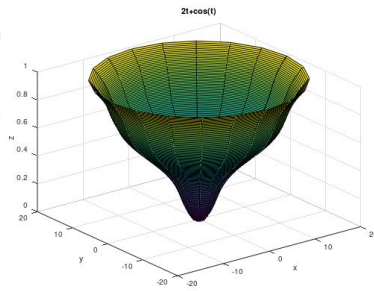
Ejemplo 1.35

```
sphere(20)
xlabel('x');
ylabel('y');
zlabel('z');
```



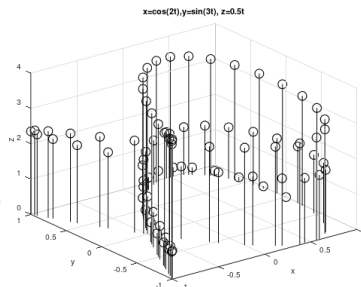
Ejemplo 1.36

```
t=0:0.1:10;
f=2*t+cos(t);
cylinder(f);
title('2t+cos(t)');
xlabel('x');
ylabel('y');
zlabel('z');
```



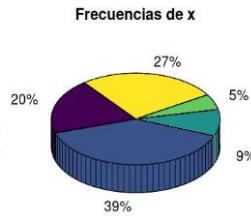
Ejemplo 1.37

```
t=0:0.1:8; x=cos(2*t); y=sin(3*t); z=0.5*t; stem3(x,y,z,'k'); title('x=cos(2t),y=sin(3t),z=0.5t'); xlabel('x'); ylabel('y'); zlabel('z');
```



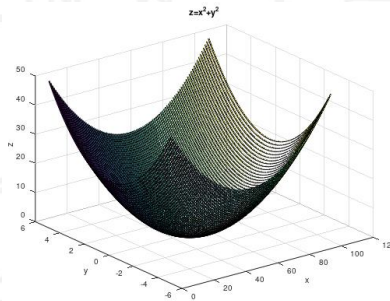
Ejemplo 1.38

```
x=[25 50 12 7 34];  
pie3(x);  
title('Frecuencias de x');  
xlabel('x');  
ylabel('y');
```



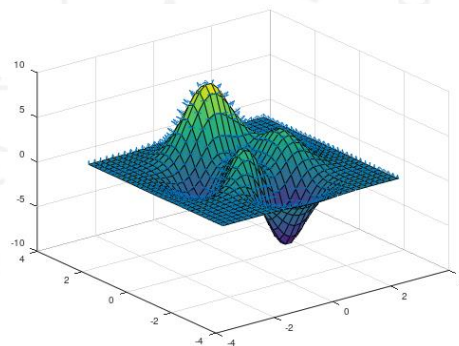
Ejemplo 1.39

```
[x,y]=meshgrid(-5:0.1:5); z=x.^2+y.^2; ribbon(y,z,0.5);  
title('z=x^2+y^2');xlabel('x');ylabel('y');zlabel('z');
```



Ejemplo 1.40

```
[x, y, z] = peaks(30); surf(x,y,z); hold on  
[u,v,w]=surfnorm(x,y,z/15); h = quiver3(x,y,z,u,v,w);  
set(h,'maxheadsize',0.50);
```

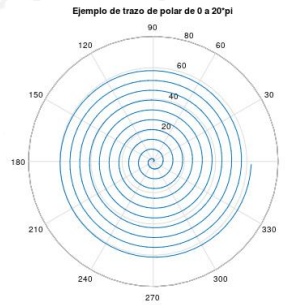


1.8 Gráficas en coordenadas polares

El trazado de datos en coordenadas polares se facilita con la función polar. Sin embargo, las coordenadas de visualización todavía son lineales y rectangulares.

Ejemplo 1.41

```
polar(0:0.1:20*pi, 0:0.1:20*pi);  
title('Ejemplo de trazo de polar de 0 a 20*pi')
```



1.9 Gráfica de un campo vectorial 2D.

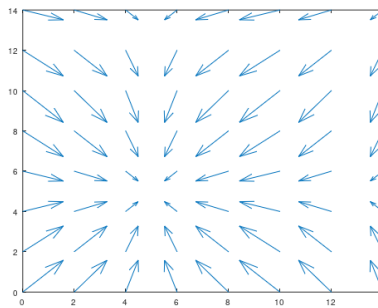
Los comandos `quiver(x1, y1, u1, v1)` y `quiver(u1, v1)`, dibujan las componentes $(u1, v1)$ de un campo de vectores en los puntos de la rejilla definidos por $(x1, y1)$. Si la rejilla es uniforme, entonces $x1$ e $y1$ se pueden expresar como vectores y el comando `meshgrid` se usa para crear la rejilla 2-D.

Si $x1$ e $y1$ no se definen, se supone que son $(1:m, 1:n)$ donde $[m, n] = \text{size}(u1)$.

Ejemplo 1.42

```
[x1, y1] = meshgrid(0:2:15);
```

```
quiver(x1, y1, cos(2*pi*x1/20), cos(2*pi*y1/20))
```



Taller N° 01 gráficos en 2D y 3D

1. Representa gráficamente la función $y = 1 + x$.
2. Representa la función $y = 2x^2 + x^2 + 1$ y resuelve el valor de y si $x = 10$.
3. Dibuja la función $y = 2 + \sin(2x)$ de color rojo y con círculo.
4. Dibuja la función $y = \cos x$, con $z = \text{sen } x$, una de color amarillo y otra de color negro.
5. Crea cuatro subplot de las siguientes funciones: seno, coseno, tangente y la cotangente de x .
6. Dibuja en una misma grafica las funciones

$$y = x^2, z = x^3 + 2.$$

7. Use octave para hallar el coeficiente de Rayleigh y grafique a partir de la siguiente formula:

$$rs = a \cdot \text{fray} \cdot \left(\frac{1}{\text{landa}} \right)^{-4}$$

$$a = 48; b = 0.7; \text{fray} = 0.41;$$

$$\text{landa} = 400 - 1200 \text{ nm}$$

8. Usar octave para hallar el índice de refracción a partir de la siguiente formula y compáralo con la longitud de onda propuesta:

$$n = \frac{A + B}{\lambda^2} + \frac{C}{\lambda^4}$$

$$A = 13696; B = 3916.8; C = 2558.8; \text{landa} = -1300$$

9. Representa en la misma gráfica:

$$x = [1, 100], y = x^2 + x^3 + 2x + 6, z = \sin x^2,$$

$$w = \frac{\sin x}{\cos x}$$

10. Represente la función:

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

11. Dada la función:

$$x = \cos(at) - \cos^3(bt)$$

$$y = \sin(ct) - \sin^3(dt)$$

construya una función en el sistema cartesiano bidimensional que reciba los parámetros a, b, c, d y haga una representación gráfica de la función dada. Evalúe la función en los siguientes valores dados: [1,80,1,80], [80,1,80,1], [80,1,1,80] y [1,100,1,50].

12. Las ecuaciones paramétricas de un hiperboloide de una hoja son:

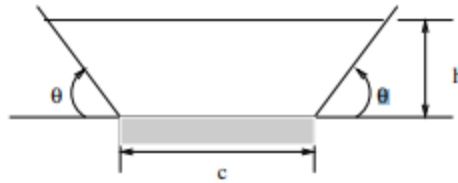
$$x = a \cdot \cosh(\theta) \cos(\phi)$$

$$y = b \cdot \cosh(\theta) \sin(\phi)$$

$$z = c \cdot \sinh(\theta)$$

Se pide representar la función con $\theta \in \mathbb{R}$, $0 < \phi \leq 2\pi$.

13. Se a diseñado un canal, como el que se muestra en la siguiente figura, se pretende que la velocidad de flujo sea máxima.



Las variables para considerar a la hora de diseñar el canal son su altura h , la anchura de la base c , y el Angulo lateral θ . Puede demostrarse que la velocidad de flujo es inversamente proporcional al perímetro, cuya expresión es:

$$p = c + \frac{2h}{\sin(\theta)}$$

Mientras que el área total de flujo es

$$A = ch + h^2 \cot(\theta)$$

Considerando que el canal debe tener 200 cms. De área transversal ($A=200$), la función $1/p$ únicamente dependerá de h y θ . Represente la superficie $\frac{1}{p} = f(h, \theta)$, mostrando el diagrama de contorno simultáneamente.

Solución del taller N° 01 gráficos en 2D y 3D

#Ejercicio 1

```
ezplot(@(x) 1+x)
```

Ejercicio 2

```
f = @(x) 2*x.^2 + x.^2 + 1
```

```
ezplot(f)
```

```
fprintf("f evaluado en x = 10\n")
```

```
feval(f,10)
```

Ejercicio 3

```
f = @(x) sin(x)
```

```
h = ezplot(f);
```

```
set(h,'color','g','marker','*','markersize',3)
```

Ejercicio 4

```
f = @(x) sin(x)
```

```
g = @(x) cos(x)
```

```
h1 = ezplot(f);
```

```
hold on
```

```
grid on
```

```
h2 = ezplot(g);
```

```
set(h1,'color','g')
```

```
set(h2,'color','r')
```

```
title("");
```

Ejercicio 5

```
subplot(2,2,1)
```

```
f = @(x) sin(x);
```

```
h1 = ezplot(f);
```

```
title("Funcion seno")
```

```
subplot(2,2,2)
```

```
f = @(x) cos(x);
```

```
h2 = ezplot(f);
```

```
title("Funcion coseno")
```

```
subplot(2,2,3)
```

```
f = @(x) tan(x);
```

```
h3 = ezplot(f);
```

```
title("Funcion tangente")
```

```
subplot(2,2,4)
```

```
f = @(x) cot(x); h4 = ezplot(f);
```

```
title("Funcion cotangente")
```

Ejercicio 6

```
f = @(x) x.^2; g = @(x) x.^3 + 2;
```

```
h1 = ezplot(f);
```

```

hold on
grid on
h2 = ezplot(g);
set (h1,'color','g')
set (h2,'color','r')
title("");

```

```

# Ejercicio 7

```

```

a = 48; b = 0.7;
fray = 0.41;
landa = 400:1:1200;
f = a.*fray.*(1./(landa./500)).^(-4);
plot(landa,f);
fprintf('%10.1f %10.1f\n',landa,f)

```

```

# Ejercicio 8

```

```

A = 13696; B = 3916.8; C = 2558.8;
lambda = 300:1:1600;
n = (A+B)./lambda.^2 + C./lambda.^4;
plot(lambda,n);
fprintf('%10.5f %10.5f\n',lambda,n)

```

```

# Ejercicio 9

```

```

x = [1:100]; y = x.^2 + x.^3 + 2.*x + 6;
z = 5000.*sin(x.^2); w = 5000.*sin(x)./cos(x);
hold on
plot(x, y, x, z, x, w)

```

```

# Ejercicio 10

```

```

f = @(x) (x.^2 + x + 1)./(x + 1)
h = ezplot(f);

```

```

# Ejercicio 11

```

```

function z = grafica(a, b, c, d)
t = linspace(0,10*pi,1000);
x = cos(a.*t) - (cos(b.*t)).^3;
y = sin(c.*t) - (sin(d.*t)).^3;
plot(t,x,'g',t,y,'r')
endfunction

```

```

# Ejercicio 12

```

```

theta = [0:0.1:2*pi];
phi = [0:0.1:2*pi];
a = 1; b = 1; c = 1;
x = a.*cosh(theta).*cos(phi);
y = b.*cosh(theta).*sin(phi);
z = c.*sinh(theta);
plot3(x,y,z)

```

Ejercicio 13

x = 2:1:30;

altura desde 2 para poder ver las curvas de nivel.

y = 0:0.1:pi*150/180;

no debe llegar a 180° porque "p" se hace infinito

generamos puntos con x,y que podamos utilizar en las # ecuaciones para "p"

[h,theta] = meshgrid(x,y);

c = (200 - h.^2.*cot(theta))./h;

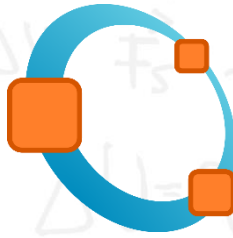
p = c + 2.*h./sin(theta); k = 1./p;

graficamos la malla y las curvas de nivel

meshc(h, theta, k)

Objetivo:

Calcular límites, derivadas e integrales con los comandos apropiados que posee Octave.

**CONTENIDO:**

- ✓ Límite de una función.
- ✓ Límite de una función seccionada.
- ✓ Límite al infinito.
- ✓ Derivada de una función.
- ✓ Integrales indefinidas.
- ✓ Integrales definidas.

Capítulo**2**

CAPITULO II. CALCULO DIFERENCIAL E INTEGRAL

2.1 Limite de una función de variable real.

Sea $f(x)$ una función que se aproxima a un número finito L tomando a x suficientemente cerca pero distinto del valor a , por la izquierda como por la derecha de a , entonces $\lim_{x \rightarrow a} f(x) = L$. (Mitacc et al. 2011).

La representación $x \rightarrow a^-$ se usará para señalar que x tiende a a por la izquierda y $x \rightarrow a^+$ para representar que x tiende a a por la derecha. De esta manera, los límites $\lim_{x \rightarrow a^-} f(x)$ y $\lim_{x \rightarrow a^+} f(x)$ tienen un valor común L .

Nota 1. En esta sección se calculan los límites de dos formas:

- 1) Usando el comando limit de Octave.
- 2) De forma numérica.

Ejemplo 2.1. Sea $f(x) = \frac{1-\cos(x)}{\sin(x)}$, $\forall x \in [-3,3]$. Calcular, si existe $\lim_{x \rightarrow 0} f(x)$ y trazar la gráfica de $f(x)$.

Solución en octave

- Usando el comando limit.

Paso1. Se carga el paquete de simbólico de la siguiente manera:

```
Ventana de comandos
>> pkg load symbolic
>> syms x
>> |
```

Paso2. Se ingresa la función de forma anónima.

```
Func=@(x) (1-cos(x))/sin(x);
```

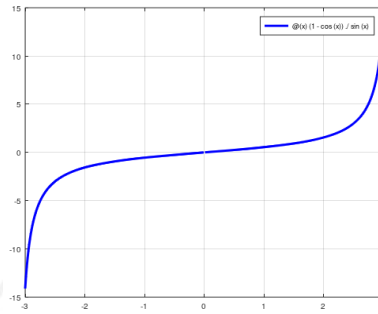
Paso3. Se calcula el límite de la función.

```
limit(Func,x,0);
```

```
>> pkg load symbolic
>> syms x
>> Func=@(x) (1-cos(x))/sin(x);
>> limit(Func,x,0)
ans = (sym) 0
>> Func=@(x) (1-cos(x))/x;
>> limit(Func,x,0)
ans = (sym) 0
```

La representación gráfica de la función se muestra en la figura 2.1.

FIGURA 2.1



- **Aproximación numérica**

El siguiente código se escribe en un fichero de Octave.

```
clear, clc % funciones para limpiar variables
disp('=====\n');
disp('Se calculan los limites laterales en x=a, de f(x)');
a=input('Ingrese el valor de a = ');
disp('=====\n');
delta=1e-10;
f=@(x)(1-cos(x))/sin(x);
disp('Se calcula el límite por la izquierda: ');
f_izq=f(a-delta)
disp('Se calcula el límite por la derecha: ');
f_der=f(a+delta)
disp('=====\n');
if f_izq==f_der
    Limite=f_izq
else
    disp('Limite no existe');
end
```

2.2 Límite de una función a trozos.

En esta sección se calcula el límite de una función con varias reglas de correspondencia.

Ejemplo 2.2. Consideremos la siguiente función:

$$f(x) = \begin{cases} x + 2 & \text{si } x \leq 5 \\ -x + 10 & \text{si } x > 5 \end{cases}$$

Calculemos $\lim_{x \rightarrow 5} f(x)$ y mostrar su gráfica.

Solución con el comando limit

El siguiente código se ha escrito en un fichero de octave.

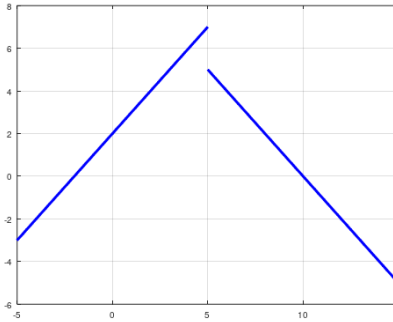
```

1 a=input('Ingrese el valor de a = ');
2 if x<=5
3     y=@(x)x+2; % esta es una función anónima.
4     limit(y,x,a)
5 else
6     y=@(x)-x+10;
7     limit(y,x,a)
8 endif

```

La gráfica de la función $f(x)$ se visualiza en figura 2.2.

FIGURA 2.2



Forma numérica

===== script =====

```

function y=fun1(x)
    if x<=5
        y=x+2; % esta es una función anónima.
    else
        y=-x+10;
    endif
endfunction

```

Nota. Guardar archivo con el nombre de fun1.

===== script =====

```
clear, clc
```

```

disp('=====\n');
disp('Vamos a calcular el límite de funciones a trozos');
a=input('Ingrese a = '); delta=1e-10;
disp('Se calcula el límite por la izquierda: '); f_izq=fun1(a-delta)
disp('Se calcula el límite por la derecha: ');
f_der=fun1(a+delta)
disp('=====\n');
if f_izq==f_der
    Limite=f_izq
else
    disp('Limite no existe');
end

```

Ejemplo 2.3. Consideremos la siguiente función:

$$f(x) = \begin{cases} 3 - x, & x < 2 \\ 1, & x = 2 \\ \frac{x}{2}, & x > 2 \end{cases}$$

Calcular el $\lim_{x \rightarrow 2} f(x)$ y mostrar su gráfica.

Solución con octave

```

===== script =====
function fun1(x)
if x<2
    y=3-x; % esta es una función anónima.
elseif x==2
    y=1;
else
    y=x/2;
end

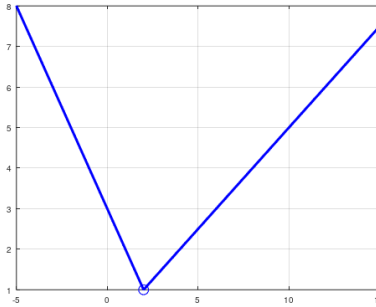
```

```
endif
```

```
endfunction
```

La gráfica de $f(x)$ se visualiza en la figura 2.3.

FIGURA 2.3



```
===== script =====
```

```
clear, clc
```

```
disp('=====\n');
```

```
disp('Vamos a calcular el límite de funciones a trozos');
```

```
a=input('Ingrese a = ');
```

```
disp('=====\n');
```

```
delta=1e-10;
```

```
disp('Se calcula el límite por la izquierda: '); f_izq=fun1(a-delta);
```

```
disp('Se calcula el límite por la derecha: '); f_der=fun1(a+delta);
```

```
disp('=====\n');
```

2.3 Los Límites al infinito

El comportamiento de las funciones es importante en muchas aplicaciones; cuando x se aproxima a números grandes, esto se puede considerar como un límite; y cuando x tiende a $-\infty$ o a $+\infty$, esto se considera como un límite. La función tiene un comportamiento asintótico si existe el límite; si no, la función crece o decrece indefinidamente, y en este caso no hay límite.

Ejemplo 2.4. Sea $f(x) = \frac{x^2-5x-7}{2x^2-11}$, calcular el $\lim_{x \rightarrow +\infty} f(x)$ si existe.

Solución con el comando limit

```
1 f=@(x) (x^2-5*x-7)/(2*x^2-11);  
2 limit(f,x,inf)
```

Taller N° 02 de Límites

Parte 1. Calcular el Límite de las siguientes funciones de una variable real.

1. $\lim_{a \rightarrow -4} \frac{4-a}{\sqrt{a^2-9}}$

2. $\lim_{a \rightarrow 1} \frac{a^6-1}{a^{10}-1}$

3. $\lim_{a \rightarrow 1} \frac{a^3-1}{\sqrt{a}-1}$

4. $\lim_{a \rightarrow 0} \frac{\sqrt{a^2+9}-3}{a^2}$

5. $\lim_{a \rightarrow 0} \frac{9^a-5^a}{a}$

6. $\lim_{a \rightarrow 0} \frac{\sqrt{1+a}-\sqrt{1-a}}{a}$

7. $\lim_{a \rightarrow 0} \frac{\cos(2a)-\cos(a)}{a^2}$

8. $\lim_{a \rightarrow 0} \frac{\tan(3a)}{\tan(5a)}$

9. $\lim_{a \rightarrow 0} a^2 \operatorname{sen}\left(\frac{1}{a}\right)$

Parte 2. Calcular el Límite de las siguientes Funciones Seccionadas

1. Consideremos la siguiente función:

$$f(x) = \begin{cases} x^2 & \text{si } x < 2 \\ 3 & \text{si } x = 2 \\ 2 & \text{si } x > 2 \end{cases}$$

Calcular el $\lim_{x \rightarrow 2} f(x)$

2. Consideremos la siguiente función:

$$f(x) = \begin{cases} 2-x^2, & x \leq 1 \\ 4-x, & x > 1 \end{cases}$$

Calcular el $\lim_{x \rightarrow 1} f(x)$

Parte 3. Límites al Infinito

1. $\lim_{a \rightarrow \infty} \frac{\text{sen}(2a)}{a}$
2. $\lim_{a \rightarrow -\infty} \frac{\cos(a)}{3a}$
3. $\lim_{a \rightarrow -\infty} \frac{2-a+\text{sen}(a)}{a+\cos(a)}$
4. $\lim_{a \rightarrow \infty} \frac{a+\text{sen}(a)}{2a+7-5\text{sen}(a)}$
5. $\lim_{a \rightarrow \infty} \frac{a+1}{a^2+3}$
6. $\lim_{a \rightarrow \infty} \frac{2a^3+7}{a^3-a^2+a+7}$
7. $\lim_{a \rightarrow \infty} \frac{2a^{5/3}-a^{1/3}+7}{a^{8/5}+3a+\sqrt{a}}$
8. $\lim_{a \rightarrow \infty} \frac{\cos(1/a)}{1+(1/a)}$
9. $\lim_{a \rightarrow \infty} \left(\frac{3}{a^2} - \cos\left(\frac{1}{a}\right)\right) \left(1 + \text{sen}\left(\frac{1}{a}\right)\right)$

Solución del taller N° 02 con octave

Antes de hacer correr estos ejercicios primero se recomienda cargar el paquete simbólico en la ventana de comandos de octave de la siguiente forma.

```
>> pkg load symbolic
```

```
>> syms a b x y
```

```
disp("\n===== Ejercicio 1 =====\n")
```

```
f=@(x) (4-x)/(sqrt(x^2-9)); limit(f,x,-4)
```

```
disp("\n===== Ejercicio 2 =====\n")
```

```
f=@(x) (x^6-1)/(x^10-1); limit(f,x,1)
```

```
disp("\n===== Ejercicio 3 =====\n")
```

```
f=@(x) (x^3-1)/(sqrt(x)-1); limit(f,x,1)
```

```
disp("\n===== Ejercicio 4 =====\n")
```

```
f=@(x) (sqrt(x^2+9)-3)/(x^2); limit(f,x,0)
```

```
disp("\n===== Ejercicio 5 =====\n")
```

```
f=@(x) (9^x-5^x)/(x); limit(f,x,0)
```

```
disp("\n===== Ejercicio 6 =====\n")
```

```

f=@(x) (sqrt(1+x)-sqrt(1-x))/(x); limit(f,x,0)
disp("\n===== Ejercicio 7 =====\n")
f=@(x) (cos(2*x)-cos(x))/(x^2); limit(f,x,0)
disp("\n===== Ejercicio 8 =====\n")
f=@(x) (tan(3*x))/(tan(5*x)); limit(f,x,0)
disp("\n===== Ejercicio 9 =====\n")
f=@(x) (x^2*sin(1/x)); limit(f,x,0)
===== script 1 =====
disp("\n==Limites de funciones seccionadas ==\n")
disp("\n===== Ejercicio 1 =====\n")
function b=funlim01(a)
    if a<2
        b=a^2;
    elseif a==2
        b=3;
    else b=2;
    endif
endfunction
===== script 2 =====
punto_critico = 2
delta=1e-10
limite_izq=double(funlim01(inicial-delta))
limite_der=double(funlim01(inicial+delta))
if abs(limite_izq - limite_der) < 1e-5
    disp(["El limite existe y es igual a " num2str(limite_izq)])
else
    disp("El limite no existe")
endif
===== script 1 =====
disp("\n===== Ejercicio 2 =====\n")
function b=funlim02(a)
    if a<=1
        b=2-a^2;
    else
        b=4-a;
    endif
endfunction
===== script 2 =====
punto_critico = 1
delta=1e-10
limite_izq=double(funlim02(inicial-delta))
limite_der=double(funlim02(inicial+delta))
if abs(limite_izq - limite_der) < 1e-5
    disp(["El limite existe y es igual a " num2str(limite_izq)])
else
    disp("El limite no existe")
endif

```

```

disp("\n===== Limites al infinito =====\n")
disp("\n===== Ejercicio 1 =====\n")
f=@(x) sin(2*x)/(x)
limit(f,x,inf)
disp("\n===== Ejercicio 2 =====\n")
f=@(x) cos(x)/(3*x)
limit(f,x,inf)
disp("\n===== Ejercicio 3 =====\n")
f=@(x) (2-x+sin(x))/(x+cos(x))
limit(f,x,inf)
disp("\n===== Ejercicio 4 =====\n")
f=@(x) (x+sin(x))/(2*x+7-5*sin(x))
limit(f,x,inf)
disp("\n===== Ejercicio 5 =====\n")
f=@(x) (x+1)/(x^2+3)
limit(f,x,inf)
disp("\n===== Ejercicio 6 =====\n")
f=@(x) (2*x^3+7)/(x^3-x^2+x+7)
limit(f,x,inf)
disp("\n===== Ejercicio 7 =====\n")
f=@(x) (2*x^(5/3)-x^(1/3)+7)/(x^(8/5)+3*x+sqrt(x))
l = limit(f,x,inf);
if l > 1e+50
    display("El limite no existe")
else
    l
endif
disp("\n===== Ejercicio 8 =====\n")
f=@(x) (cos(1/x))/(1+(1/x))
limit(f,x,inf)
disp("\n===== Ejercicio 9 =====\n")
f=@(x) (3/(x^2)-cos(1/x))*(1+sin(1/x))
limit(f,x,inf).

```

2.4 Derivada de una función real en un punto

Mitacc et al. (2011) define la derivada de una función de la siguiente manera:

Definición 1. Sea $f: \mathbb{R} \rightarrow \mathbb{R}$ una función definida en el punto $a \in D_f$. Se dice que f es derivable en a si el siguiente limite

$$f'(a) = \lim_{h \rightarrow 0} \frac{f(a+h) - f(a)}{h}$$

existe.

El comando diff.

La función diff es usado en octave para calcular de forma numérica la derivada de cualquier orden de funciones reales.

Ejemplo 2.5. Halle la derivada de la función $f(x) = \sqrt{x}$ en $x = 4$.

Para calcular la derivada de una función en Octave se usa el comando **diff** y para evaluar la derivada en un punto se usa el comando **subs**, como se muestra a continuación.

```
>> pkg load symbolic
>> syms x
>> derivada=diff(sqrt(x),x)
derivada = (sym)
      1
-----
 2*\ / x
>> subs(derivada,x,4)
ans = (sym) 1/4
```

2.5 Derivadas laterales

Sea $f: \mathbb{R} \rightarrow \mathbb{R}$ una función y $a \in D_f$. Mitacc et al. (2011) define la derivada lateral de la siguiente manera.

Definición 2. La derivada por la izquierda de f en a está definida por:

$$f'(a^-) = \lim_{h \rightarrow 0^-} \frac{f(a+h) - f(a)}{h}$$

si el límite existe. Mitacc et al. (2011).

Definición 3. La derivada por la derecha de f en a está definida por:

$$f'(a^+) = \lim_{h \rightarrow 0^+} \frac{f(a+h) - f(a)}{h}$$

si el límite existe. Mitacc et al. (2011).

Proposición 1. La función $f: \mathbb{R} \rightarrow \mathbb{R}$ es derivable en el punto $a \in D_f$ sí y solo sí existen y son iguales $f'(a^+)$ y $f'(a^-)$. Mitacc et al. (2011).

Ejemplo 2.6. Estudie la derivabilidad de la función:

$$f(x) = \begin{cases} 2 - x^2, & \text{si } x \leq 2 \\ x^2 - 4x + 2, & \text{si } x > 2 \end{cases}$$

en el punto $x = 2$.

El ejemplo 2.6 se resolvió usando un fichero de Octave de la siguiente forma.


```

1 % Primero cargar el siguiente paquete en la ver
2 % pkg load symbolic
3 % syms x
4
5 der_izq=diff(2-x^2,x);
6 Derivada_izquierda=double(subs(der_izq,x,2))
7 der_derecha=diff(x^2-4*x+2,x);
8 Derivada_derecha=double(subs(der_derecha,x,2))

```

Los resultados obtenidos son:

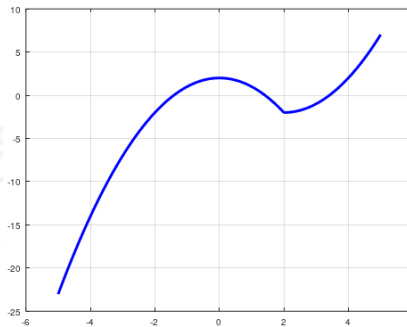
Derivada_izquierda = -4

Derivada_derecha = 0

Se confirma la derivada de $f(x)$ evaluada en el punto $x = 2$, no existe.

La gráfica de la función $f(x)$ se muestra en la figura 2.4.

FIGURA 2.4



Nota 2. Los comandos usados en el ejemplo 2.6 son:

- diff: Calcula la derivada de una función.
- Double: Convierte una cadena de caracteres a numérica.
- Subs: Para evaluar la función en un valor de x dado.

2.6 Derivadas de orden superior

Mitacc et al. (2011) define la derivada de orden superior de la siguiente manera. Sea $f(x)$ una función con derivada, la derivada de la función $f'(x)$ se denomina segunda derivada de f , y se denota de la siguiente manera:

$$f''(x) = \frac{d^2 f(x)}{dx^2}$$

Ejemplo 2.7. Si $y = \sqrt{x^2 + 5}$, hallar $f''(x)$.

```

Ventana de comandos
>> pkg load symbolic
>> syms x
>> y=@(x) sqrt(x^2+5);
>> segunda_der=diff(y,x,2);
>> simplify(segunda_der)
ans = (sym)
          5
-----
          3/2
 / 2    \
 \|x + 5/

```

Nota 3. Para calcular la derivada de orden superior en Octave, en la ventana de comando primero se escribe el comando diff, luego entre paréntesis se escribe la función, luego la variable independiente y finalmente el orden de la derivada, todo separado por comas.

Programación en octave

===== script =====

% Fómula de diferencias progresivas.

```

x=linspace(0, pi, 100);
a=0; b=pi; n=99;
h=(b-a)/n;
y=sin(x);
Progresivas=(y(2:end)-y(1:end-1))./h;
plot(x(1: end-1), Progresivas, 'r')
z1=[x(1: end-1)', Progresivas']

```

% Fómula de diferencias regresivas.

```

x=linspace(0, pi, 50);
a=0; b=pi; n=49;
h=(b-a)/n;
y=sin(x);
Regresivas=(y(end:-1:2) - y(end-1:-1:1))./h;
plot(x(end:-1:2), Regresivas, 'b')
z2=[x(end:-1:2)', Regresivas']
hold on

```

% Fómula de diferencias centradas.

```

x=linspace(0, pi, 50);
a=0; b=pi; n=49;
h=(b-a)/n;
y=sin(x);
centrada=(y(3:end) - y(1:end-2))./(2*h);
plot(x(2: end-1), centrada, 'k')
z3=[x(2: end-1)', centrada']

```

% Grafica de la derivada analitica

```

y=cos(x);
plot(x, y, 'm')

```

===== script =====

% Primera derivada de una función en forma general

function funderiv(a, b, N)

```
f=@(x)(sin(x)); h=(b-a)/(N-1); x=linspace(0, pi, 20);
dy=(feval(f, a+h*(1:N)) - feval(f, a+h*(-1:(N-2))))/(2*h);
z=[x', dy']
endfunction
```

Taller N° 03 derivadas

1. Sea f una secuencia de puntos x_i , representados en la siguiente tabla:

x_i	0	1	2	3
f_i	0	1	8	27

a) Aproxima la derivada de f en $x = 2$.

b) Aproxima la derivada de f en $x = 2.5$.

c) Aproxima la integral de f en $[0, 3]$.

2. Con los comandos flip y diff, calcule las diferencias regresivas.

3. Halle una formula centrada para la segunda derivada de la función $f(x)$ en el punto $x = a$ a partir de los valores $f(a - h)$, $f(a)$ y $f(a + h)$, donde $h \neq 0$. A partir del polinomio de interpolacion, justifica que la formula buscada es

$$f''(a) \approx \frac{f(a - h) - 2f(a) + f(a + h)}{h^2}$$

y prográmelo en octave.

4. $y = \tan(\sin(x^7))$; Halle $\frac{dy}{dx}$

5. $y = [\cos(11x - 7)]^5$; Halle $\frac{dy}{dx}$

6. $y = \cos(\sin(x^7))$; Halle $\frac{dy}{dx}$

7. $y = \cot g^{-\frac{1}{2}}(x^2)$; Halle $\frac{dy}{dx}$

8. $y = \cos[\ln(2x^4 + 5x)]$; Halle $\frac{dy}{dx}$

9. $y = e^{5x} \sin(7x)$; Halle $\frac{dy}{dx}$

10. $y = \cos[\cos(\cos(2x))]$. Halle $\frac{dy}{dx}$

11. $y = e^{\arcsin(2x)}$; Halle $\frac{dy}{dx}$

12. $y = (\sin x)e^{\tanh x}$; Halle $\frac{dy}{dx}$

13. $y = \frac{\sinh x}{\sqrt{x^3+3}}$, Halle $\frac{dy}{dx}$

14. Calcular la derivada en un punto de las funciones.

a) $f(x) = 5x^2 - 3$, $a = 2$ b) $f(x) = 8 - 2x^3$, $a = -1$

c) $f(x) = \sqrt{4 + 2x}$, $a = 0$ d) $f(x) = \frac{1}{11\sqrt{5+11x}}$, $a = 1$

e) $f(x) = |x - 1|^3$, $a = 1$

15. Si $f(x) = \sqrt{x^2 + 5}$, $g(x) = \frac{|x|}{1+x^2}$ y $h(x) = \frac{x^4+x^3+4}{\sqrt{2x+9}}$

Hallar: a) $f''(x)$ b) $g''(x)$ c) $h'''(x)$

Solución del taller N° 03 derivadas con octave

Antes de hacer correr estos ejercicios primero se recomienda cargar el paquete simbólico en la ventana de comandos de octave de la siguiente forma:

```
>> pkg load symbolic
```

```
>> syms x
```

```
disp("\n===== Ejercicio 1 =====\n")
```

```
x = [0, 1, 2, 3]
```

```
y = [0, 1, 8, 27]
```

```
disp("a. Derivada progresiva en x = 2\n")
```

```
% progresiva = (f(a+h)-f(a))/h
```

```
progresiva = (y(4) - y(3))/(x(4)-x(3))
```

```
disp("a. Derivada regresiva en x = 2\n")
```

```
% regresiva = (f(a) - f(a-h))/h
```

```
regresiva = (y(3) - y(2))/(x(3)-x(2))
```

```
disp("a. Derivada central en x = 2\n")
```

```
% central = (f(a+h) - f(a-h))/2h
```

```
central = (y(4) - y(2))/(x(4) - x(2))
```

```
disp("b. derivada en x = 2.5\n")
```

```
central2 = (y(4) - y(3))/(x(4) - x(3))
```

```
disp("c. Integral entre x [0, 3]\n")
```

```
disp("Aproximando con sumas de riemman con rectangulos por la derecha\n")
```

```
integral_rect = sum(y(2:end))
```

```
disp("Aproximando con sumas de riemman con trapecios\n")
```

```
integral_trap = sum((y(1:end-1) + y(2:end))/2)
```

```
disp("\n===== Ejercicio 2 =====\n")
```

```
x = [0, 1, 2, 3]
```

```
y = [0, 1, 8, 27]
```

```
regresivas = diff(flip(y))./diff(flip(x))
```

```
disp("\n===== Ejercicio 3 =====\n")
```

```
a=input('Ingresar el valor de a = ');  
b=input('Ingresar el valor de b = ');  
N=input('Ingresar el valor de N = ');  
x=linspace(a, b, N);  
h=(b-a)/(N-1);  
y=sin(x);  
centrada2=(y(1:end-2)-2*y(2:end-1)+y(3:end))/(h.^2);  
plot(x(2:end-1),centrada2, 'r')  
z1=[x(2:end-1)', centrada2'];  
hold on  
y1=-sin(x);  
plot(x, y1, 'b')
```

```
fprintf('===== Ejercicio 04 =====\n');  
f=@(x)tan(sin(x^7));  
simplify(diff(f,x))
```

```
fprintf('===== Ejercicio 05 =====\n');  
f=@(x)(cos(11*x-7))^5; simplify(diff(f,x))
```

```
fprintf('===== Ejercicio 06 =====\n');  
f=@(x)cos(sin(x^7)); simplify(diff(f,x))
```

```
fprintf('===== Ejercicio 07 =====\n');  
f=@(x)(cot(x^2))^(0.5); simplify(diff(f,x))
```

```
fprintf('===== Ejercicio 08 =====\n');  
f=@(x)cos(log(2*x^4+5*x)); simplify(diff(f,x))
```

```
fprintf('===== Ejercicio 09 =====\n');  
f=@(x)exp(5*x)*sin(7*x); simplify(diff(f,x))
```

```
fprintf('===== Ejercicio 10 =====\n');  
f=@(x)cos(cos(cos(2*x))); simplify(diff(f,x))  
fprintf('===== Ejercicio 11 =====\n');  
f=@(x)exp(asin(2*x)); simplify(diff(f,x))
```

```
fprintf('===== Ejercicio 12 =====\n');  
f=@(x)sin(x)*exp(tanh(x)); simplify(diff(f,x))
```

```
fprintf('===== Ejercicio 13 =====\n');  
f=@(x)sinh(x)/sqrt(x^3+3);  
simplify(diff(f,x))
```

```
fprintf('===== Ejercicio 14 apartado a =====\n');  
f=@(x)5*x^2-3;
```

```
Der=diff(f,x);
subs(Der,x,2)
```

```
f=printf('===== Ejercicio 14 apartado b =====\n');
f=@(x)8-2*x^3;
Der=diff(f,x);
subs(Der,x,-1)
```

```
f=printf('===== Ejercicio 14 apartado c =====\n');
f=@(x)sqrt(4+2*x);
Der=diff(f,x);
subs(Der,x,0)
```

```
f=printf('===== Ejercicio 14 apartado d =====\n');
f=@(x)1/(11*sqrt(5+11*x));
Der=diff(f,x);
subs(Der,x,1)
```

```
f=printf('===== Ejercicio 14 apartado e =====\n');
f=@(x)sqrt(x-1)^3/2;
Der=diff(f,x);
subs(Der,x,1)
```

```
f=printf('===== Ejercicio 15 apartado a =====\n');
f=@(x)sqrt(x^2+5);
Der=diff(f,x,2);
simplify(Der)
```

```
f=printf('===== Ejercicio 15 apartado b =====\n');
f=@(x)sqrt(x^2)/(1+x^2); Der=diff(f,x,2); simplify(Der)
```

```
f=printf('===== Ejercicio 15 apartado c =====\n');
f=@(x)(x^4+x^3+4)/sqrt(2*x+9); Der=diff(f,x,3);
simplify(Der)
```

2.7 Integral indefinida

Definición 4. Sea I un intervalo y $f: I \rightarrow \mathbb{R}$ una función. Una función $F: I \rightarrow \mathbb{R}$, tal que $F'(x) = f(x), \forall x \in I$, se denomina primitiva o antiderivada de f en I y se escribe $F(x) = \text{Ant}(f(x))$ en I . Mitacc et al. (1992).

Definición 5. Si $F(x)$ es una antiderivada de $f(x)$ en I , la integral indefinida de $f(x)$ es el conjunto de las antiderivadas de $f(x)$ en dicho intervalo y es denotado por:

$$\int f(x)dx = F(x) + c.$$

Mitacc et al. (1992)

Ejemplo 2.8. Calcular la siguiente integral indefinida

$$\int \frac{dx}{x^2 + 4}$$

```
Ventana de comandos
>> pkg load symbolic
>> syms x
>> y=1/(x^2+4);
>> int(y)
ans = (sym)

      /x\
atan|-|
      \|2/
-----
      2
```

Nota 3. Para calcular una integral indefinida se usa el comando int de Octave.

2.8 Integral definida

Mitacc et al. (1992) menciona en el teorema 1 la integral definida.

Teorema 1. Si f es una función continua en $I = [a, b]$ y F es una antiderivada de f en I esto es $F'(x) = f(x), \forall x \in I$, entonces se tiene

$$\int_a^b f(x)dx = [F(x)]_a^b = F(b) - F(a)$$

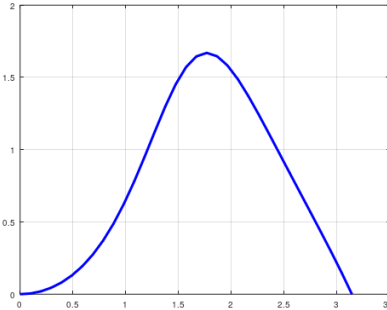
Ejemplo 2.9. Calcular la integral de $f(x) = \frac{x \sin(x)}{1 + \cos(x)^2}$

$$\int_0^{\pi} f(x)dx$$

```
Ventana de comandos
>> pkg load symbolic
>> syms x
>> y=@(x) (x*sin(x))/(1+cos(x)^2);
>> quad(y,0,pi)
ans = 2.4674
```

La gráfica de $f(x)$ se ve en la figura 2.5.

FIGURA 2.5



Nota 3. Para calcular una integral definida se usa el comando **quad** o **quadl** que permiten calcular integrales definidas de funciones continuas. La función **quad** es una función de bajo orden basado en la regla de Simpson, mientras que **quadl** es una función de alto orden.

Ejemplo 2.10.

Resolver la integral definida de $f(x) = \sqrt{1 - x^2}$, $\int_0^1 f(x) dx$.

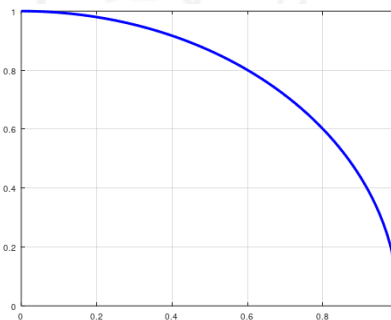
Solución con octave

Se define f , como una función anónima:

```
>> f = @(x) sqrt(1 - x^2);
>> quad(f, 0, 1);
>> 0.7854
```

La gráfica de la función $f(x)$ se ve en la figura 2.6.

FIGURA 2.6.



2.8 Integrales impropias.

La función **quad** de Octave calcula también integrales impropias.

Ejemplo 2.11. Calcular las siguientes integrales.

$$\int_0^4 \frac{1}{\sqrt{x+1}} dx, \int_{-\infty}^{\infty} e^{-\frac{x^2}{3}} dx, \int_4^{\infty} x^3 e^{-x} dx.$$


```
>> quad (@(x) 1/sqrt(x+1), 0,4)
>> quad (@(x) 4/sqrt(pi)*exp(-x^2/3), -inf, inf)
>> quad (@(x) x^3*exp(-*x), 4, inf)
```

2.8.1 Comandos similares a quad.

Octave ha implementado cinco algoritmos para calcular integrales definidas.

Comando	Definiciones
quadgk	quadgk, desarrolla funciones oscilatorias y/o intervalos infinitos.
quadl	quadl, es más efectiva que quad cuando la función no es muy regular.
quadcc	Es más efectiva de todos, pero con más tiempo computacional.
quadv	Versión vectorial de quad usando Simpson compuesto.

Taller N° 04 Integrales

1. Resolver con octave la siguiente integral $\int_1^2 \frac{e^{x^2}}{x^4} dx$ usando:

- Un refinamiento regular de 150 subintervalos donde el punto considerado en cada subintervalo sea su extremo superior derecho.
- Un refinamiento regular de 150 subintervalos donde el punto considerado en cada subintervalo sea el punto medio.

2. Considere la función $f(x) = \text{sen}(x^2)$.

- Desarrolle su grafica en el intervalo $[0, 1]$ ¿crece o decrece?
- Desarrollar la integral $\int_0^1 \text{sen}(x^2) dx$ con sumas de Riemann con $n = 70$ subintervalos y considerando como punto de cada subintervalo las siguientes opciones.
 - Extremo inferior del subintervalo
 - Extremo superior del subintervalo
 - Punto medio del subintervalo
- Si $f(x)$ es una función creciente, ¿Qué suma anterior es una cota superior y cual es una cota inferior?

3. Considere la función $f(x) = \text{sen}(x^2)$.

- Visualizar la gráfica en el intervalo $[\frac{\pi}{2}, \frac{2\pi}{3}]$ ¿crece o decrece?

b) Calcular un valor aproximado de $\int_{\pi/2}^{2\pi/3} \text{sen}(x^2) dx$ con sumas de Riemann, $n = 60$ subintervalos y tomando de cada subintervalo los siguientes apartados:

- i) Extremo inferior del subintervalo.
- ii) Extremo superior del subintervalo.
- iii) Punto medio del subintervalo.

4. Calcular las siguientes integrales indefinidas

- a) $\int (\sqrt{x} + 3) dx$
- b) $\int \sqrt{x}(x + 1) dx$
- c) $\int \frac{4dx}{\sqrt{6-x^2}}$
- d) $\int \frac{dx}{x(x^2-8)}$
- e) $\int \frac{7x^2+16}{x^4+4x^2} dx$
- f) $\int \frac{\cos 6x+6 \cos 4x+15 \cos 2x+10}{\cos 5x+5 \cos 3x+10 \cos x} dx$

5. Calcular las siguientes integrales definidas

- a) $\int_{-2}^2 x^3 dx$
- b) $\int_{-1}^2 (x + 1)^3 dx$
- c) $\int_0^{1/2} \frac{1}{\sqrt{1-x^2}} dx$
- d) $\int_1^2 \frac{x}{1+x^2} dx$
- e) $\int_0^{\pi/6} \text{sen}^2(3x) dx$

6. Calcular las siguientes integrales impropias con limites infinitos.

- a) $\int_0^{\infty} \text{sen } x dx$
- b) $\int_1^{\infty} \frac{4}{x^3} dx$
- c) $\int_0^{\infty} e^{-x} dx$
- d) $\int_{-\infty}^{+\infty} |x| e^{-x^2} dx$
- e) $\int_0^{\infty} \frac{dx}{\sqrt{e^x}}$

7. Calcular las siguientes integrales impropias con limites finitos.

- a) $\int_0^2 \frac{dx}{(x-2)^{3/5}}$
- b) $\int_{-2}^1 \frac{dx}{x^3}$
- c) $\int_0^3 \frac{dx}{\sqrt{9-x^2}}$
- d) $\int_{-2}^0 \frac{dx}{\sqrt[3]{x+1}}$

Solución del taller N° 04 Integrales con octave

Antes de hacer correr estos ejercicios primero se recomienda cargar el paquete simbólico en la ventana de comandos de octave de la siguiente forma.

```
>> pkg load symbolic
>> syms x
disp("===== Ejercicio 1 =====\n")
disp("a. considerando el extremo superior derecho\n")
a=input('ingrese a = ');
b=input('ingrese b = ');
```

```

N=input('ingrese N = ');
h=(b-a)/(N);
x=linspace(a,b,N+1);
f=@(x)(exp(x^2)/(x^4));
L=0;
for k=1:(N)
    L=L+abs(feval(f,a+k*h));
end
L1=h*L;
disp(L1)
disp("b. considerando el punto medio\n")
L=0;
for k=1:N
    L=abs(feval(f,a+(k-0.5)*h))+L;
end
L1=h*L;
disp(L1)

disp("===== Ejercicio 2 =====\n")
disp("a. considerando el extremo superior derecho\n")
a=0; b=1; N=50;
h=(b-a)/(N);
x=linspace(a,b,N+1);
y=sin(x.^2);
plot(x,y,'ro-','markersize',2,'linewidth',2)
der = diff(y)./diff(x);
flag=1;
for k=1:N
    if der(k) < 0
        flag = -1;
        break
    endif
endfor
if flag == 1
    disp("La funcion es creciente en el intervalo [0,1]\n")
else
    disp("La funcion es decreciente en el intervalo [0,1]\n")
endif
f = @(x) sin(x.^2);
disp("b.1 Integral con extremo inferior del subintervalo\n")
L = 0;
for k=1:N
    L=abs(feval(f,a+(k-1)*h))+L;
end
L1=h*L;
disp(L1)
disp("b.2 Integral con extremo superior del subintervalo\n")

```

```

L = 0;
for k=1:N
    L=abs(feval(f,a+k*h))+L;
end
L2=h*L;
disp(L2)
disp("b.2 Integral con punto medio del subintervalo\n")
L = 0;
for k=1:N
    L=abs(feval(f,a+(k-0.5)*h))+L;
end
L3=h*L;
disp(L3)
disp("c. Cota superior del valor exacto es \n")
L2
disp("c. Cota inferior del valor exacto es \n")
L1
resp = input('continuar (0/1): ');

disp("===== Ejercicio 3 =====\n")
disp("a. considerando el extremo superior derecho\n")
a=pi/2; b=2*pi/3; N=50;
h=(b-a)/(N);
x=linspace(a,b,N+1);
y=sin(x.^2);
plot(x,y,'go-','markersize',2,'linewidth',2)
der = diff(y)./diff(x);
flag=1;
for k=1:N
    if der(k) < 0
        flag = -1;
        break
    endif
endfor
if flag == 1
    disp("La funcion es creciente en el intervalo [0,1]\n")
else
    disp("La funcion es decreciente en el intervalo [0,1]\n")
endif
f = @(x) sin(x.^2)
disp("b.1 Integral con extremo inferior del subintervalo\n")
L = 0;
for k=1:N
    L=abs(feval(f,a+(k-1)*h))+L;
end
L1=h*L;
disp(L1)

```

```
disp("b.2 Integral con extremo superior del subintervalo\n")
```

```
L = 0;
```

```
for k=1:N
```

```
    L=abs(feval(f,a+k*h))+L;
```

```
end
```

```
L2=h*L;
```

```
disp(L2)
```

```
disp("b.2 Integral con punto medio del subintervalo\n")
```

```
L = 0;
```

```
for k=1:N
```

```
    L=abs(feval(f,a+(k-0.5)*h))+L;
```

```
end
```

```
L3=h*L;
```

```
disp(L3)
```

```
fprintf('=====\n Ejercicio 04 apartado a =====\n');
```

```
f=@(x)sqrt(x)+3;
```

```
Integral=simplify(int(f,x))
```

```
fprintf('=====\n Ejercicio 04 apartado b =====\n');
```

```
f=@(x)sqrt(x)*(x+1);
```

```
Integral=simplify(int(f,x))
```

```
fprintf('=====\n Ejercicio 04 apartado c =====\n');
```

```
f=@(x)4/sqrt(6-x^2);
```

```
Integral=simplify(int(f,x))
```

```
fprintf('=====\n Ejercicio 04 apartado d =====\n');
```

```
f=@(x)1/(x*(x^2-8));
```

```
Integral=simplify(int(f,x))
```

```
fprintf('=====\n Ejercicio 04 apartado e =====\n');
```

```
f=@(x)(7*x^2+16)/(x^4+4*x^2);
```

```
Integral=simplify(int(f,x))
```

```
fprintf('=====\n Ejercicio 04 apartado f =====\n');
```

```
f=@(x)(cos(6*x)+6*cos(4*x)+15*cos(2*x)+10)/(cos(5*x)+5*cos(3*x)+10*cos(x));
```

```
Integral=simplify(int(f,x))
```

```
fprintf('=====\n Ejercicio 5 apartado a =====\n');
```

```
f=@(x)x^3;
```

```
Integral=quad(f,-2,2)
```

```
fprintf('=====\n Ejercicio 5 apartado b =====\n');
```

```
f=@(x)(x+1)^3;
```

```
Integral=quad(f,-1,2)
```

```
fprintf('=====\n Ejercicio 5 apartado c =====\n');
```

```
f=@(x)1/sqrt(1-x^2);  
Integral=quad(f,0,1/2)
```

```
fprintf('===== Ejercicio 5 apartado d =====\n');  
f=@(x)x/(1+x^2);  
Integral=quad(f,1,2)
```

```
fprintf('===== Ejercicio 5 apartado e =====\n');  
f=@(x)(sin(3*x))^2;  
Integral=quad(f,0,pi/6)
```

```
fprintf('===== Ejercicio 6 apartado a =====\n');  
f=@(x)sin(x); Integral=quad(f,0,inf)
```

```
fprintf('===== Ejercicio 6 apartado b =====\n');  
f=@(x)4/(x^3); Integral=quad(f,1,inf)
```

```
fprintf('===== Ejercicio 6 apartado c =====\n');  
f=@(x)exp(-x); Integral=quad(f,0,inf)
```

```
fprintf('===== Ejercicio 6 apartado d =====\n');  
f=@(x)sqrt(x^2)*exp(-x^2); Integral=quad(f,-inf,inf)
```

```
fprintf('===== Ejercicio 6 apartado e =====\n');  
f=@(x)1/sqrt(exp(x)); Integral=quad(f,0,inf)
```

```
fprintf('===== Ejercicio 7 apartado a =====\n');  
f=@(x)(x-2)^(-3/5);  
Integral=int(f,x,0,2)
```

```
fprintf('===== Ejercicio 7 apartado b =====\n');  
f=@(x)x^(-3);  
Integral=int(f,x,-2,1)
```

```
fprintf('===== Ejercicio 7 apartado c =====\n');  
f=@(x)(9-x^2)^(-1/2);  
Integral=int(f,x,0,3)
```

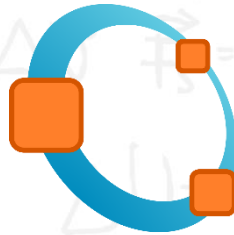
```
fprintf('===== Ejercicio 7 apartado d =====\n');  
f=@(x)(x+1)^(-1/3);  
Integral=int(f,x,-2,0)
```

Objetivo:

Aproximar numéricamente la solución de una EDO.

CONTENIDO:

- ✓ Ecuaciones diferenciales ordinarias.
- ✓ Ecuaciones diferenciales algebraicas.
- ✓ Método de Euler.
- ✓ Método de Taylor.
- ✓ Método de runge Kutta.
- ✓ Ecuaciones diferenciales de segundo orden.
- ✓ Ecuaciones diferenciales de tercer orden.
- ✓ Ecuaciones diferenciales de cuarto orden.

**Capítulo****3****CAPITULO III. ECUACIONES DIFERENCIALES LINEALES**

En este capítulo se presentan diferentes métodos numéricos que aproximan soluciones de ecuaciones diferenciales con condiciones iniciales.

3.1 Ecuaciones diferenciales ordinarias

Definición 4. Una ecuación diferencial es la que contiene derivadas o diferenciales de una función incognita. (Espinoza, 2002).

3.2. El comando lode de Octave

Octave tiene incorporado el comando lode para resolver problemas de valor inicial. Los parámetros de entrada que necesita la función lode son:

- Una función anónima.
- Un vector unidimensional con los valores iniciales u_0 .
- Un vector que contenga las iteraciones t_n para calcular la aproximación.

La función `lsode` tiene como salida una matriz por defecto, las columnas contienen los valores y_n , que son los resultados aproximados del vector solución u sobre cada uno de los nodos t_n especificados en el tercer argumento.

[x, i_inicial, msg] = lsode(Nombre, x_0, t)

La ecuación diferencial a resolver es:

$$\frac{dx}{dt} = f(x, t)$$

con

$$x(t_0) = x_0$$

El jacobiano de $f(x)$ tiene la forma:

$$\mathbf{jac} = \mathbf{j}(x, t)$$

donde

jac es la matriz de derivadas parciales.

$$J = \frac{\partial f_i}{\partial x_j} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_N} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_N}{\partial x_1} & \frac{\partial f_N}{\partial x_2} & \dots & \frac{\partial f_N}{\partial x_N} \end{bmatrix}$$

Ejemplo 3.1. Aproximar numéricamente el sistema Van Der Pol y visualizar su gráfica.

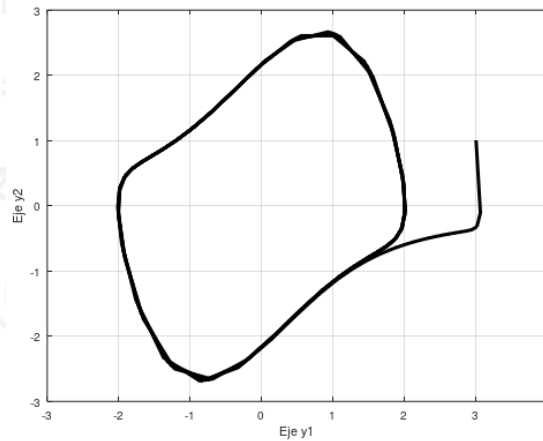
```
===== Inicio del script =====
funcion = @(y, t) [y(2); (1-y(1)^2)*y(2)-y(1)];
t = linspace(0, 30, 150); y = lsode(funcion, [3; 1], t);
fprintf('y1    y2\n'); disp([y(:, 1) y(:, 2)]);
plot(y(:, 1), y(:, 2), 'k', 'linewidth', 1.5)
xlabel('Eje y1'); ylabel('Eje y2'); grid;
===== Fin del script =====
```

Solución numérica del sistema de ecuaciones de Van Der Pol.

y_1	y_2
3.0000	1.0000
3.0606	-0.1092
3.0118	-0.3203
2.9415	-0.3677
2.8654	-0.3868
2.7861	-0.4014
2.7038	-0.4163
2.6183	-0.4326
2.5294	-0.4508
2.4367	-0.4714
2.3394	-0.4950
2.2371	-0.5222
2.1288	-0.5541
2.0136	-0.5919
1.8899	-0.6376

Visualización gráfica del sistema de ecuaciones de Van Der Pol. Ver figura 3.1.

FIGURA 3.1



Grafica del sistema de Van Der Pol con la función Isode.

Ejemplo 3.2. Aproximar numéricamente el siguiente sistema de ecuaciones y visualizar su gráfica.

$$\begin{cases} x' = 77.27(y - x * y + x - 0.005 * x^2) \\ y' = (z - x * y - y)/77.27 \\ z' = 0.161 * (x - z) \end{cases}$$

===== Inicio de script =====
function X=f(x, t)

```

X=zeros(3,1);
X(1)=77.27*(x(2)-x(1)*x(2)+x(1)-0.005*x(1)^2);
X(2)=(x(3)-x(1)*x(2)-x(2))/77.27;
X(3)=0.161*(x(1)-x(3));
endfunction
===== Fin de script =====

```

En otro script se obtiene la solución del sistema aproximado de forma numérica.

```

===== Inicio de script =====
x0 = [6;0.1; 6];
t = linspace (0, 300, 1500);
y = lsode('f', x0, t);
disp(y(1:15,:))
plot3(y(:, 1), y(:, 2), y(:, 3), 'k','linewidth',1.5)
grid
===== Fin de script =====

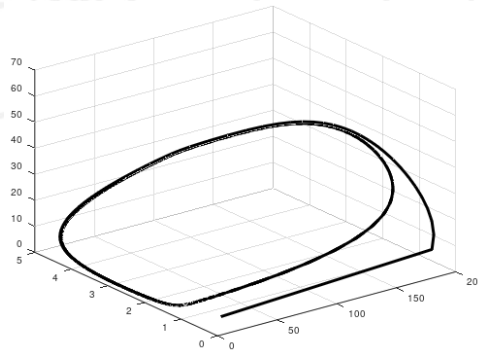
```

Solución numérica del sistema de ecuaciones tridimensional.

x	y	z
6.0000e+00	1.0000e-01	6.0000e+00
1.8263e+02	8.6185e-02	1.0172e+01
1.8400e+02	8.0498e-02	1.5671e+01
1.8263e+02	8.8147e-02	2.0992e+01
1.7957e+02	1.0406e-01	2.6072e+01
1.7546e+02	1.2505e-01	3.0876e+01
1.7071e+02	1.4920e-01	3.5386e+01
1.6555e+02	1.7536e-01	3.9596e+01
1.6013e+02	2.0285e-01	4.3504e+01
1.5451e+02	2.3131e-01	4.7112e+01
1.4875e+02	2.6052e-01	5.0426e+01
1.4286e+02	2.9041e-01	5.3451e+01
1.3685e+02	3.2099e-01	5.6190e+01
1.3069e+02	3.5231e-01	5.8650e+01
1.2438e+02	3.8451e-01	6.0834e+01

Los resultados obtenidos se grafican en octave. (ver figura 3.2)

FIGURA 3.2



Gráfica de un sistema tridimensional usando la función Isode.

Los comandos que aproximan soluciones actualmente en Octave son:

- Método de Runge-Kutta.

La función `ode45`, resuelve sistemas de ecuaciones diferenciales ordinarias no rígidas, esta función usa el método Dormand-Prince de paso variable y orden alto, también ofrece un rendimiento mejor con tolerancias más pequeñas.

La función `ode23`, resuelve un sistema de ecuaciones diferenciales ordinarias no rígidas, esta función aplica el método Bogacki-Shampine de tercer orden y acopla el tamaño de paso local para satisfacer una tolerancia especificada por el usuario.

La función `ode23s`, al igual que las funciones `ode45` y `ode23` resuelve sistemas de ecuaciones diferenciales ordinarias rígidas usando el método modificado de Rosenbrock de segundo orden.

- Métodos lineales de varios pasos.

La función `ode15s`, resuelve un conjunto de ecuaciones diferenciales ordinarias rígidas utilizando un paso variable.

La función `ode15i`, resuelve un conjunto de ecuaciones diferenciales ordinarias totalmente implícitas usando el mismo método de paso variable y orden variable que `ode15s`.

`[t, y] = ode45 (Función, Rango, Condición Inicial)`

Función. Es una cadena que almacena el nombre de la función que se define en la EDO: $y' = f(t, y)$. La función debe tener dos entradas, el primero es el tiempo t y el segundo es un vector columna de variable y .

Rango. Son los límites de tiempo durante el cual se evaluará la EDO. Típicamente es un vector de dos elementos que tiene un inicio y un final (`[t_init, t_final]`).

Condición Inicial. Es el valor inicial del PVI.

Ejemplo 3.3. Aproximar numéricamente el sistema de ecuaciones de Van Der Pol con octave.

```
===== inicio de script =====
```

```
F= @(t, y) [y(2); (1 - y(1)^2) * y(2) - y(1)];
```

```
[t, y] = ode45 (F, [0, 20], [2, 0]);
```

```
disp([y(:, 1),y(:, 2)]);
```

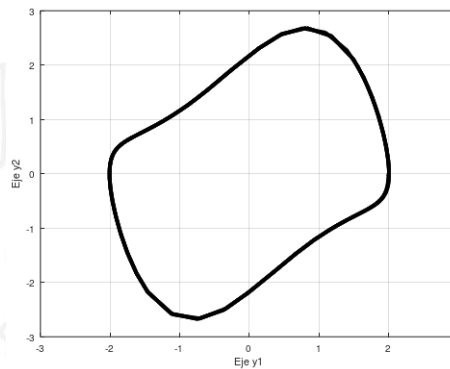
```
plot(y(:, 1), y(:, 2), 'k', 'linewidth', 2)
```

```
xlabel('Eje y1'); ylabel('Eje y2'); grid;
```

```
===== fin de script =====
```

Se visualiza la gráfica del sistema de Van Der Pol. (Ver figura 3.3).

FIGURA 3. 3



Gráfica del sistema de ecuaciones de Van Der Pol usando la función ode45.

A continuación, se muestra con un ejemplo cómo se trabaja con la función ode23.

```
[t, y] = ode23 (Función, Rango, Condición Inicial)
```

Ejemplo 3.4. Aproximar numéricamente el sistema de ecuaciones de Van Der Pol.

```
===== Inicio de script =====
```

```
Fun = @(t, y) [y(2); (1 - y(1)^2) * y(2) - y(1)];
```

```
[t, y] = ode23 (Fun, [0, 20], [2, 0]);
```

```
disp([y(:,1),y(:,2)]);
```

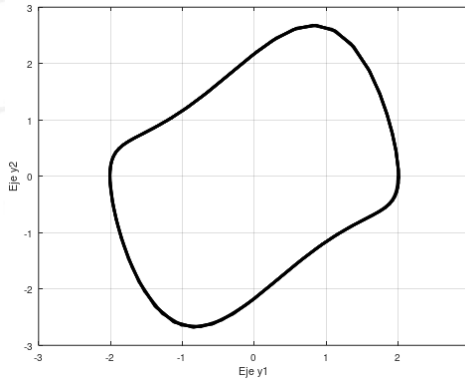
```
plot(y(:,1),y(:,2),'k','linewidth',1.5)
```

```
xlabel('Eje y1');ylabel('Eje y2'); grid;
```

```
===== Fin de script =====
```

Se visualiza la gráfica del sistema de ecuaciones de Van Der Pol en la figura 3.4.

FIGURA 3. 4



Grafica del sistema de ecuaciones de Van Der Pol usando la función ode23.

El ejemplo 3.5, muestra cómo se trabaja con la función ode23s.

Ejemplo 3.5. Aproximar numéricamente el sistema de ecuaciones rígida de Van Der Pol.

===== Inicio de script =====

```
F = @(t, y) [y(2); 1000*(1-y(1)^2)*y(2)-y(1)];
```

```
Opt = odeset ('Mass', [1 0; 0 1], 'MaxStep', 1e-1);
```

```
[vt, vy]=ode23s(F, [0 2000], [2 0], Opt);
```

```
disp([vy(:, 1), vy(:, 2)]);
```

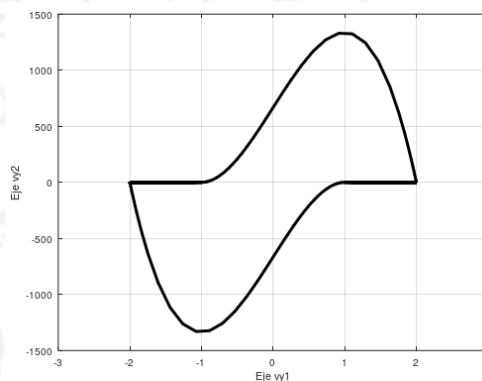
```
plot(vy(:, 1), vy(:, 2), 'k', 'linewidth', 1.5)
```

```
xlabel('Eje vy1'); ylabel('Eje vy2'); grid;
```

===== Fin de script =====

La gráfica del sistema de ecuaciones de Van Der Pol se visualiza en la figura 3.5.

FIGURA 3.5



Grafica del sistema de ecuaciones de Van Der Pol usando la función ode23s.

Ejemplo 3.6. Aproximar numéricamente el sistema bidimensional de Lotka-Volterra.

$$\begin{cases} u_1'(t) = au_1(t) - bu_2(t)u_1(t) \\ u_2'(t) = -cu_2(t) + du_1(t)u_2(t) \end{cases}$$

Con valor en los parámetros $a = b = c = d = 1$, se obtiene una aproximación numérica de soluciones de este sistema.

```
===== Inicio de script =====
t=linspace(0, 10, 100); x0=[0.5; 0.5]; a=b=c=d=1;
dp=@(u, t)[a*u(1)-b*u(2)*u(1); -c*u(2)+d*u(1)*u(2)];
sol=lsode(dp, x0, t); disp([sol(:, 1), sol(:, 2)])
plot(sol(:, 1), sol(:, 2), 'k', 'linewidth', 1.5)
xlabel('Eje u1'); ylabel('Eje u2'); grid;
```

A continuación, se muestran las 10 primeras iteraciones de las soluciones aproximadas.

0.5000	0.5000
0.5265	0.4760
0.5558	0.4544
0.5879	0.4352
0.6229	0.4182
0.6611	0.4033
0.7027	0.3905
0.7477	0.3798
0.7964	0.3712
0.8489	0.3646

La gráfica de la ecuación de Lotka-Volterra se visualiza en la figura 3.6.

FIGURA 3.6



Grafica del sistema de ecuaciones depredador-presa usando la función lsode.

3.3 Método de Euler

Euler utilizó el polinomio de Taylor de primer orden para deducir un método simple para aproximar la solución de una ecuación diferencial ordinaria.

$$w_{i+1} = w_i + hf(t_i, w_i) \quad i = 0, 1, 2, \dots, N-1,$$

Donde

h = tamaño de paso, con distribución uniforme en $[a, b]$.

$t_i = a + ih$, donde $i = 0, 1, 2, \dots, N$.

$w_i \approx y(t_i)$ representa el valor exacto de la solución en t_i .

Ejemplo 3.7. Resuelva la siguiente EDO por el método analítico y graficar la solución.

$$\frac{dy}{dx} = \frac{x^2 - 1}{y^2}, \quad y(0) = 2$$

Solución analítica

Paso 1. Separar variables

$$y^2 dy = (x^2 - 1) dx$$

Paso 2. Integrar ambos lados

$$\frac{y^3}{3} = \frac{x^3}{3} - x + c$$

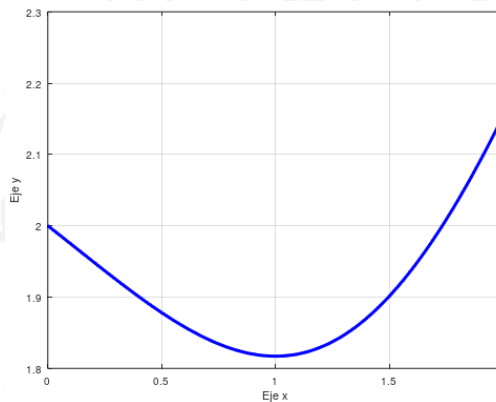
Paso 3. Despejar y

$$y = \sqrt[3]{x^3 - 3x + c}$$

Paso 4. Determinar el valor de c , usando las condiciones iniciales.

$$y = \sqrt[3]{x^3 - 3x + 8}$$

Gráfica de la solución del ejemplo 3.7.

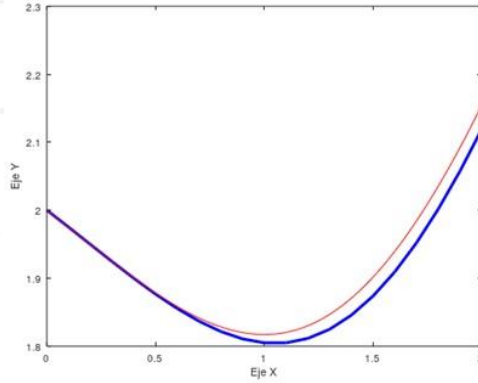


Resuelva el problema anterior, pero de forma numérica usando el método de Euler.

```
===== script =====  
function R=Euler(a,b,ya,M)  
% - Antes de correr el programa en octave deben  
% cargar el siguiente paquete.  
% pkg load symbolic  
% syms x y t  
% - [a, b]: dominio de la función.  
% - ya: Valor inicial.  
% - M: número de iteraciones.  
f=@(x,y)(x*x-1)/(y*y);  
h=(b-a)/M; T=zeros(1,M+1);  
Y=zeros(1,M+1); T=a:h:b; Y(1)=ya;  
for j=1:M  
Y(j+1)=Y(j)+h*feval(f,T(j),Y(j));  
end  
R=[T' Y'];  
plot(T,Y,'b','linewidth',1.5)  
hold on  
x=a:0.01:b; y=(x.^3-3*x+8).^(1/3);  
plot(x,y,'r')  
xlabel('Eje X'); ylabel('Eje Y');  
endfunction
```

La grafica de color azul representa la solución aproximada y la gráfica de color rojo la solución exacta. Ver figura 3.8.

Figura 3.8.



Aproximación numérica de la ecuación $\frac{dy}{dx} = \frac{x^2-1}{y^2}$,

$y(0) = 2$, usando el método de Euler.

3.4 Método de Taylor de orden n

$$W_0 = \alpha$$

$$W_{i+1} = W_i + hF^{(n)}(t_i, W_i)$$

para cada $i = 0, 1, \dots, N - 1$.

Donde

$$F^{(n)}(t_i, W_i) = f(t_i, W_i) + \frac{h}{2}f'(t_i, W_i) + \dots + \frac{h^{n-1}}{n!}f^{(n-1)}(t_i, W_i)$$

Ejemplo 3.8. Aproximar numéricamente la solución de la siguiente ecuación diferencial con valor inicial.

$$y' = y - t^2 + 1, \quad 0 \leq t \leq 2, \quad y(0) = 0.5$$

Derivar la siguiente función $f(t, y(t)) = y(t) - t^2 + 1$ respecto a la variable t .

Para orden dos es el siguiente:

$$w_{i+1} = w_i + h \left[\left(1 + \frac{h}{2}\right)(w_i - t_i^2 + 1) - ht_i \right],$$

$$w_0 = 0.5.$$

$$i = 0, 1, 2, \dots, N - 1.$$

Para orden cuatro:

$$w_0 = 0.5$$

$$w_{i+1} = w_i + h \left[\left(1 + \frac{h}{2} + \frac{h^2}{6} + \frac{h^3}{24}\right)(w_i - t_i^2) - \left(1 + \frac{h}{3} + \frac{h^2}{12}\right)ht_i + 1 + \frac{h}{2} + \frac{h^2}{6} + \frac{h^3}{24} \right]$$

$$i = 0, 1, \dots, N - 1.$$

Algoritmo de la serie de Taylor de orden dos.

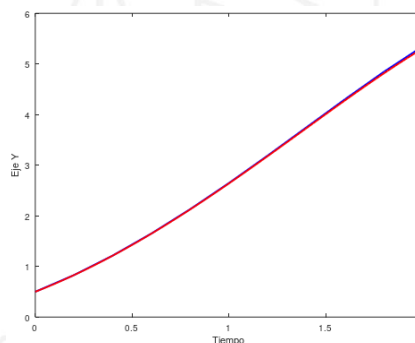
```

===== script =====
function R=SerieTaylor2(a, b, ya, M)
% - [a, b]: dominio de la función.
% - ya: valor inicial.
% - M: numero de iteraciones.
f=input('ingrese f entre comillas, f(t, y)= ');
f=inline(f,'t','y'); h=(b-a)/M;
T=zeros(1,M+1); Y=zeros(1,M+1); T=a:h:b; Y(1)=ya;
for j=1:M
Y(j+1)=Y(j)+h*((1+h/2)*(Y(j)-T(j)^2+1)-h*T(j));
end
R=[T' Y'];
plot(T,Y,'b','linewidth',1.5)
xlabel('Tiempo'); ylabel('Eje Y');
hold on
t=linspace(0,2,100);
y=(t+1).^2-0.5*exp(t);
plot(t,y,'r','linewidth',1)
endfunction

```

La gráfica del ejemplo 3.8 se visualiza en la figura 3.9.

FIGURA 3.9



La grafica rojo es solución real y la gráfica azul es la solución aproximada.

Burden & Faires. (2022), en su libro titulado “Análisis numérico” menciona diferentes metodos de aproximacion de soluciones de ecuaciones diferenciales ordinarias, a continuacion se mencionan algunos de ellos.

3.5 Método de Runge Kutta

3.5.1 Método del punto medio

$$W_0 = \alpha$$

$$W_{i+1} = W_i + hf(t_i + \frac{h}{2}, W_i + \frac{h}{2}f(t_i, W_i)), \text{ para cada } i = 0, 1, \dots, N - 1.$$

3.5.2 Método modificado de Euler

$$W_0 = \alpha$$

$$W_{i+1} = W_i + \frac{h}{2} [f(t_i, W_i) + f(t_{i+1}, W_i + hf(t_i, W_i))], \text{ para cada } i = 0, 1, \dots, N - 1.$$

3.5.3 Método de Heun

$$W_0 = \alpha$$

$$W_{i+1} = W_i + \frac{h}{4} [f(t_i, W_i) + 3f(t_i + \frac{2}{3}h, W_i + \frac{2}{3}hf(t_i, W_i))], \text{ para cada } i = 0, 1, \dots, N - 1.$$

3.5.4 Método de RK de orden cuatro.

$$W_0 = \alpha$$

$$K_1 = hf(t_i, W_i)$$

$$K_2 = hf(t_i + \frac{h}{2}, W_i + \frac{K_1}{2})$$

$$K_3 = hf(t_i + \frac{h}{2}, W_i + \frac{K_2}{2})$$

$$K_4 = hf(t_i + h, W_i + K_3)$$

$$W_{i+1} = W_i + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$

Para cada $i = 0, 1, \dots, N - 1$.

Ejemplo 3.9. Sea la ecuación diferencial $y' = 0.5 * (t - y)$ con $y(0) = 1$ en $[0, 3]$, usar Runge Kutta de orden cuatro para aproximar la ecuación de la ecuación diferencial.

===== script =====

function R=rk4(a, b, ya, M)

% - [a, b]: dominio de la función.

% - ya: valor inicial.

% - M: número de iteraciones.

f=input('ingrese f entre comillas, f(t, y)= ');

```

f=inline(f, 't', 'y');
h=(b-a)/M; T=zeros(1, M+1); Y=zeros(1, M+1);
T=a:h:b; Y(1)=ya;
for j=1:M
K1=h*feval(f, T(j), Y(j));
K2=h*feval(f, T(j)+h/2, Y(j)+K1/2);
K3=h*feval(f, T(j)+h/2, Y(j)+K2/2);
K4=h*feval(f, T(j)+h, Y(j)+K3);
Y(j+1)=Y(j)+(K1+2*K2+2*K3+K4)/6;
end
R=[T' Y']; plot(T,Y,'b','linewidth',1.5);
xlabel('Tiempo'); ylabel('Eje Y')
endfunction

```

En la ventana de comandos de octave escribimos:

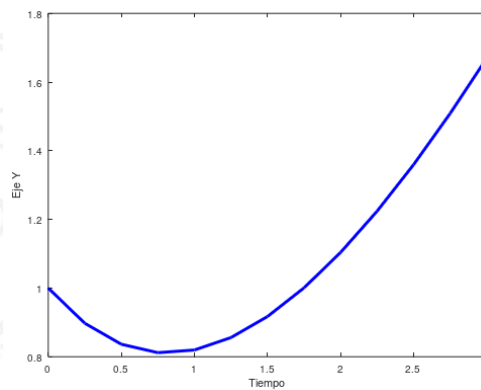
```

>> R=rk4(0,3,1,12)           presione enter
ingrese f entre comillas, f(t, y) = '0.5*(t-y)'

```

La gráfica del ejemplo 3.9 se muestran en la figura 3.10.

FIGURA 3. 10



Aproximación numérica de la ecuación $y' = 0.5 * (t - y)$, usando el método de RK de orden 4.

3.6 Ecuación diferencial ordinaria de segundo orden

Mediante cambios de variables en una ecuación diferencial de orden n , se logra obtener un sistema de ecuaciones de primer orden como se detalla:

$$y''(x) + a_1y'(x) + a_2y(x) = R(x) \quad (1)$$

$$y'(x_0) = y'_0$$

$$y(x_0) = y_0$$

Calcular: $y(x)$

Sea $v(x) = y'(x)$, reemplazando en (1) se obtiene la siguiente ecuación diferencial de primer orden con valor inicial.

$$y'(x) = v(x)$$

$$v'(x) = R(x) - a_1v(x) - a_2y(x)$$

$$y(x_0) = y_0$$

$$v(x_0) = y'_0$$

3.7 Ecuación diferencial ordinaria de tercer orden

$$y'''(x) + a_1y''(x) + a_2y'(x) + a_3y(x) = R(x) \quad (2)$$

$$y''(x_0) = y''_0$$

$$y'(x_0) = y'_0$$

$$y(x_0) = y_0$$

Calcular: $y(x)$.

Sea

$$u(x) = y'(x), u'(x) = y''(x), v(x) = u'(x),$$

$$v'(x) = u''(x).$$

Reemplazando en (2) se obtiene la siguiente ecuación diferencial de primer orden con valores iniciales.

$$y'(x) = u(x)$$

$$u'(x) = v(x)$$

$$v'(x) = R(x) - a_1v(x) - a_2u(x) - a_3y(x)$$

$$y(x_0) = y_0$$

$$u(x_0) = y'_0$$

$$v(x_0) = y''_0$$

3.8 Ecuación diferencial ordinaria de cuarto orden

$$y''''(x) + a_1y''''(x) + a_2y'''(x) + a_3y''(x) + a_4y'(x) + a_5y(x) = R(x) \dots \dots (3)$$

$$y''''(x_0) = y''''_0$$

$$y'''(x_0) = y'''_0$$

$$y''(x_0) = y''_0$$

$$y'(x_0) = y'_0$$

$$y(x_0) = y_0$$

Calcular: $y(x)$.

Sea

$$u(x) = y'(x), u'(x) = y''(x), v(x) = u'(x), v'(x) = u''(x), w(x) = v'(x), w'(x) = v''(x).$$

reemplazando y despejando de la ecuación (3) se obtienen el siguiente sistema de ecuaciones diferenciales de primer orden con valores iniciales.

$$\begin{aligned}y'(x) &= u(x) \\u'(x) &= v(x) \\v'(x) &= w(x) \\w'(x) &= R(x) - a_1 w(x) - a_2 v(x) - a_3 u(x) - a_4 y(x) \\y(x_0) &= y_0 \\u(x_0) &= y'_0 \\v(x_0) &= y''_0 \\w(x_0) &= y'''_0\end{aligned}$$

Ejemplo 3.10. Dada la ecuación diferencial

$$\text{PVI} \begin{cases} y'' + \text{sen}(x)y' + x^2y = 3/x \\ y_0 = 1, y'_0 = -2, x_0 = 2. \end{cases}$$

Hallar el valor de $y(4)$, para $n = 4$.

Solución

===== script =====

% - [a, b]: dominio de la función.

% - ya, yb: valor inicial.

% - M: numero de iteraciones.

function R=EulerS(a, b, ya, yb, M)

h=(b-a)/M; T=zeros(1, M+1);

Y1=zeros(1, M+1); Y2=zeros(1, M+1);

T=a:h:b; Y1(1)=ya; Y2(1)=yb;

f1=@(v)v; % función anónima uno

f2=@(t, y, v) 3/t-sin(t)*v-t^2*y; % función anónima dos.

for j=1:M

Y1(j+1)=Y1(j)+h*feval(f1, Y2(j));

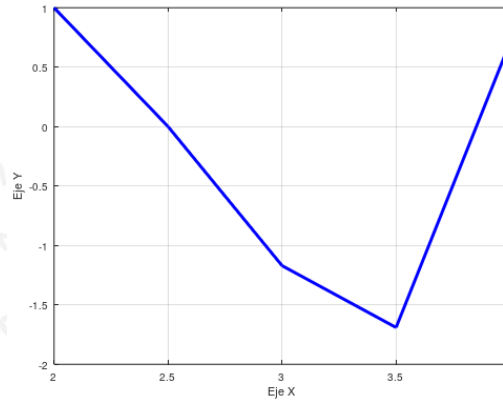
Y2(j+1)=Y2(j)+h*feval(f2, T(j), Y1(j),Y2(j));

end

```
R=[T' Y1'];
plot(T, Y1, 'b', 'linewidth', 1.5)
xlabel('Eje X'); ylabel('Eje Y');
endfunction
```

La gráfica del ejemplo 3.10 se visualiza en la figura 3.11

FIGURA 3.11



Gráfica de la solución numérica de

$y'' + \text{sen}(x)y' + x^2y = \frac{3}{x}$, desarrollado con el método de Euler.

Ejemplo 3.11. Dada la ecuación diferencial

$$\text{PVI} \begin{cases} y''' + 3xy'' + \cos(x)y' - xy = x^3 \\ y_0 = 1, y'_0 = -1, y''_0 = 3, x_0 = 2. \end{cases}$$

Hallar el valor de $y(4)$, para $n = 8$.

Solución

```
===== script =====
function R=EulerS3(a, b, ya, yb, yc, M)
h=(b-a)/M; T=zeros(1,M+1);
Y1=zeros(1,M+1); Y2=zeros(1,M+1); Y3=zeros(1,M+1);
T=a:h:b; % variable independiente x=t;
Y1(1)=ya; Y2(1)=yb; Y3(1)=yc;
f1=@(u) u; % Primera ec. del sist.
f2=@(v) v; % Segunda ec. del sist.
f3=@(t,y,u,v) t^3+t*y-cos(t)*u-3*t*v; % Tercera ec. del sist.
```

```
for j=1:M
```

```
Y1(j+1)=Y1(j)+h*feval(f1,Y2(j)); Y2(j+1)=Y2(j)+h*feval(f2,Y3(j));
```

```
Y3(j+1)=Y3(j)+h*feval(f3,T(j),Y1(j),Y2(j),Y3(j));
```

```
end
```

```
R=[T' Y1'];
```

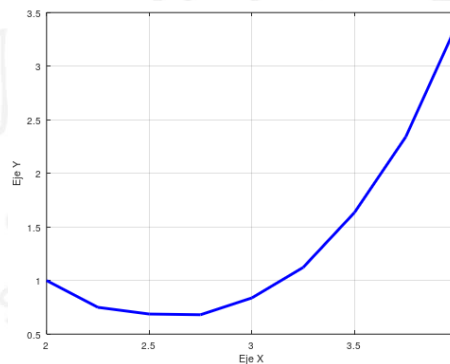
```
plot(T,Y1,'b','linewidth',1.5)
```

```
xlabel('Eje X'); ylabel('Eje Y');
```

```
endfunction
```

La gráfica del ejemplo 3.11 se muestra en la figura 3.12.

FIGURA 3.12



Gráfica de la solución numérica de $y''' + 3xy'' + \cos(x)y' - xy = x^3$, usando el método de Euler.

Taller N° 05 EDO con octave

1. Con el método de Euler aproxime las soluciones de los siguientes problemas de valor inicial.

a. $y' = te^{3t} - 2y$, $0 \leq t \leq 1$, $y(0) = 0$, con $h = 0.5$

b. $y' = 1 + (t - y)^2$, $2 \leq t \leq 3$, $y(2) = 1$, con $h = 0.5$

2. Con el método de Taylor de orden dos aproxime las soluciones de los siguientes problemas de valor inicial.

a. $y' = te^{3t} - 2y$, $0 \leq t \leq 1$, $y(0) = 0$, con $h = 0.5$

b. $y' = 1 + (t - y)^2$, $2 \leq t \leq 3$, $y(2) = 1$, con $h = 0.5$

3. Aplique Euler modificado para aproximar numéricamente las soluciones del ejercicio 1.

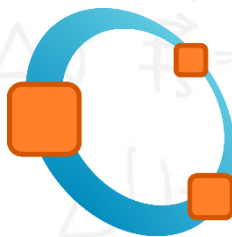
4. Desarrolle el ejercicio 1 usando el método de Heun.

5. Desarrolle el ejercicio 1 usando el método del punto medio.

6. Desarrolle el ejercicio 1 usando RK4.

Objetivo:

Aproximación numérica de soluciones en sistemas de ecuaciones diferenciales no lineales.

**CONTENIDO:**

- ✓ Preliminares.
- ✓ Sistema no lineal de Lorenz en \mathbb{R}^4 .
- ✓ Visualización gráfica de las trayectorias de fase.

Capítulo**4****CAPITULO IV. ECUACIONES DIFERENCIALES NO LINEALES.****4.1 Modelo de proyección de Velezmoro.**

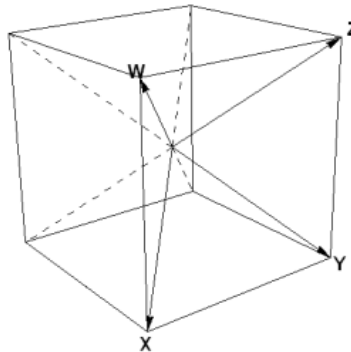
Velezmoro et al, (2019), propusieron un modelo matemático que permite graficar objetos 4D en 3D siguiendo el mismo método para visualizar objetos 3D en un espacio bidimensional. Los modelos matemáticos se construyen utilizando un proyector de hologramas 3D.

4.1.1 Modelo de proyección 4D

Considere un cubo de dos unidades de longitud cuyo centro es el origen de coordenadas del espacio tridimensional. Cuatro ejes de coordenadas, x, y, z y w están ubicados en el cubo. (ver figura 4.1).

FIGURA 4.1

Cubo con cuatro ejes coordenados.



Nota. En las diagonales del cubo se ubican los cuatro ejes coordenados x, y, z, w .

Considere los vectores unitarios:

$$\begin{aligned} e_1 &= \left(\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}} \right) & e_2 &= \left(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}} \right) \\ e_3 &= \left(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}} \right) & e_4 &= \left(\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}} \right) \end{aligned}$$

en la dirección de los ejes x, y, z y w (Ver figura 4.2).

Con los vectores e_1, e_2, e_3, e_4 se construye una proyección que por sí misma constituirá el modelo para graficar objetos 4D en un ambiente 3D.

Se define la proyección de la siguiente manera:

$$\psi: \mathbb{R}^4 \rightarrow \mathbb{R}^3$$

tal que

$$\psi(p) = p_1 e_1 + p_2 e_2 + p_3 e_3 + p_4 e_4, \forall p \in \mathbb{R}^4.$$

La proyección ψ como caso particular se escribe de la siguiente forma:

$$\psi(p) = \frac{1}{\sqrt{3}}(p_1 + p_2 + p_3 + p_4, -p_1 + p_2 + p_3 - p_4, -p_1 - p_2 + p_3 + p_4), \forall p \in \mathbb{R}^4.$$

4.2. Sistema de Lorenz 4D

Si Gang et al, (2011), propusieron el nuevo sistema de Lorenz en cuatro dimensiones (4D). Este sistema es no lineal en la segunda y tercera ecuación, tiene cuatro variables de estado x, y, z y w y cuatro parámetros positivos a, b, c y d .

$$\begin{cases} \frac{dx}{dt} = a(y - x) \\ \frac{dx}{dt} = cx - y - xz + w \\ \frac{dx}{dt} = xy - bz \\ \frac{dx}{dt} = -dx \end{cases} \quad (4.1)$$

También se puede elegir otros modelos de Lorenz 4D y visualizar de forma gráfica las superficies de fase y analizar su comportamiento cualitativo.

4.3. Algoritmo para graficar diagramas de fase de un sistema no lineal en 4D.

En esta subsección se muestran los pasos que permitieron reducir soluciones 4D en 3D para graficar diagramas de fase y realizar análisis cualitativo del sistema no lineal de Lorenz 4D.

Paso 1. Resolver el sistema utilizando un método numérico.

Paso 2. Utilice el modelo de proyección de Velezmoro e Ipanaque para reducir soluciones 4D en 3D y visualizar diagramas de fase del sistema.

4.4. Programa SystemLorenz4D.

El programa SystemLorenz4D es una implementación del algoritmo de la Sección 4.3 en el software científico Octave, este programa permite transformar soluciones 4D en 3D, graficar diagramas de fase y ciclos límites atractores para realizar comportamientos cualitativos del sistema.

Implementación del paso 1.

Este programa permitió encontrar soluciones aproximadas del sistema de Lorenz 4D.

```
a=10; b=8/3; c=28; d=0.5;
```

```
Lorenz4D=@(t,x)[a*(x(2)-x(1));c*x(1)-x(1)*x(3)-x(2)+x(4);x(1)*x(2)-b*x(3);-d*x(1)];
```

```
x0=[0.1 2 0.1 0.1];
```

```
t=linspace(0, 100, 10100); % tiempo.
```

```
[t, x] = ode45 (Lorenz4D, t, x0);
```

```
disp(x)
```

Implementación del paso 2.

Este programa permitió reducir las soluciones 4D en 3D.

```
a=10;b=8/3;c=28;d=0.5;
```

```

Lorenz4D=@(t,x)[a*(x(2)-x(1));c*x(1)-x(1)*x(3)-x(2)+x(4);x(1)*x(2)-b*x(3);-
d*x(1)];
x0=[0.1 2 0.1 0.1]; % condición inicial
t=linspace(0,100,10100); % tiempo
[t,x] = ode45 (Lorenz4D,t,x0);
p1=(1/sqrt(3)).
*[x(:,1)+x(:,2)+x(:,3)+x(:,4)];
p2=(1/sqrt(3)).
*[-x(:,1)+x(:,2)+x(:,3)-x(:,4)];
p3=(1/sqrt(3)).
*[-x(:,1)-x(:,2)+x(:,3)+x(:,4)];
plot3(p1,p2,p3,'k','linewidth',1.5)
xlabel('Axis x'); ylabel('Axis y'); zlabel('Axis z'); grid; view(-10,60)

```

4.5 Aplicación del algoritmo al nuevo sistema de Lorenz 4D.

La justificación de los comportamientos cualitativos del sistema de Lorenz 4D se dio tomando diferentes valores de los parámetros de control.

Por ejemplo, Si Gang et al, (2011), al tomar los valores $a=10$; $b=8/3$; $c=28$; $d=0.5$ obtuvo un sistema hipercaótico. (Ver ecuación 4.2).

$$\begin{cases} x' = 10(y - x) \\ y' = 28x - y - xz + w \\ z' = xy - (8/3)z \\ w' = -0.5x \end{cases} \quad (4.2)$$

Resolvemos el nuevo sistema de Lorenz 4D utilizando el método numérico de Runge Kutta (biblioteca ode45 en Octave). Se muestra la solución aproximada para las primeras 20 iteraciones. (Ver tabla 1).

TABLA 1

SOLUCIÓN NUMÉRICA DEL SISTEMA (4.2).

Nº	x	y	z	w
01	0	2.0000	0	0
02	0.1886	2.0066	0.0019	-0.0005
03	0.3623	2.0631	0.0073	-0.0018
04	0.5271	2.1657	0.0164	-0.0040
05	0.6882	2.3119	0.0293	-0.0071
06	0.8500	2.5009	0.0466	-0.0109
07	1.0164	2.7330	0.0692	-0.0155
08	1.1911	3.0093	0.0984	-0.0209
09	1.3777	3.3320	0.1356	-0.0273
10	1.5794	3.7041	0.1829	-0.0346
11	1.7996	4.1294	0.2427	-0.0430
12	2.0418	4.6125	0.3184	-0.0525
13	2.3099	5.1582	0.4141	-0.0632
14	2.6074	5.7727	0.5347	-0.0754
15	2.9383	6.4627	0.6862	-0.0891
16	3.3068	7.2350	0.8770	-0.1045
17	3.7176	8.0963	1.1171	-0.1219
18	4.1754	9.0536	1.4186	-0.1414
19	4.6847	10.1118	1.7972	-0.1634
20	5.2515	11.2799	2.2702	-0.1879

Con la aplicación del modelo de Velezmoro e Ipanaque, la solución del sistema 4D se reduce a una solución 3D. Se muestran a continuación los resultados. (Ver tabla 2).

TABLA 2.

REDUCCIÓN DE LA SOLUCIÓN 4D EN 3D.

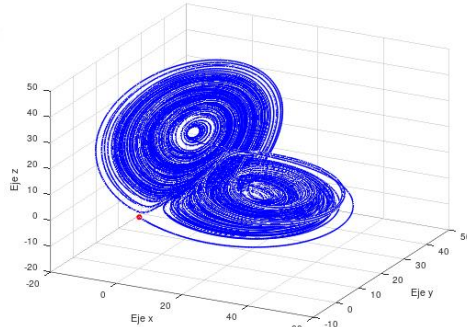
Nº	x	y	z
01	1.1547	1.1547	-1.1547
02	1.2682	1.0510	-1.2666
03	1.4035	0.9872	-1.3971
04	1.5618	0.9578	-1.5476
05	1.7450	0.9584	-1.7193
06	1.9553	0.9863	-1.9141
07	2.1958	1.0400	-2.1337
08	2.4699	1.1186	-2.3804
09	2.7817	1.2224	-2.6566
10	3.1360	1.3522	-2.9648
11	3.5385	1.5101	-3.3078
12	3.9954	1.6983	-3.6883
13	4.5143	1.9201	-4.1091
14	5.1034	2.1797	-4.5731
15	5.7724	2.4825	-5.0829
16	6.5323	2.8346	-5.6403
17	7.3953	3.2434	-6.2462
18	8.3751	3.7171	-6.9003
19	9.4861	4.2653	-7.5995
20	10.7466	4.8997	-8.3422

4.6 Análisis de estabilidad en el origen

Caso 1. Si $a > 0$, $b = 8/3$, $c = 28$ y $d = 0.5$, el primer valor propio es real negativo, el segundo es real positivo, el tercero es real positivo y el cuarto es real negativo. El origen es un punto crítico hiperbólico. (Ver figura 4.2).

FIGURA 4.2

Punto crítico hiperbólico

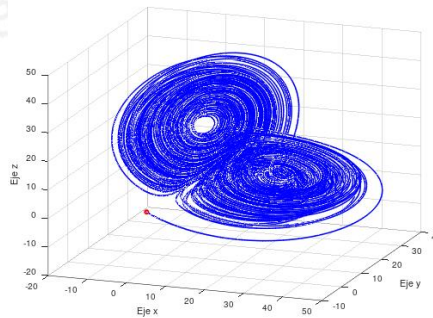


Nota. Las trayectorias salen del origen y vuelven a entrar.

Caso 2. Si $a=10$, $b>0$, $c=28$ y $d=0.5$, el primer valor propio es real negativo, el segundo es real positivo, el tercero es real positivo y el cuarto es real negativo. El origen es un punto crítico hiperbólico. (Ver figura 4.3).

FIGURA 4.3

Punto crítico hiperbólico

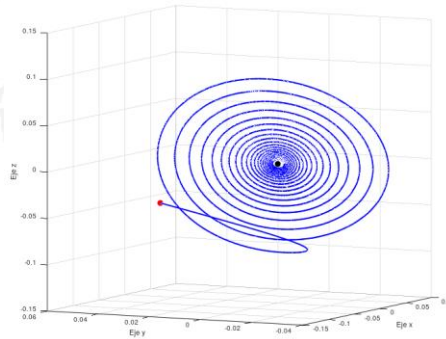


Nota. Las trayectorias salen del origen y vuelven a entrar.

Caso 3. Si $a=10$, $b=8/3$, $0<c<0.95455$ y $d=0.5$, el primer valor propio es real negativo, el segundo es complejo con parte real negativa, el tercero es complejo con parte real negativa y el cuarto es real negativo. El origen es un espiral estable. (Ver figura 4.4).

FIGURA 4.4

Estabilidad en el origen.

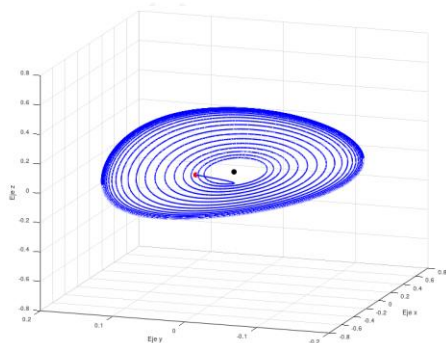


Nota. Cuando $c=0.9$, las trayectorias convergen al punto crítico en forma de espiral.

Caso 4. Si $a=10$, $b=8/3$, $0.95445 < c < 2.6$ y $d=0.5$, el primer valor propio es real negativo, el segundo es complejo con parte real positiva, el tercero es complejo con parte real positiva y el cuarto es real negativo. El origen es un espiral inestable. (Ver figura 4.5).

FIGURA 4.5

Inestabilidad en el origen.

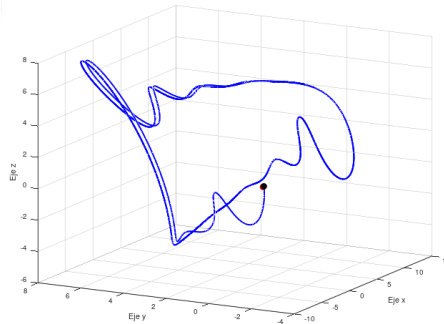


Nota. Cuando $a = 10$, $b = 8/3$, $d = 0.5$ y $c = 1$, las trayectorias divergen del punto crítico y forma un ciclo limite atractor.

Caso 5. Si $a=10$, $b=8/3$, $c > 2.6$ y $d=0.5$, el primer valor propio es real negativo, el segundo es real positivo, el tercero es real positivo y el cuarto es real negativo. El origen es un punto crítico hiperbólico. (Ver figura 4.6).

FIGURA 4.6

Inestabilidad en el origen.

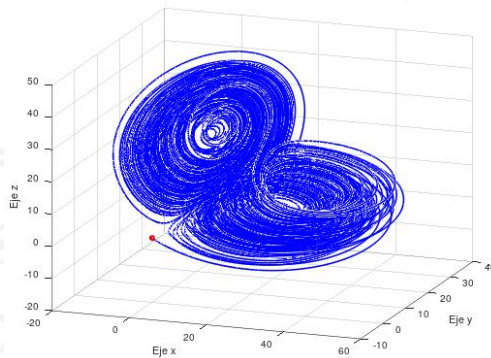


Nota. Cuando $a = 10, b = 8/3, d = 0.5$ y $c = 5$, las trayectorias divergen del punto crítico y forma un ciclo limite atractor

Caso 6. Si $a=10, b=8/3, c=28$ y $d>0$, el primer valor propio es real negativo, el segundo es real positivo, el tercero es real positivo y el cuarto es real negativo. El origen es un punto hiperbólico. (Ver figura 4.7).

FIGURA 4.7

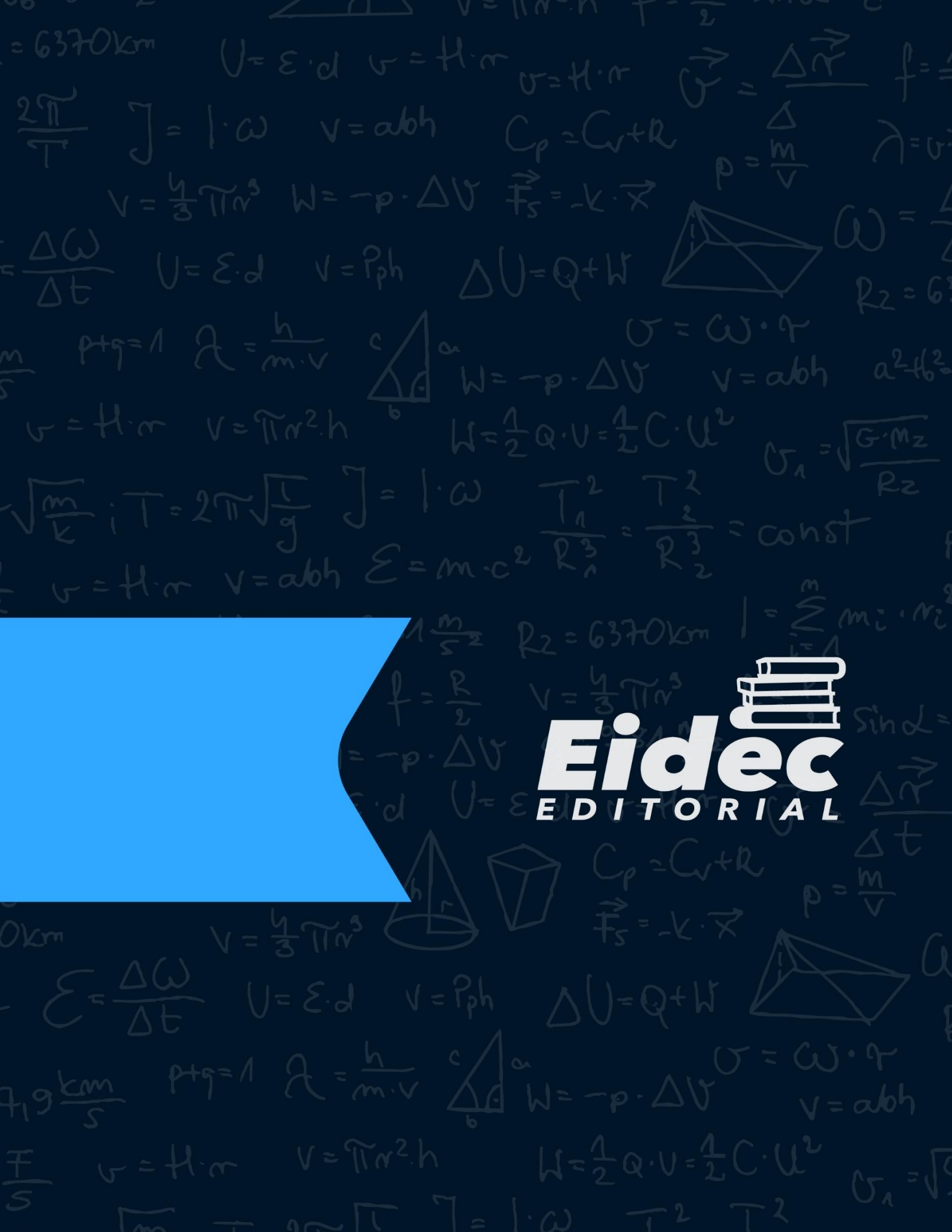
Inestabilidad en el origen.



Nota. Cuando $a = 10, b = 8/3, d = 1$ y $c = 28$, las trayectorias salen y vuelven al origen.

REFERENCIAS BIBLIOGRAFICAS

- Burden, R., & Faires, D. (2022). *Análisis Numérico* (Thomson Learning ed., Vol. 07).
- Escobar Gomez, E. (2021). *Visualizacion Grafica del atractor extraño de Lorens y Rossler en 3D y 4D*. (Vol. 01). Piura, Perú: Sieteventos.
- Espinoza ramos, E. (2000). *Análisis Matemático I* (Vol. 03). Lima, Peru: J. J.
- Espinoza Ramos, e. (2002). *Análisis Matemático IV* (Vol. 01). Lima, Peru: J. J.
- Mitacc, M., & Toro, I. (1992). *Temas de cálculo II* (Vol. 02). Lima, Peru: Impoffot.
- Mitacc, m., & Toro, L. (2011). *Temas de cálculo I* (Vol. 01). Lima, Peru: San Marcos.
- Si Gang, Q., Cao, H., & Zhang Yan, B. (2011). A new four-dimensional hyperchaotic Lorenz system and its adaptive control. *Sociedad Física China e IOP Publishing Ltd*, (Vol 20), págs. 1-9. doi:10.1088/1674-1056/20/1/010509.
- Velezmoro, R., Ipanaque, R., & Mechato, J. (2019). A mathematica package for visualizing objects immersed in R^4 . *International Conference on Computational Science and Its Applications (ICCSA)*. (págs. 479-493.). Cham: Springer. doi:https://doi.org/10.1007/978-3-030-24311-1_35




Eiddec
EDITORIAL