# Introducción a GNU Octave

## Aplicaciones en ingeniería

CMCM

*Rev.02A*

30 de abril de 2014

# Índice general


**1. INTRODUCCIÓN** **4**

1.1. ¿Qué es Octave? . . . 4

1.2. ¿Cómo se escriben los números? . . . 4

1.3. ¿Cómo escribir las operaciones aritméticas elementales ? . . . 5

1.4. Orden en que se realizan las operaciones aritméticas . . . 5

1.5. Variables . . . 6

1.6. Variables predefinidas . . . 7

1.7. Funciones matemáticas elementales . . . 8

1.8. Números complejos . . . 9

**2. VECTORES Y MATRICES** **11**

2.1. Definición de vectores y matrices . . . 11

2.1.1. Definir vectores . . . 11

2.1.2. Definir matrices . . . 11

2.2. Operaciones con vectores y matrices . . . 12

2.3. Resolución de sistemas lineales de ecuaciones . . . 15

**3. REPRESENTACIÓN GRÁFICA DE FUNCIONES** **17**

3.1. Curvas planas . . . 17

3.2. Curvas en el espacio . . . 23

3.3. Superficies . . . 23

**4. CÁLCULO DE RAÍCES DE ECUACIONES, MÍNIMOS DE FUNCIONES E INTEGRALES DEFINIDAS** **26**

4.1. Resolución de ecuaciones no lineales . . . 26

4.2. Raíces de polinomios . . . 29

4.3. Resolución de sistemas de ecuaciones no lineales . . . 30
4.4. Mínimos de funciones . . . 33
4.4.1. Funciones de una variable . . . 33
4.4.2. Funciones de varias variables . . . 34
4.5. Cálculo de integrales definidas . . . 35
4.5.1. Integral definida de un conjunto de datos . . . 35
4.5.2. Integral definida de una función . . . 36
4.5.3. Integrales dobles . . . 38

**5. RESOLUCIÓN NUMÉRICA DE ECUACIONES DIFERENCIALES ORDINARIAS 40**

5.1. Problemas de valor inicial para ecuaciones diferenciales ordinarias . . . 40
5.1.1. Ejemplo: EDO lineal. LEY DE ENFRIAMIENTO DE NEWTON . . . 41
5.1.2. Ejemplo :EDO no lineal. TEOREMA DE TORRICELLI . . . 42
5.1.3. Ejemplo :EDO con funciones discontinuas. TANQUES DE MEZCLA. . . . 43
5.2. Problemas de valor inicial para ecuaciones diferenciales ordinarias de orden superior (n>1) . . . 44
5.2.1. Ejemplo: EDO 2º Orden. ECUACIÓN DE VAN DER POL . . . 44
5.3. Problema de valor inicial para sistemas de ecuaciones diferenciales ordinarias . . . 45
5.3.1. Ejemplo: Sistema EDO. MODELO DE LOTKA-VOLTERRA . . . 46

**6. OPTIMIZACIÓN 47**

6.1. INVESTIGACIÓN OPERATIVA. Programación lineal . . . 47
6.1.1. APLICACIONES DE LA PROGRAMACIÓN LINEAL A LA LOGÍSTICA . . . 50
6.1.2. APLICACIONES DE LA PROGRAMACIÓN LINEAL A PROBLEMAS DE BLENDING O MEZCLA. 51

CAPÍTULO 1

# INTRODUCCIÓN

## 1.1. ¿Qué es Octave?

GNU Octave es un lenguaje de alto nivel destinado para el cálculo numérico. Provee una interfaz sencilla, orientada a la línea de comandos (consola), que permite la resolución de problemas numéricos, lineales y no lineales, además permite la ejecución de scripts y puede ser usado como lenguaje orientado al procesamiento por lotes.

Octave nació alrededor del año 1988, y fue concebido originalmente para ser usado en un curso de diseño de reactores químicos para los alumnos de Ingeniería Química de la Universidad de Texas y la Universidad de Wisconsin-Madison.

Octave posee una gran cantidad de herramientas que permiten resolver problemas de algebra lineal, cálculo de raíces de ecuaciones no lineales, integración de funciones ordinarias, manipulación de polinomios, integración de ecuaciones diferenciales ordinarias y ecuaciones diferenciales algebraicas. Sus funciones también se pueden extender mediante funciones definidas por el usuario escritas en el lenguaje propio de Octave o usando módulos dinámicamente cargados escritos en lenguajes como C, C++ y Fortran entre otros.

## 1.2. ¿Cómo se escriben los números?

- Enteros (sin punto decimal):

```
>> 23 321 −34
```

- Reales (con punto decimal):

```
>> 23. −10.1 11.321
```

- Reales (notación científica o exponencial):

```
>> 2.e−2 = 2*10^(−2) = 0.02
>> 2.1e+5 = 2.1*10^(5) = 210000
```

**Atención:** *para separar la parte entera de la parte decimal hay que usar PUNTO DECIMAL.*

## 1.3. ¿Cómo escribir las operaciones aritméticas elementales ?

**Tabla 1.1.** Operaciones aritméticas elementales. [Fuente:GNU Octave]

| Operación | Símbolo Octave |
|---|---|
| Suma | + |
| Resta | - |
| Multiplicación | * |
| División | / |

```
>> 2.01*4*3.1416
>> −2.98+0.23−14+2
>> 6+4/2+3.111
>> 5.22*3.1416/6−4
```

Se puede utilizar **Octave** como simple calculadora, escribiendo expresiones aritméticas y terminando por **RETURN (<R>)**. Se obtiene el resultado inmediatamente a través de la variable del sistema **ans** (de answer). Si no se desea la secuencia (es decir, la respuesta inmediata a cada orden) en el terminal, deben terminarse las órdenes por "**punto y coma**".

## 1.4. Orden en que se realizan las operaciones aritméticas

Cuando en una expresión hay varios operadores aritméticos, el orden en que se realizan las operaciones es determinante: las operaciones **NO SE EFECTÚAN SIEMPRE EN EL ORDEN EN QUE ESTÁN ESCRITAS**. El orden viene determinado por las reglas siguientes:

1. Potencias.
2. Multiplicaciones y divisiones.
3. Sumas y restas.
4. Dentro de cada grupo, de izquierda a derecha.

**PARA MODIFICAR ESTE ORDEN SE USAN PARÉNTESIS**:

5. Si hay paréntesis, su contenido se calcula antes que el resto
6. Si hay paréntesis anidados, se efectúan primero los más internos

```
>> 2+3*4 = 2+(3*4) = 2+12 = 14
>> (2+3)*4 = 5*4 = 20
>> 1/3*2 = (1/3)*2 = 0.3333*2 = 0.6666
>> 1/(3*2) = 1/6 = 0.1666
>> 2+3^4/2 = 2+(3^4)/2 = 2+81/2 = 2+(81/2) = 2+40.5 = 42.5
>> 2+3^(4/2) = 2+3^2 = 2+(3^2) = 2+9 = 11
>> (2+3^4)/2 = (2+(3^4))/2 = (2+81)/2 = 83/2 = 41.5
```

**Ejemplo 1.**

Escribir en OCTAVE:

$$\frac{3+4^2}{\frac{2}{\sqrt[5]{3}}+\left(\frac{1}{3{,}1\cdot 2}\right)^{\frac{3}{4}}}$$

```
>> (3+4^2)/((2/3^(1/5))−(1/(3.1*2))^(3/4))
```

**Ejemplo 2.**

Escribir en OCTAVE:

$$\frac{1}{\frac{2}{(0,1)^{\frac{1}{2}}}-\frac{0,4}{(2)^{\frac{1}{3}}}}$$

```
>> 1/((2/0.1^(1/2))−(0.4/2^(1/3)))
```

**Ejemplo 3.**

Escribir en OCTAVE:

$$\frac{(4,1)^{\frac{0,2+1}{2}}}{\frac{2}{0,1^{\frac{1}{2}}}-\frac{0,4}{2^{\frac{1}{3}}}}$$

```
>> 4.1^((0.2+1)/2)/(2/0.1^(1/2)−0.4/2^(1/3))
```

## 1.5. Variables

Una VARIABLE es un nombre simbólico que identifica una parte de la memoria, y en la que podemos guardar números u otro tipo de datos. ES UN “SITIO” EN LA MEMORIA DEL ORDENADOR PARA “GUARDAR” DATOS.

El contenido de una variable lo podemos recuperar y modificar cuantas veces queramos, a lo largo de una sesión de trabajo. Se le pueden dar a las variables los nombres que queramos, formados por letras y números, hasta un máximo de 19, y comenzando por una letra.

No se pueden utilizar los caracteres especiales: $+ - = * \wedge < >$ ...

**Atención:** *OCTAVE distingue entre letras mayúsculas y minúsculas*

EJEMPLOS DE NOMBRES DE VARIABLES

| Variables | Variables | Variables |
|---|---|---|
| a | peso | Tierra |
| XY | Peso | Juan |
| ab12 | PESO | Plata |
| Pascal | PeSo | TESLA |

Las variables en OCTAVE no necesitan ningún tipo de declaración y pueden almacenar sucesivamente distintos tipos de datos: enteros, reales, escalares, matriciales, caracteres, etc. Para CREAR una variable basta con asignarle un valor. Para ASIGNAR un valor a una variable se utiliza una instrucción de asignación:

>> **nombre de variable = expresión**

```
>> ab=321
>> AB=3
>> x1=1/2
>> y1=1−4^2
```

## 1.6. Variables predefinidas

Algunos nombres están predefinidos por OCTAVE:

EJEMPLOS VARIABLES PREDEFINIDAS

| Variables | Función |
|---|---|
| **ans** | Variable del sistema para almacenar el resultado de evaluar expresiones |
| **i , j** | unidad imaginaria : raiz cuadrada de -1 |
| **pi** | número $\pi$ |
| **Inf** | "Infinito": número mayor que el más grande que se puede almacenar |
| **NaN** | "Not a Number : magnitud no numérica resultado de cálculos indefinidos |

**Ejemplo 4** (EJEMPLOS VARIABLES PREDEFINIDAS)**.**

Escribir en OCTAVE:

$$\frac{(4,1)^{\frac{0,2+1}{2}}}{\frac{2}{0,1^{\frac{1}{2}}}-\frac{0,4}{2^{\frac{1}{3}}}}$$

## 1.7. Funciones matemáticas elementales

A continuación se ofrece una lista con las funciones matemáticas más comunes que posee OCTAVE.

**Tabla 1.2.** Funciones elementales

| Variables | Función |
|---|---|
| **sqrt(x)** | Raíz cuadrada: $\sqrt{x}$ |
| **exp(x)** | Exponencial: $e^x$ |
| **log(x)** | Logaritmo neperiano: $ln(x)$ |
| **log10(x)** | Logaritmo decimal: $log_{10}(x)$ |
| **abs(x)** | valor absoluto: $\|x\|$ |
| **sign(x)** | devuelve el signo del argumento |

**Tabla 1.3.** Funciones Trigonométricas

| Directa | Función | Inversa | Función |
|---|---|---|---|
| **sin(x)** | Seno (en radianes) | **asin(x)** | Arco-seno |
| **cos(x)** | Coseno (en radianes) | **acos(x)** | Arco-coseno |
| **tan(x)** | Tangente (en radianes) | **atan(x)** | Arco-tangente |
| **sec(x)** | Secante | **asec(x)** | Arco-secante |
| **csc(x)** | Cosecante | **acsc(x)** | Arco-cosecante |
| **cot(x)** | Cotangente | **acot(x)** | Arco-cotangente |

**Tabla 1.4.** Funciones hiperbólicas

| Directa | Función | Inversa | Función |
|---|---|---|---|
| **sinh(x)** | Seno hiperbólico | **asinh(x)** | Arco-seno hiperbólico |
| **cosh(x)** | Coseno hiperbólico | **acosh(x)** | Arco-coseno hiperbólico |
| **tanh(x)** | Tangente hiperbólica | **atanh(x)** | Arco-tangente hiperbólica |

El argumento de las funciones puede ser un número, una variable o una expresión conteniendo ambas cosas. Cuando en una expresión aparece alguna función, su valor se calcula antes que cualquier otra cosa.

```
>> sqrt(7)
>> sqrt(7/5)
>> a=2.1; sqrt(2*a)
>> exp(3)
>> exp(x)
>> 7*exp(5/4)+3.54
```

## 1.8. Números complejos

OCTAVE utiliza la forma binómica para representar un número complejo, $a + bi$, donde $a$ es la parte real, $b$ la parte imaginaria e $i = j = \sqrt{-1}$.

**Forma binómica:**

```
>> z1=sqrt(−4)
>> z2=3+4i
>> z3=3+4*j
```

También, se puede trabajar con números complejos en su forma polar y en su forma exponencial.

$$z = a + jb = \mathrm{Re}[z] + j\,\mathrm{Im}[z]$$
$$z_e = \rho \cdot e^{j\theta} = \sqrt{a^2 + b^2} \cdot e^{j\tan^{-1}(b/a)}$$
$$z_p = \sqrt{a^2 + b^2}\angle\tan^{-1}(b/a)$$

**Forma polar y exponencial:**

```
>> Z=3+4j , rho=abs(Z), theta=angle(Z);
>> Zp=rho*(cos(theta)+j*sin(theta))
>> Ze=rho*exp(j*theta)
```

**Operaciones básicas con números complejos:**

Las operaciones matemáticas con números complejos se escriben de la misma forma que con números reales.

```
>> z1 = 3 + 2i;
>> z2 = 4 − i;
>> z1+z2
>> z2−z1
>> z1*z2
>> z2/z1
>> z1^2
>> sin(z2)
```

**Funciones de manipulación de complejos:**

**Tabla 1.5.** Funciones de manipulación de complejos:

| Variables | Función |
|---|---|
| **z= complex(a,b)** | Definir número complejo |
| **abs(z)** | Módulo de un número complejo |
| **angle(z)** | Argumento de un número complejo |
| **conj(z)** | Conjugado del número complejo |
| **imag(z)** | Parte imaginaria del número complejo |
| **real(z)** | Parte real de un número complejo |

**Ejemplo 5** (Operarciones con números complejos)**.**

Se considera $z_1 = 3 + 4i$ , $z_2 = 4 + 3i$ . Determinar, con OCTAVE, los siguientes apartados:

1. Realizar las siguientes operaciones:

$$z_1 + z_2, z_1 \cdot z_2, \bar{z}_1, \frac{z_1}{z_2}, {z_1}^2$$

2. Calcular el módulo y el argumento de $z_1$.

3. Escribir la forma trigonométrica y exponencial de $z_1$.

4. Calcular $sin(z_1)$, $cos(z_1)$.

```
>> % Apartado (1)
>> z1=3+4i
>> z1_barra=conj(z1)
>> z2=4+3i
>> z_suma=z1+z2
>> z_producto=z1*z2
>> z_cociente=z1/z2
>> z_potencia=z1^2
>> % Apartado (2)
>> ro=sqrt(z1*z1_barra)
>> alfa=atan(b/a)
>> ro=abs(z1)
>> alfa=angle(z1)
>> % Apartado (3)
>> z1_trig=ro*(cos(alfa)+i*sin(alfa))
>> z1_polar=ro*exp(i*alfa)
>> % Apartado (4)
>> sin(z1)
>> cos(z1)
```

CAPÍTULO 2

# VECTORES Y MATRICES

## 2.1. Definición de vectores y matrices

### 2.1.1. Definir vectores

Un *vector-fila* de dimension **n** se puede definir en OCTAVE escribiendo sus componentes entre corchetes rectos (**[ ]**) y separándolos por comas o espacios en blanco:

```
>> v=[1,−1,0,2.88]
 v =
   1.00000 −1.00000 0.00000 2.88000
```

La orden anterior crea en OCTAVE una variable de nombre **v** que "contiene" un *vector-fila* de longitud 4.

Un *vector-columna* se crea igual, pero separando las componentes por "punto y coma":

```
>> w=[0;1;2;3;4;5]
w =
          0
          1
          2
          3
          4
          5
```

crea una variable de nombre **w**, que "almacena" un vector-columna de longitud 6.

### 2.1.2. Definir matrices

Las matrices se definen de forma similar a los vectores, introduciendo sus filas como vectores-fila y separando unas filas de otras mediante punto y coma o saltos de línea.

```
>> A=[1,2,3 ; 4,5,6 ; 7,8,9]
A =
        1 2 3
        4 5 6
        7 8 9
```

Si **A** y **B** son dos matrices que tienen el mismo número de filas, entonces **[A,B]** es la matriz que se obtiene añadiendo B al lado de A. Análogamente, si **A** y **B** tienen el mismo número de columnas, entonces **[A;B]** es la matriz que se obtiene añadiendo **B** debajo de **A**:

**Ejemplo 6** (Definición de matrices).

```
>> A=[1,2;3,4]
>> B=[1;1]
>> C=[A,B]
```

$$A=\begin{pmatrix}1&2\\3&4\end{pmatrix};B=\begin{pmatrix}1\\2\end{pmatrix}$$
$$C=[A,B]=\begin{pmatrix}1&2&1\\3&4&1\end{pmatrix}$$

## 2.2. Operaciones con vectores y matrices

Si son de las mismas dimensiones, los *vectores/matrices* se pueden sumar y restar.

**Ejemplo 7** (Suma y resta).

```
>> v=[1;−3;0]
>> w=[0;3;−2]
>> z=v+w
```

$$v=\begin{pmatrix}1\\-3\\0\end{pmatrix}\quad w=\begin{pmatrix}0\\3\\-2\end{pmatrix}$$
$$z=v+w=\begin{pmatrix}1\\0\\-2\end{pmatrix}$$

Los *vectores/matrices* se pueden multiplicar por un número; se multiplica cada elemento por dicho número.

**Ejemplo 8** (producto por un escalar).

```
>> A=[1, 2;−3, −1];
>> z=3*A;
```

$$A=\begin{pmatrix}1&2\\-3&-1\end{pmatrix}$$
$$z=3\cdot A=\begin{pmatrix}3&6\\-9&-3\end{pmatrix}$$

Una matriz se puede multiplicar por un vector columna si coincide el número de columnas de la matriz con la longitud del vector.

**Ejemplo 9** (Producto de vectores / matrices).

```
>> A=[1, 2;−3, −1]
>> v=[2;−1]
>> z=A*v
```

$$A = \begin{pmatrix} 1 & 2 \\ -3 & -1 \end{pmatrix}; v = \begin{pmatrix} 2 \\ -1 \end{pmatrix}$$

$$z = A \cdot v = \begin{pmatrix} 0 \\ -5 \end{pmatrix}$$

Las setencias:

```
>>det(A)
>>rank(A)
```

permiten calcular, respectivamente, el determinante y el rango de una matriz A.

**Tabla 2.1.** Funciones de operación vectorial:

| Función | Descripción |
|---|---|
| **cart2pol** | transforma coordenadas cartesianas a polares |
| **cart2sph** | transforma coordenadas cartesianas a esféricas |
| **pol2cart** | transforma coordenadas polares a cartesianas |
| **sph2cart** | transforma coordenadas esféricas a cartesianas |
| **cross** | producto vectorial |
| **dot** | producto escalar |
| **norm** | módulo de un vector |

**Ejemplo 10** (Operaciones vectoriales).

Dados los vectores:

$$\vec{a} = 3\vec{i} - 1\vec{j} + 2\vec{k}$$
$$\vec{b} = 1\vec{i} + 1\vec{j} - 2\vec{k}$$

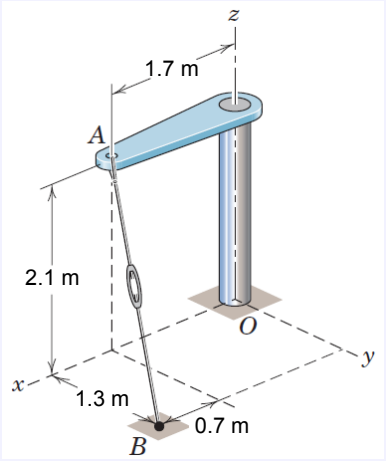
Calcular, con OCTAVE, los siguientes apartados:

1. El producto escalar de $\vec{a}$ y $\vec{b}$.
2. El producto vectorial de $\vec{a}$ y $\vec{b}$.
3. El vector unitario perpendicular al plano formado por los dos vectores.

```
>> a = [3 −1 2];
>> b = [1 1 −2];
>> %Apartado (1)
>> S=dot(a,b)
>> %Apartado (2)
>> V=cross(a,b)
>> %Apartado (3)
>> N=norm(V)
>> U=V/N
```

**Ejemplo 11** (Mecánica vectorial).

El tensor de la figura, se ajusta hasta que la tensión del cable $AB$ es de $2,5\text{kN}$. Determinar, con OCTAVE, el momento respecto al punto O de la tensión del cable ,que actúa en el punto A, y la magnitud de ese momento.




```
>> n_AB=[0.7 1.3 −2.1];
>> T = 2.5*(n_AB/norm(n_AB));
>> r = [1.7 0 2.1];
>> M_o = cross(r,T);
>> M_mag = norm(M_o);
>> fprintf ('M_o = r x T = ( %g)i + ( %g)j + ( %g)k [kN.m]\n',M_o(1),M_o(2),M_o(3));
>> fprintf('M_mag = |M_o| = %1.4f [kN.m]\n', M_mag)
```

## 2.3. Resolución de sistemas lineales de ecuaciones

Un sistema lineal de ecuaciones:

$$\begin{cases} a_{11}x_1 & + & a_{12}x_2 & + & \dots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \dots & + & a_{2n}x_n & = & b_2 \\ \vdots & & & & \vdots & & \vdots & & \\ a_{n1}x_1 & + & a_{n2}x_2 & + & \dots & + & a_{nn}x_n & = & b_n \end{cases}$$

se puede escribir en forma matricial Ax=b, con

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}; b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

Conocidos **A** y **b**, el problema de encontrar **x** tal que **Ax=b** se resuelve fácilmente con OCTAVE, con la orden:

```
>>x=A\b
```

**Ejemplo 12** (Resolución de sistemas lineales).

Resolver el sistema lineal de ecuaciones:

$$\begin{array}{rcrcrcr} 3x_1 & + & 2x_2 & - & x_3 & = & 1 \\ 2x_1 & - & x_2 & + & x_3 & = & -1 \\ x_1 & - & 2x_2 & + & x_3 & = & 2 \end{array}$$

Definimos la matriz y el segundo miembro del sistema.

```
>> A=[3,2,−1;2,−1,1;1,−2,1];
>> b=[1;−1;2];
```

Determinamos los rangos. De este modo, como ambos rangos son iguales a 3, el sistema es compatible determinado, es decir, con solución única.

```
>> rank(A)
>> rank([A,b])
```

Calculamos y comprobamos la solución del sistema.

```
>> x=A\b
>> A*x
```

**Solución OCTAVE :**

**x = 0.75000**

**-3.75000**

**-6.25000**

**Ejemplo 13** (Modelo de petróleo refinado)**.**

Una compañía petrolera dispone de tres refinerías de petróleo. Estas se denominan de la siguiente forma: Refinería 1, Refinería 2 y Refinería 3. Cada refinería produce tres productos basados en el crudo: Alquitrán, Gasóleo y Gasolina. Supongamos que, de un barril de petróleo, se sabe que:

- la primera refinería produce 4 litros de alquitrán, 2 de gasóleo, y 1 de gasolina.
- la segunda refinería produce 2 litros de alquitrán, 5 de gasóleo y 2.5 de gasolina.
- y la tercera refinería produce 2 litros de alquitrán, 2 de gasóleo y 5 de gasolina.

Supongamos que hay una demanda de estos productos de la siguiente manera:

- 600 litros de alquitrán.
- 800 litros de gasóleo.
- 1000 litros de gasolina.

¿Cuántos barriles de crudo necesitará cada refinería para satisfacer la demanda?.
El enunciado se puede representar de la siguiente forma:

$$\begin{array}{rcrcrcr} 4x_1 & + & 2x_2 & + & 2x_3 & = & 600 \\ 2x_1 & + & 5x_2 & + & 2x_3 & = & 800 \\ 1x_1 & + & 2{,}5x_2 & + & 5x_3 & = & 1000 \end{array}$$

Definimos la matriz y el segundo miembro del sistema.

```
>> A=[4 2,2; 2,5,2; 1,2.5,5];
>> b=[600;800;1000];
```

Calculamos y comprobamos la solución del sistema.

```
>> x=A\b
>> A*x
```

**Solución OCTAVE :**
**La Refinería 1 necesitará 31.25 barriles de crudo**
**La Refinería 2 necesitará 87.50 barriles de crudo**
**La Refinería 3 necesitará 150.00 barriles de crudo**

CAPÍTULO 3

# REPRESENTACIÓN GRÁFICA DE FUNCIONES

## 3.1. Curvas planas

La forma más sencilla de dibujar, con OCTAVE, una función y=f(x) es con la orden:

>> **ezplot('expresion de la funcion')**

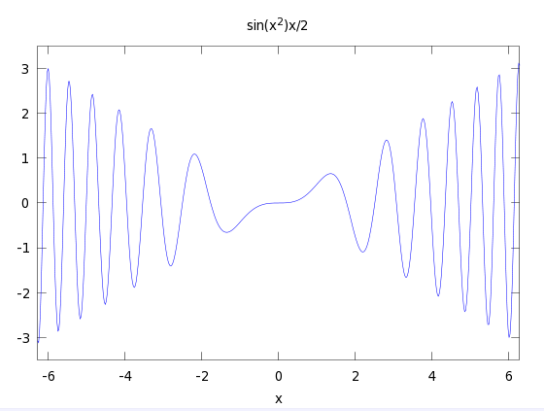
**Ejemplo 14.**

Dibuja la gráfica de la función:

$$f(x) = \frac{x \cdot \sin(x^2)}{2} \quad \text{en} \quad [-2\pi, 2\pi]$$

```
>> ezplot('sin(x^2)*x/2')
```

Esta orden dibuja la gráfica de la función dada por la expresión anterior, para x variando en el intervalo $[-2\pi, 2\pi]$.




Si se quiere dibujar la función en un intervalo distinto, [a,b], hay que indicarlo expresamente en la orden:

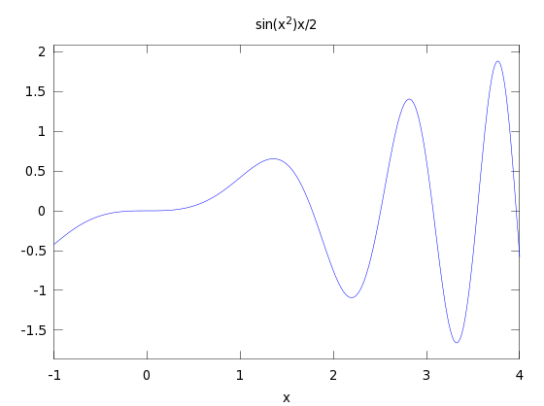
>> **ezplot('expresion de la funcion',[a,b])**

**Ejemplo 15.**

Dibuja la gráfica de la función:

$$f(x) = \frac{x \cdot \sin(x^2)}{2} \quad \text{en} \quad [-1, 4]$$

```
>> ezplot('sin(x^2)*x/2',[-1,4])
```




Si no se indica el intervalo y la función que se quiere dibujar no está definida en todo el intervalo $[-2\pi, 2\pi]$, OCTAVE la dibujará sólo en el intervalo en que esté definida.

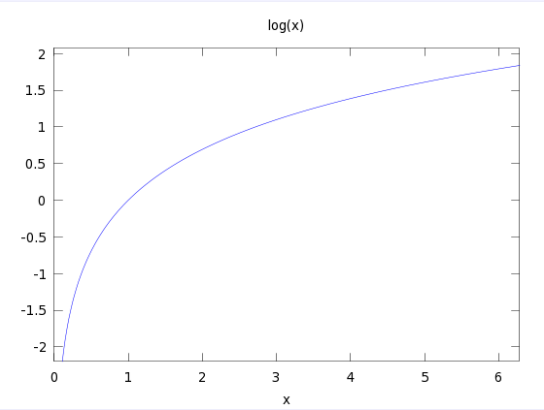
**Ejemplo 16.**

Dibuja la gráfica de la función:

$$f(x) = \ln(x)$$

```
>> ezplot('log(x)')
```

La función $ln(x)$ sólo está definida para $x > 0$; la orden anterior dibuja su gráfica en $[0, 2\pi]$



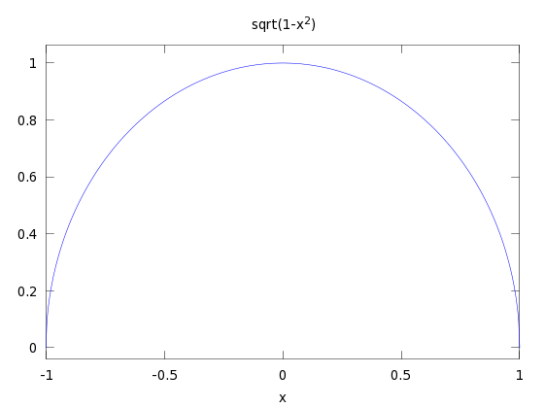
**Ejemplo 17.**

Dibuja la gráfica de la función:

$$f(x) = \sqrt{1^2 - x^2}$$

```
>> ezplot('sqrt(1−x^2)')
```

La función sólo está definida en $[-1, 1]$.

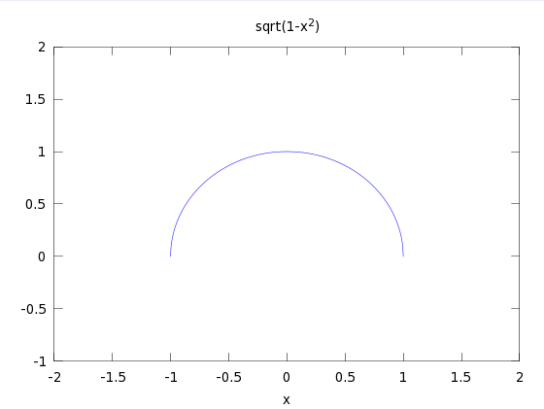



Una vez que se ha representado una gráfica, se puede modificar la amplitud de los ejes (el recinto del plano XY que es visible), mediante la orden:

```
>>axis([ xmin , xmax , ymin , ymax])
```

**Ejemplo 18** (MODIFICACIONES DE LAS GRÁFICAS).

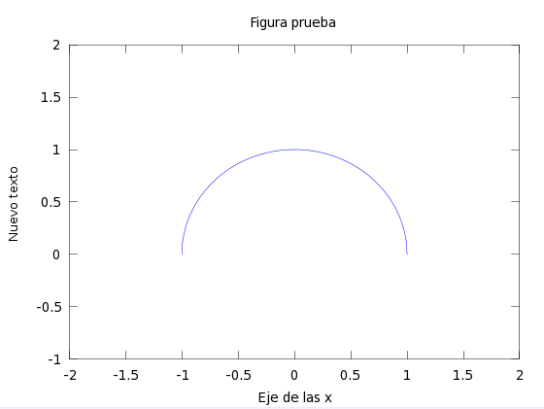
```
>> ezplot('sqrt(1−x^2)')
>> axis([−2,2,−1,2])
```



**Ejemplo 19** (MODIFICACIONES DE LAS GRÁFICAS).

Se puede añadir un título y etiquetas a los ejes:

```
>> title('Figura prueba')
>> xlabel('Eje de las x')
>> ylabel('Lo que quieras')
```



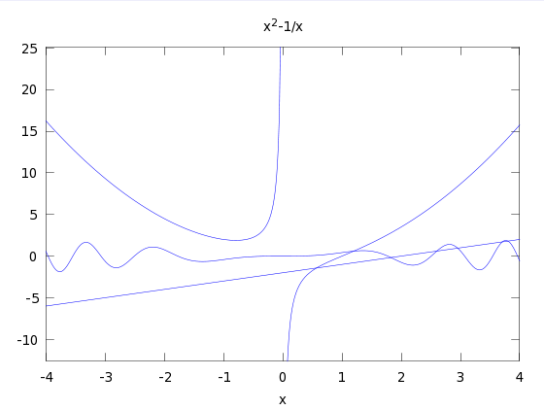

Cada vez que se dibuja una gráfica nueva se borra la anterior, si la había. Si se desean hacer varias gráficas, “una encima de otra", sin que se borren las anteriores, se pueden usar las siguientes órdenes:

```
>>hold on
...
>>hold off
```

La orden **hold on** hace que no se borre el contenido de la ventana gráfica cuando se den nuevas órdenes de dibujo. Se suspende con **hold off**.

**Ejemplo 20** (AÑADIR NUEVAS GRÁFICAS).

```
>> ezplot('sin(x^2)*x/2',[-4,4])
>> hold on
>> ezplot('x-2',[-4,4])
>> ezplot('x^2-1/x',[-4,4])
>> hold off
```




También se pueden dibujar varias **gráficas separadas** en la misma ventana, usando la orden:

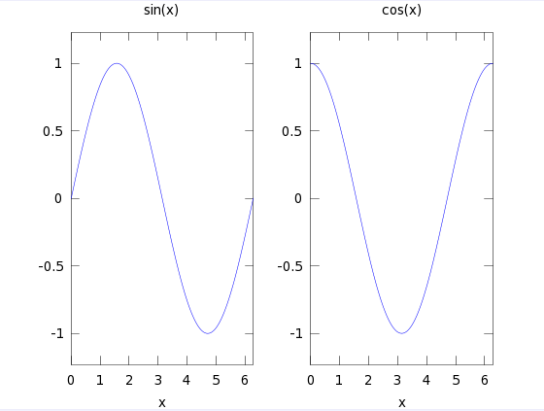
```
>>subplot(m,n,p)
```

Esta orden divide la ventana gráfica, en subgráficos de m filas y n columnas, y se dispone a dibujar en el p-ésimo de ellos. Los ejes se numeran correlativamente, de izquierda a derecha y de arriba hacia abajo.

**Ejemplo 21** (SUBGRÁFICAS DE 1 FILA Y 2 COLUMNAS).

Representar en dos subgráficas una fila y dos columnas:

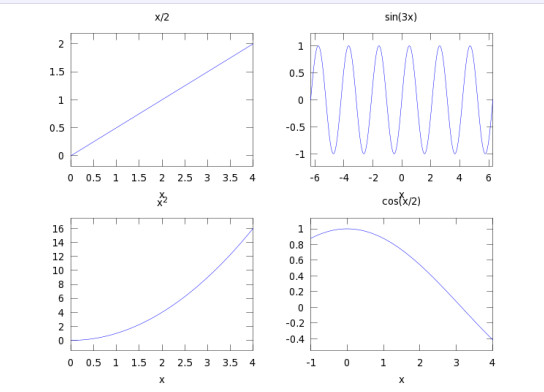
```
>> ezplot('sin(x^2)*x/2',[-4,4])
>> hold on
>> ezplot('x-2',[-4,4])
>> ezplot('x^2-1/x',[-4,4])
>> hold off
```




**Ejemplo 22** (SUBGRÁFICAS DE 2 FILAS Y 2 COLUMNAS).

Representar cuatro subgráficas dos filas y dos columnas:

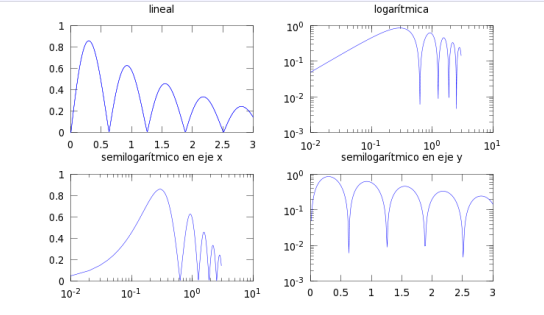
```
>> subplot(2,2,1)
>> ezplot('x/2',[0,4])
>> subplot(2,2,2)
>> ezplot('sin(3*x)')
>> subplot(2,2,3)
>> ezplot('x^2',[0,4])
>> subplot(2,2,4)
>> ezplot('cos(x/2)',[-1,4])
```



**Ejemplo 23** (SUBGRÁFICAS A DIFERENTES ESCALAS).

Representar cuatro subgráficas dos filas y dos columnas a diferentes escalas (lineal, logarítmica, semilogarítmica eje X, semilogarítmica eje Y) :

```
>> x = 0:0.01:3;
>> y = abs(exp(−0.5*x).*sin(5*x));
>> subplot(221);
>> plot(x,y)
>> title('lineal')
>> hold on
>> subplot(222)
>> loglog(x,y)
>> title('logaritmica')
>> subplot(223)
>> semilogx(x,y)
>> title('semilogaritmico en eje x')
>> subplot(224)
>> semilogy(x,y)
>> title('semilogaritmico en eje y')
```

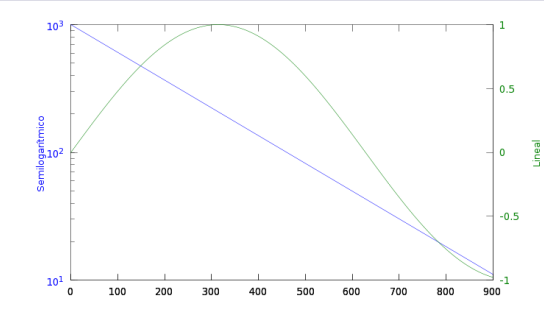



**Ejemplo 24** (EJES A DIFERENTES ESCALAS).

Representar dos gráficas con el eje y a diferentes escalas (normal y semilogarítmica eje Y) :
Para representar ejes a diferentes escalas

se utiliza el comando **plotyy**.

```
>> t = 0:900; A = 1000;
>> a = 0.005; b = 0.005;
>> z1 = A*exp(−a*t);
>> z2 = sin(b*t);
>> [haxes,hline1,hline2] = plotyy(t,z1,t,z2,'
      semilogy','plot');
>> axes(haxes(1)), ylabel('Semilogaritmico')
>> axes(haxes(2)), ylabel('Lineal')
>> set(hline2,'LineStyle','−−')
```



## 3.2. Curvas en el espacio

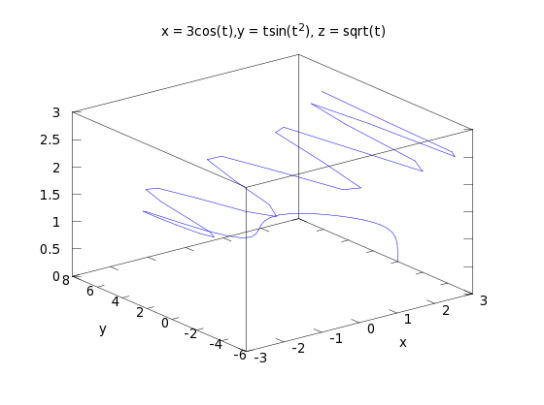
Para dibujar curvas en el espacio tridimensional, OCTAVE dispone del comando **ezplot3**:

>> **ezplot3(x,y,z)**
>> **ezplot3(x,y,z,[a,b])**
donde

- x , y , z son tres cadenas de caracteres conteniendo las expresiones de tres funciones x(t) , y(t) , z(t).
- Los comandos dibujan la curva de ecuaciones paramétricas $x = x(t)\ y = y(t)\ z = z(t)$ para $t$ en el intervalo [0,2$\pi$], en el primer caso y para $t$ en el intervalo [a,b] en el segundo.

**Ejemplo 25** (Curva en el espacio).

```
>> ezplot3('3*cos(t)','t*sin(t^2)','sqrt(t)')
```




## 3.3. Superficies

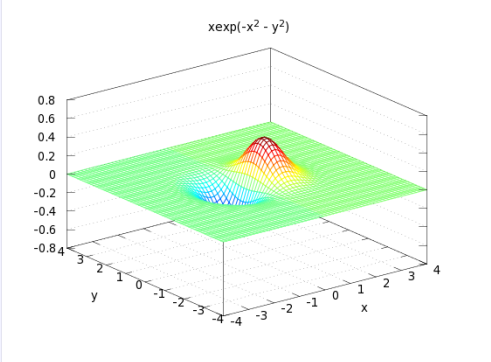
Para dibujar superficies en el espacio tridimensional,OCTAVE dispone de los siguientes comandos:

**Ejemplo 26** (Gráfico de una superficie con malla).

```
>> ezmesh(f)
>> ezmesh(f,[a,b])
>> ezmesh(f,[a,b,c,d])
```

donde, f es una expresión de dos variables. Representa la superficie $z = f(x,y)$ para $(x,y)$ considerando los intervalos indicados en la función.

```
>> ezmesh('x*exp(−x^2 − y^2)')
```

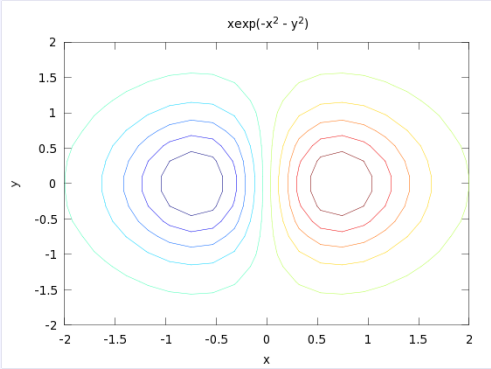


**Ejemplo 27** (Trazar curvas de nivel).

```
>> ezcontour(f)
>> ezcontour(f,[a,b])
>> ezcontour(f,[a,b,c,d])
```

Representa las curvas de nivel (isovalores) de la función $z = f(x, y)$

```
>> ezcontour('x*exp(-x^2 - y^2)')
```

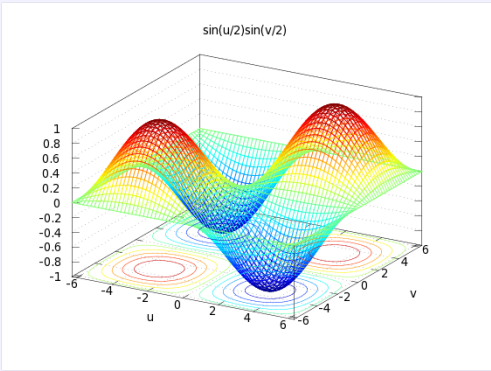



**Ejemplo 28** (Superficie con malla y curvas de nivel proyectadas en el plano xy).

```
>> ezmeshc(f)
```

Representa simultáneamente las líneas de nivel y la superficie.

```
>> ezmeshc('sin(u/2)*sin(v/2)')
```

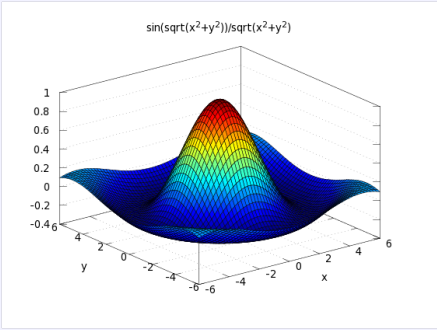



**Ejemplo 29** (Trazar el gráfico de una superficie.).

```
>> ezsurf(f)
```

Representa una superficie coloreada $z = f(x, y)$.

```
>> ezsurf('sin(sqrt(x^2+y^2))/sqrt(x^2+y^2)')
```

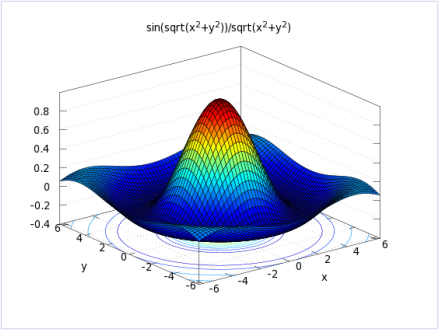


**Ejemplo 30** (Superficie con sus curvas de nivel proyectadas en el plano xy).

```
>> ezsurfc(f)
```

Representa simultáneamente las curvas de nivel y la superficie.

```
>>ezsurfc('sin(sqrt(x^2+y^2))/sqrt(x^2+y^2)')
```



CAPÍTULO 4

# CÁLCULO DE RAÍCES DE ECUACIONES, MÍNIMOS DE FUNCIONES E INTEGRALES DEFINIDAS

## 4.1. Resolución de ecuaciones no lineales

Para calcular con OCTAVE una raíz de la ecuación $f(x) = 0$ , es decir, un punto x en el cual la función $f$ vale $0$, se usa la orden:

>> **x = fzero(fun,x0)**
>> **[x,fval] = fzero(fun,x0)**

- **fun** es una definición de la función que calcula $f(x)$. Debe responder a la forma: **[y]=fun(x)**
- **x0** es el valor inicial de x , a partir del cual se comienza a "buscar" la solución. En general, debe ser un valor próximo a la solución buscada.
- **x** es la solución de la ecuación.
- **fval** (opcional) es el valor de f en la solución.

**Ejemplo 31** (Raíz simple.).

Calcular, con OCTAVE, una raíz de la ecuación:

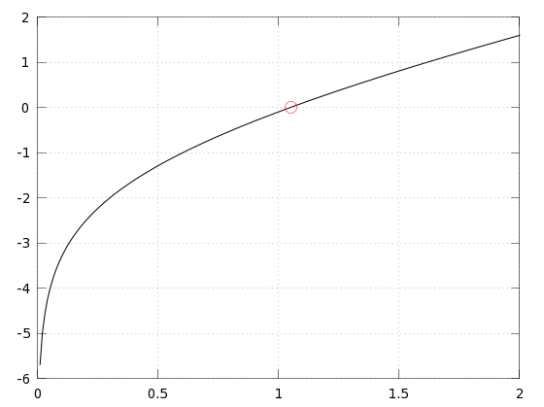
$$x + \log\left(\frac{x}{3}\right) = 0$$

```
>> ezplot('x+log(x/3)')
```

Vemos, *a simple vista*, que la raíz está cerca de x=1.

```
>> fun = @(x) x+log(x/3)
>> z = fzero(fun,1)
```

**Solución OCTAVE: z = 1.0499**




En el mejor de los casos, **fzero** sólo calcula **UNA** solución de la ecuación. En caso de múltiples soluciones, hay que utilizarla una vez para cada solución que interese encontrar, y tiene mucha importancia, el hecho de que el punto inicial, $\boldsymbol{x_0}$, esté próximo a la solución.

**Ejemplo 32** (Raíz múltiple).

Calcular, con OCTAVE, las raíces de la ecuación:

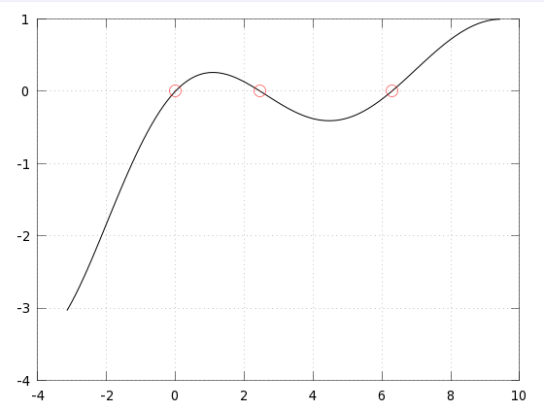
$$\sin\left(\frac{x}{2}\right) + \cos\left(\sqrt{x}\right) = 0 \quad \text{en } [\pi, 3\pi]$$

```
>> ezplot('sin(x/2)*cos(sqrt(x))',[-pi,3*pi])
```

A “simple vista"se observa que tiene 3 raíces: una “cerca"de x=0, otra “cerca"de x=2 y otra “cerca"de x=6

```
>> fun = @(x)sin(x/2).*cos(sqrt(x))
>> z1 = fzero(fun,0)
>> z2 = fzero(fun,2)
>> z3 = fzero(fun,6)
```

**Solución OCTAVE: z1 = 0.0 ; z2 = 2.4674 ; z3 = 6.2832**



**Ejemplo 33** (Ecuación cartesiana de la catenaria)**.**

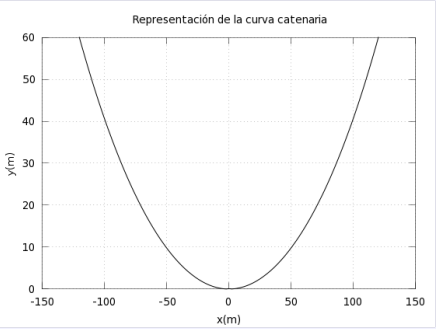
Un cable de una línea de transmisión de energía eléctrica, cuyos anclajes están a la misma altura tiene una cuerda de 240 [m]. El cable pesa 83 [N/m] y la flecha en su punto medio es de 60 [m]. Determinar, con OCTAVE, la tensión del cable en el punto medio. La ecuación cartesiana de la catenaria se expresa de la siguiente forma :

$$y(x) = \frac{T_0}{w} \cdot \left( \cosh\left( \frac{w \cdot x}{T_0} \right) - 1 \right)$$

La tensión del cable en el punto medio puede determinarse utilizando la ecuación de la curva catenaria, que representa la forma que adopta el cable bajo la acción exclusiva de su peso.

$$60 = \left( \frac{T_0}{83} \right) \cdot \left( \cosh\left( \frac{83 \cdot 240}{T_0} \right) - 1 \right)$$

```
>> fun = @(x)sin(x/2).*cos(sqrt(x))
>> w=83;
>> L=240;
>> h=60;
>> T_0=w*L^2/(8*h);
>> fun = @(T_o) ((T_o/w)*(cosh((w*L)/(2*T_o))-1))-h;
>> T_o = fzero(fun,T_0);
```




**Solución OCTAVE:**

**Tensión del cable en su punto medio T = 10700.23 N**

## 4.2. Raíces de polinomios

Si lo que queremos calcular son las raíces de un polinomio:

$$c_1 x^N + c_2 x^{N-1} + \ldots + c_N x + c_{N+1} = 0$$

se puede usar la orden **roots**, que calcula TODAS las raíces del polinomio (incluídas las raíces complejas, si las tiene):

```
>> roots(p)
```

donde **p** es el vector cuyas componentes son los coeficientes del polinomio, ordenados en orden decreciente de potencias de $x$.

$$p = (c_1, c_2, \ldots, c_N, c_{N+1})$$

**Ejemplo 34** (Raíces de un polinomio).

Calcular, con OCTAVE, las raíces de la ecuación:

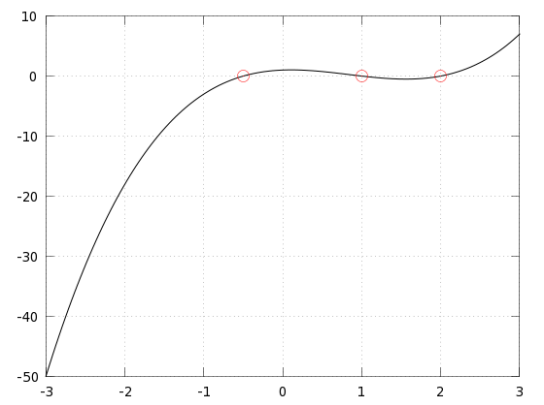
$$x^3 - \frac{5}{2}x^2 + \frac{1}{2}x + 1 = 0$$

Observamos que se trata de un polinomio de grado 3:

```
>> p=[1,−5/2,1/2,1]
>> sol=roots(p)
```

**Solución OCTAVE:**

**sol = 2.00000 1.00000 -0.50000**




**Ejemplo 35** (Raíces de un polinomio).

Calcular, con OCTAVE, las raíces de la ecuación:

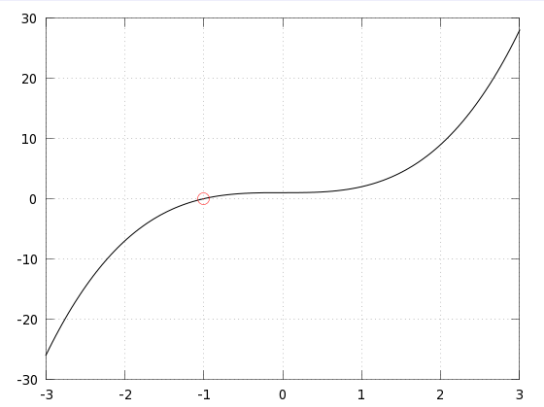
$$x^3 + 1 = 0$$

Observamos que se trata de un polinomio de grado 3:

```
>> p=[1,0,0,1]
>> sol=roots(p)
```

**Solución OCTAVE:**

**sol = -1.00000 0.50000 + 0.86603i**
**0.50000 - 0.86603i**



## 4.3. Resolución de sistemas de ecuaciones no lineales

Para resolver sistemas de ecuaciones no lineales:

$$\vec{f}(\vec{x}) = \vec{0}$$

OCTAVE dispone de la función *fsolve*, cuya utilización más sencilla es:

>> **[x, info , msg ] = fsolve (fcn , x0)**

- **fcn**: Nombre de la función que calcula las $\vec{f}(\vec{x})$.
- **x0**: Vector que contiene el valor inicial de las variables. Si el sistema tiene múltiples soluciones.
- **x**: Solución encontrada.
- **info**: Condición de terminación opcional.
- **msg**: Mensaje de terminación opcional.

**Ejemplo 36** (Sistemas de ecuaciones no lineales)**.**

Resolver, con OCTAVE, el siguiente sistema no lineal partiendo del punto inicial (x, y) = (1, 2).

$$f_1(x, y) = -2x^2 + 3xy + 4 \text{ sen } y - 6 = 0$$
$$f_2(x, y) = 3x^2 - 2xy^2 + 3 \cos x + 4 = 0$$

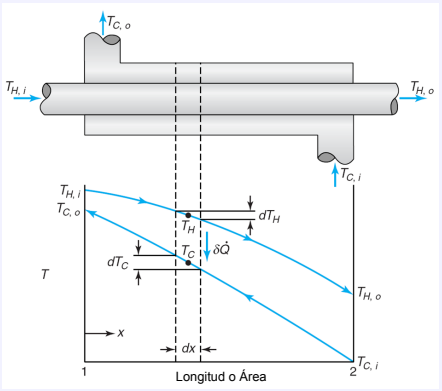
```
>> function y = f(x)
>> y(1) = -2*x (1)^2 + 3*x (1)* x (2) + 4* sin (x (2)) - 6;
>> y(2) = 3*x (1)^2 - 2*x (1)* x (2)^2 + 3* cos (x(1)) + 4;
>> endfunction
>> [x, info] = fsolve ("f", [1;2])
```

**Solución OCTAVE:**

**x = 0.57983 ; 2.54621**

**Ejemplo 37** (Transferencia de calor. Intercambiador de calor).

Se desea enfriar la corriente de productos de una columna de destilación, que fluye a razón de 4 $[\mathrm{kg/s}]$, por medio de una corriente de agua de 3 $[\mathrm{kg/s}]$ en un intercambiador en contracorriente. Las temperaturas de entrada de las corrientes caliente y fría son de 400 y 300 $[\mathrm{K}]$, respectivamente, y el área de transferencia de calor del intercambiador es de 30 $[\mathrm{m}^2]$. Si se estima que el coeficiente global de transferencia de calor es de 820 $[\mathrm{W/m^2K}]$.



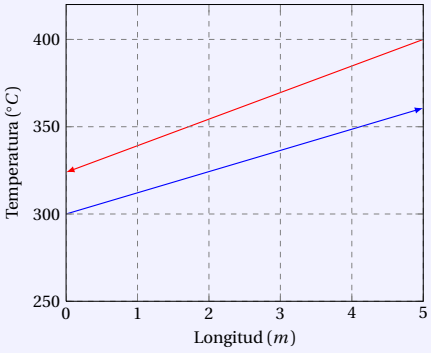

Determinar, con OCTAVE, la temperatura de salida de la corriente de productos. Puede suponerse que el calor específico de la corriente de productos es de 2500 $[\mathrm{J/kgK}]$.

Las ecuaciones del modelo son:

$$m_c \cdot Ce_c \cdot (Tc_2 - Tc_1) = m_f \cdot Ce_f \cdot (Tf_1 - Tf_2)$$

$$m_c \cdot Ce_c \cdot (Tc_2 - Tc_1) = U \cdot A \cdot \frac{(Tc_2 - Tf_2) - (Tc_1 - Tf_1)}{\log\left(\frac{Tc_2 - Tf_2}{Tc_1 - Tf_1}\right)}$$

```
>> global CpDest CpAgua mDest mAgua U A Tc2 Tf1
>> CpDest = 2500;  %[J/kg K]
>> CpAgua = 4180;  %[J/kg K]
>> mDest = 4.0; %[kg/s]
>> mAgua = 3.0;  %[kg/s]
>> U = 820;  %[W/h m^2]
>> A = 30;  %[m^2]
>> Tc2 = 400; %[K]
>> Tf1 = 300; %[K]
>> function fx = ecsistem(x)
>> global CpDest CpAgua mDest mAgua U A Tc2 Tf1
>> n = size(x,1);
>> fx = zeros(n,1);
>> Tc1 = x(1);
>> Tf2 = x(2);
>> fx(1) = mDest*CpDest*(Tc2-Tc1)-mAgua*CpAgua*(
       Tf2-Tf1);
>> fx(2) = mDest*CpDest*(Tc2-Tc1)-...
>> (U*A*((Tc2 - Tf2) - (Tc1 - Tf1))/log((Tc2 - Tf2)/(
       Tc1 - Tf1)));
>> x0 = [330;350];
>> [xs,info] = fsolve("ecsistem",x0);
```



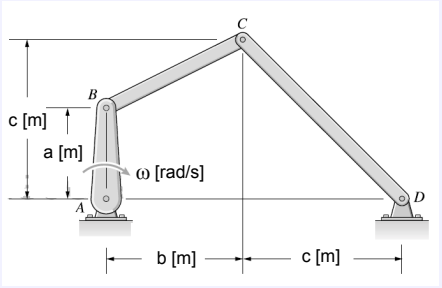

**Solución OCTAVE:**

**Tc1 = 323.874 [K]**

**Tf2 = 360.707 [K]**

**Ejemplo 38** (Cinemática del sólido rígido. Sistema de ecuaciones vectoriales).

En el sistema de la figura, la barra **AB** gira con velocidad angular de 12 $[\mathrm{rad/s}]$ en sentido horario. Determinar, con OCTAVE, las velocidades angulares de las barras **BC** y **CD** en el instante mostrado. Datos **a**=0.20 $[\mathrm{m}]$, **b**=0.30 $[\mathrm{m}]$, **c**=0.35 $[\mathrm{m}]$.




Las ecuaciones del sistema son:

$$\vec{v}_C = \vec{v}_A + \vec{\omega} \times \vec{r}_{B/A} + \vec{\omega}_{BC} \times \vec{r}_{C/B}$$
$$\vec{v}_C = \vec{v}_D + \vec{\omega}_{CD} \times \vec{r}_{C/D}$$

```
>> global w a b c
>> w=12;
>> a=0.2;
>> b=0.30;
>> c=0.35;
>> function y = Solid_BarK(x)
>> global w a b c
>> wBC=x(1);
>> wCD=x(2);
>> w_BA=[0 0 −w];
>> r_BA=[0 a 0];
>> wB_BC=[0 0 wBC];
>> rB_BC=[b c−a 0];
>> wB_CD=[0 0 wCD];
>> rB_CD=[−c c 0];
>> y=cross(w_BA,r_BA)+cross(wB_BC,rB_BC)−cross(wB_CD,rB_CD);
>> endfunction
>> sol= fsolve ("Solid_BarK",[1;1] );
>> fprintf ('\n w_ BC= (%g)k [rad/s]\n',sol(1));
>> fprintf ('\n w_ CD= (%g)k [rad/s]\n',sol(2));
```

**Solución OCTAVE:**

$w_{BC}$**= (5.33333) k [rad/s]**

$w_{CD}$**= (-4.57143) k [rad/s]**

## 4.4. Mínimos de funciones

### 4.4.1. Funciones de una variable

Para calcular el (punto en el que se produce el) mínimo de una función **y=f(x)** en un intervalo [a,b], OCTAVE dispone de la función **fminbnd**, cuya utilización más sencilla es:

>> **x = fminbnd(fun,a,b)**
>> **[x,fval] = fminbnd(fun,a,b)**

- **fun**: es un definición de la función que calcula f(x). Debe responder a la forma: **[y]=fun(x)**.
- **a,b** son los extremos del intervalo.
- **x** es una aproximación del punto que produce el mínimo.
- **fval** (opcional) es el valor de f en la solución.

**Ejemplo 39** (Mímino de una función de una variable)**.**

Calcular el mínimo de la función:

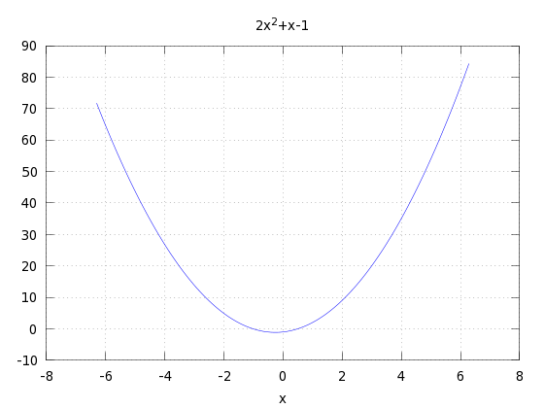
$$f(x) = 2x^2 + x + 1 \text{ en } [-2, 2]$$

```
>> fun=@(x) 2*x^2+x-1
>> [x,fval] = fminbnd(fun,-2,2)
```

**Solución OCTAVE:**

**x = -0.25000**

**fval = -1.1250**




Para calcular el máximo de una función $y = f(x)$ en un intervalo [a,b], hay que calcular el mínimo de la función $y = -f(x)$ en el mismo intervalo.

### 4.4.2. Funciones de varias variables

OCTAVE dispone de la función **fminunc** para calcular mínimos de funciones escalares de varias variables.

>> **x = fminunc(fun,x0)**
>> **[x,fval] = fminunc(fun,x0)**
Si la función a minimizar depende de **N** variables, entonces **fun** debe responder a la forma **[y]=fun(x)**, siendo $\boldsymbol{x}$ un vector de **N** componentes.

**Ejemplo 40** (Mímino de una función de varias variables)**.**

Calcular con OCTAVE, un mínimo de la función :

$$f : \mathbb{R}^2 \to R, \quad f(x) = \sin\left(\frac{x_1}{2}\right)\sin\left(\frac{x_2}{2}\right)$$

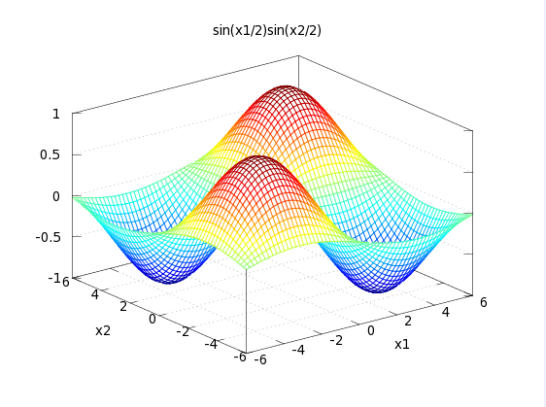
Iniciando el cálculo en (1.5,1).

```
>> fun = @(x) sin(x(1)/2)*sin(x(2)/2) ;
>> [x,fval]=fminunc(fun,[1.5,1.0])
```

**Solución OCTAVE:**

**x = 3.1416 -3.1416**

**fval = -1.00000**



## 4.5. Cálculo de integrales definidas

### 4.5.1. Integral definida de un conjunto de datos

OCTAVE cuenta con la función:

>>**v = trapz(x,y)**
que realiza la integración de una función $f$ dada como tabla de datos mediante la regla de los trapecios.

**Ejemplo 41** (Integral definida de un conjunto de datos).

Se utiliza una placa circular para distribuir el peso que soporta un columna. Como la placa no es rígida, la presión entre placa y suelo no es constante. Los sensores colocados en la parte inferior de la placa indican la siguiente distribución radial de presión:

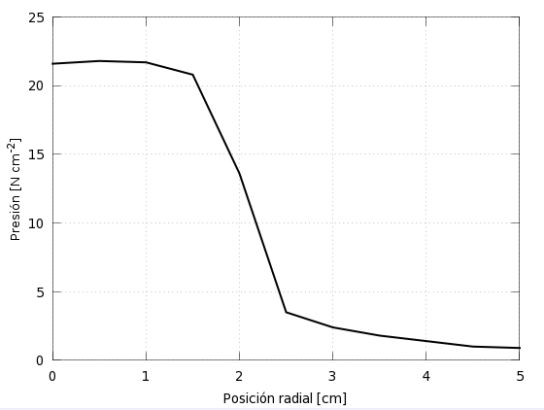
| $r$(cm) | 0,0 | 0,5 | 1,0 | 1,5 | 2,0 | 2,5 | 3,0 | 3,5 | 4,0 | 4,5 | 5,0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $P(\mathrm{N\,cm^{-2}})$ | 21,6 | 21,8 | 21,7 | 20,8 | 13,6 | 3,5 | 2,4 | 1,8 | 1,4 | 1,0 | 0,9 |

Determinar, con OCTAVE,el peso total que soporta la placa.
El peso W total que soporta la placa es:

$$W = 2\pi \int_a^b P\,dr$$

Como los datos corresponden a valores de $r$ igualmente espaciados, se puede utilizar directamente la regla del trapecio.

```
>> r = [0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5]
>> P = [21.6 21.8 21.7 20.8 13.6 3.5 2.4 1.8
       1.4 1 0.9]
>> Integral=2*pi*trapz(r,P)
```




**Solución OCTAVE:**
**Integral = 311.80 N**

### 4.5.2. Integral definida de una función

Para calcular el valor de la integral definida:

$$\int_a^b f(x)\,dx$$

se puede usar la orden:

>>**v = quad(fun,a,b)**

- **fun** es una definición de la función que calcula el integrando $f(x)$. Debe responder a la forma: **[y]=fun(x)** y su código debe estar vectorizado, esto es, debe poder admitir como argumento $\boldsymbol{x}$ un vector y devolver un vector.
- **a,b** son los límites de integración.

**Ejemplo 42** (Integral definida de una función)**.**

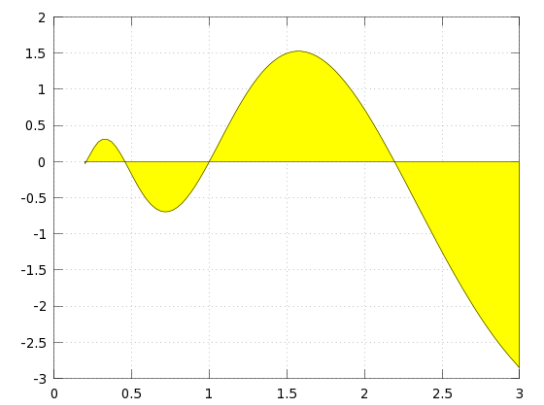
Calcular, con OCTAVE, el valor de la integral:

$$\int_{0,2}^{3} x\,\sin(4\cdot\ln(x))dx$$

```
>> fun = @(x) x.*sin(4*log(x));
>> Integral = quad(fun,0.2,3)
```

**Solución OCTAVE:**

**Integral = -0.28367**



**Ejemplo 43** (Trabajo mecánico).

Un depósito hemiesférico de 20 [m] de radio está lleno de agua hasta una profundidad de 15 [m]. Determinar, con OCTAVE, la energía necesaria para bombear toda el agua hasta la parte superior del depósito.

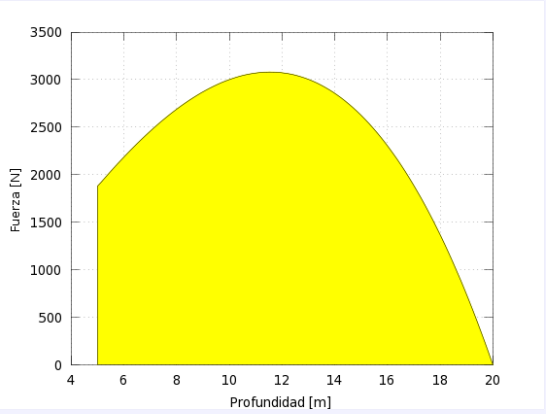
La energía necesaria para bombear el líquido desde un depósito esférico es:

$$E = (\text{Fuerza}) \cdot (\text{distancia}) = \pi \cdot \rho \cdot g \cdot \int_{h_o}^{h_f} \left(R^2 - y^2\right) \cdot y \; dy$$

```
>> R=20;
>> ho=5;
>> hf=20;
>> g=9.81;
>> rho=1000;
>> fun = @(y)(R^2−y.^2).*y ;
>> Energia = pi*rho*g*quad(fun,ho,hf);
```

**Solución OCTAVE:**

**Energía necesaria para bombear el agua, E = 1083.48 [MJ]**



### 4.5.3. Integrales dobles

Para calcular integrales dobles

$$w = \int_a^b \int_c^d f(x,y)\,dx\,dy$$

se puede usar la función **dblquad**:

>>**w = dblquad(fun,a,b,c,d)**

- La función **fun** debe responder a la forma **[v]=fun(x,y)** y admitir un vector como argumento $x$ (y devolver un vector de su misma dimensión).

**Ejemplo 44** (Integral doble de una función)**.**

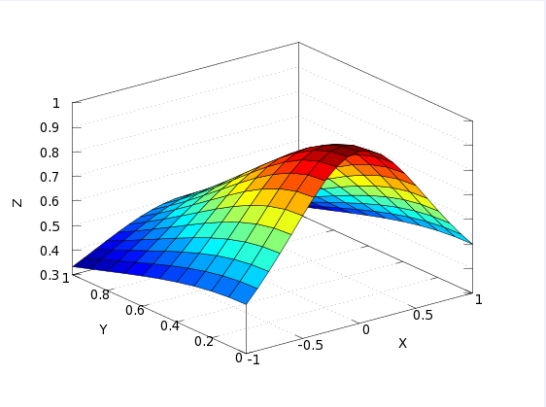
Calcular, con OCTAVE, el valor de la integral doble:

$$\int_{-1}^{1}\int_{0}^{1} \frac{1}{1+x^2+y^2} dydx$$

```
>> fun = @(x,y) 1./(1+x.^2+y^2);
>> Integral = dblquad(fun,−1,1,0,1)
```

**Solución OCTAVE:**

**Integral = 1.2790**



**Ejemplo 45** (Fuerzas distribuidas).

La presión sobre una superficie se distribuye mediante la siguiente función:

$$p(x,y) = 1000 + 230x - 210x^2 + 120y - 70y^2$$

Los valores x e y van de 0 a 1 [m]. La presión se expresa en [Pa]. Determinar, con OCTAVE, la magnitud y la ubicación de la fuerza resultante .

La magnitud de la fuerza resultante se obtiene por integración de la siguiente función:

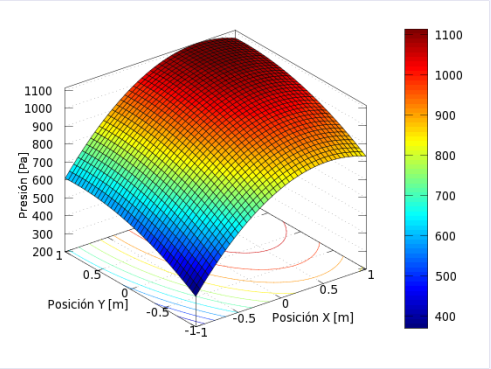
$$F_R = \int_0^1 \int_0^1 p(x,y)dydx = \int_0^1 \int_0^1 \left(1000 + 230x - 210x^2 + 120y - 70y^2\right) dydx$$

La ubicación en la que actúa la fuerza resultante se determinará mediante el cálculo del centroide de volumen:

$$\bar{x} = \frac{M_x}{F_R} = \frac{\int_0^1 \int_0^1 x \cdot p(x,y)dydx}{F_R}$$

$$\bar{y} = \frac{M_y}{F_R} = \frac{\int_0^1 \int_0^1 y \cdot p(x,y)dydx}{F_R}$$

```
>> Pres= @(x,y) 1000 + 230 .*x − 210 .*x.^2 + 120 .* y −
       70 .* y.^2;
>> MoX= @(x,y) x .* (1000 + 230 .*x − 210 .*x.^2 + 120
       .* y − 70 .* y.^2);
>> MoY= @(x,y) y .* (1000 + 230 .*x − 210 .*x.^2 + 120
       .* y − 70 .* y.^2);
>> F_R = dblquad(Pres, 0, 1, 0, 1);
>> x_loc = dblquad(MoX, 0, 1, 0, 1) ./ F_R;
>> y_loc = dblquad(MoY, 0, 1, 0, 1) ./ F_R;
```




**Solución OCTAVE:**

**La magnitud de la fuerza resultante es: FR = 1082 [N]**

**La ubicación de la fuerza resultante es: CF= (0.502, 0.504) [m]**

CAPÍTULO 5

# RESOLUCIÓN NUMÉRICA DE ECUACIONES DIFERENCIALES ORDINARIAS

## 5.1. Problemas de valor inicial para ecuaciones diferenciales ordinarias

La orden **lsode()** resuelve sistemas de ecuaciones diferenciales de primer orden, $\dot{\vec{r}} = \vec{f}(\vec{r}, t)$, con condiciones iniciales $\vec{r}(t_0) = \vec{r}_0$

>> **[x, info , msg ] = lsode (fcn , x0 , t, tcrit )**

- **fcn**: Nombre de la función que calcula las $\vec{f}(\vec{r}, t)$.
- **x0**: Condiciones iniciales $\vec{r}(t_0) = \vec{r}_0$.
- **t**: Vector con los valores de $t$ en los que debe evaluarse la función integrada.
- **tcrit**: Puntos singulares que deben ser evitados (opcional).
- **x**: Matriz de resultados.
- **info**: Condición de terminación opcional (2 si todo ha ido bien).
- **msg**: Mensaje de terminación opcional.

### 5.1.1. Ejemplo: EDO lineal. LEY DE ENFRIAMIENTO DE NEWTON

**Ejemplo 46 (LEY DE ENFRIAMIENTO DE NEWTON).**

**Templado de una pieza de acero.**

Una placa de acero se extrae de un horno a 600 ℃ y se sumerge en un baño de aceite a 30 ℃. Se sabe que la constante de enfriamiento de la pieza es 0.023. Determinar, con OCTAVE, la temperatura que tendrá la pieza después de 92 [s].

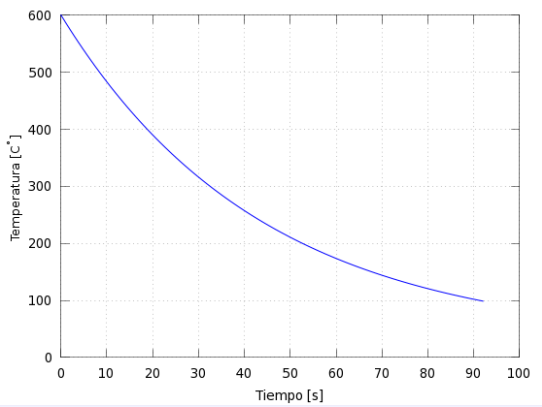
De acuerdo con la ley de enfriamiento de Newton, la rapidez de cambio de la temperatura de un cuerpo es directamente proporcional a la diferencia de temperaturas entre el cuerpo y el medio circundante. Esto es :

$$T'(t) = k\left(T_a - T(t)\right)$$
$$T(0) = 92$$

donde $k$ es una constante de proporcionalidad, $T(t)$ es la temperatura del objeto cuando $t > 0$ y $Ta$ es la temperatura ambiente, o sea; la temperatura del medio que rodea al objeto.

Como se desea saber cuánto vale la temperatura en t=92 [s], lo adecuado es resolver este problema en el intervalo [0,92].

```
>> k=0.023;
>> Ta=30;
>> T0=600;
>> t=[0:0.1:92];
>> fun= @(T,t) k*(Ta−T) ;
>> T= lsode (fun, T0, t);
>> plot(t,T,'Color','blue','linewidth', 1)
```



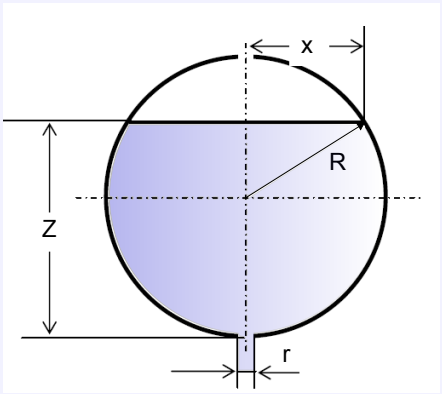

"A simple vista", se observa que la respuesta a la pregunta es 100 ℃.

### 5.1.2. Ejemplo :EDO no lineal. TEOREMA DE TORRICELLI

**Ejemplo 47 (TEOREMA DE TORRICELLI).**

**Proceso de drenaje de un depósito esférico.**

Un depósito de forma esférica con un radio de 1 [m] está parcialmente lleno de agua hasta una altura de 1,5 [m], respecto al polo inferior, donde se encuentra el punto de descarga. Dispone de un orifico circular de radio 5 [cm] en el fondo de la superficie convexa. Determinar, con OCTAVE, la altura que tendrá el agua, en el depósito esférico, después de 100 [s]. Se considera el coeficiente de descarga con valor la unidad ($Cd = 1$).




La ecuación diferencial para expresar la altura del agua en cualquier instante $t$ es:

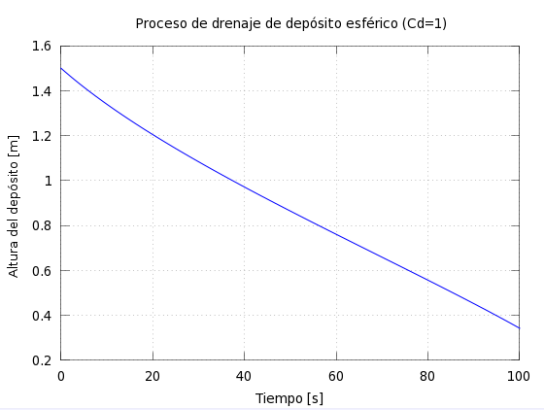
$$A(z)\frac{dz}{dt} = -Cd \cdot A_o \cdot \sqrt{2gz}$$

Si el depósito es esférico, $A(z) = \pi \cdot (2Rz - z^2)$, y la ecuación a integrar queda de la siguiente forma:

$$\frac{dz}{dt} = -Cd \cdot \frac{r^2}{(2Rz - z^2)} \cdot \sqrt{2gz}$$
$$z(0) = 1,5$$

Como se desea conocer cuánto vale la altura en t=100 [s], lo adecuado es resolver este problema en el intervalo [0,100].

```
>> R=1;
>> r=5e-2;
>> Cd=1;
>> g=9.81;
>> z0=1.5;
>> t=[0:0.1:100];
>> fun= @(z,t) -Cd*r^2*(sqrt(2*g.*z)/(2*R.*z
       -z.^2)) ;
>> zd= lsode (fun, z0, t);
>> plot(t,zd,'Color','blue','linewidth', 1);
```

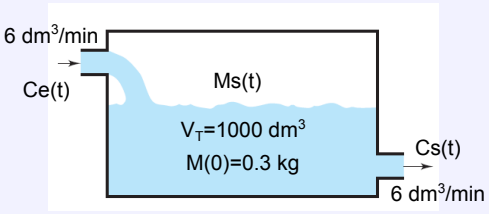



Se puede observar que la respuesta a la pregunta es 0,34 [m].

### 5.1.3. Ejemplo :EDO con funciones discontinuas. TANQUES DE MEZCLA.

**Ejemplo 48 (TANQUE DE MEZCLA.).**

Un tanque de mezcla contiene inicialmente 300 [g] de sal disuelta en 1000 $[\mathrm{dm}^3]$ de agua. En el instante t = 0 [min], se bombea al tanque una solución de 4 $[\mathrm{g}/\mathrm{dm}^3]$ de sal que entra en el tanque a 6 $[\mathrm{dm}^3/\mathrm{min}]$. En el instante t = 10 [min], la solución de entrada se ha variado a 2 $[\mathrm{g}/\mathrm{dm}^3]$ de sal, y se mantiene el flujo de 6 $[\mathrm{dm}^3/\mathrm{min}]$. Se realiza el proceso de mezcla en el tanque, y la solución uniformemente mezclada se bombea hacia afuera a razón de 6 $[\mathrm{dm}^3/\mathrm{min}]$. Representar, con OCTAVE, la concentración de sal ($[\mathrm{g}/\mathrm{dm}^3]$) en el depósito como una función del tiempo.



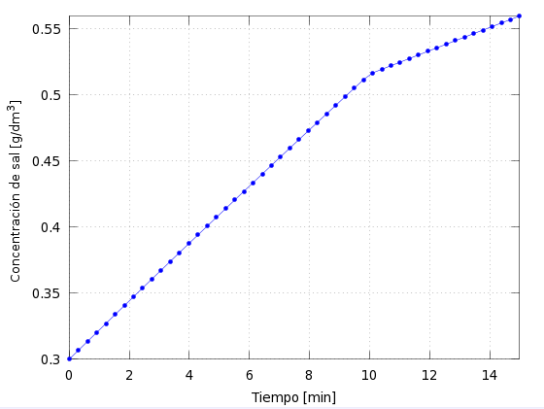

La ecuación diferencial que permite modelar el proceso de mezclado es la siguiente:

$$\frac{dMs(t)}{dt} = Q_e \cdot Cs_e(t) - Q_s \cdot \frac{Ms(t)}{V}$$

$$Cs_e(t) = \begin{cases} 0 & t \leqslant 0 \\ 4 & 0 < t \leqslant 10 \\ 2 & t > 10 \end{cases}$$

$$Ms(t=0) = 300$$

```
>> function Cs = Cs_e(t)
>> if t < 0
>> Cs = 0; % [g/dm^3]
>> elseif t > 0 && t <= 10
>> Cs = 4; % [g/dm^3]
>> else
>> Cs = 2; % [g/dm^3]
>> end
>> end
>> function dMsdt = Balance_Masa(Ms,t)
>> V = 1000; % [dm^3]
>> Q = 6; % [dm^3/min]
>> dMsdt = Q*Cs_e(t) - Q*Ms/V;
>> end
>> t=[0:0.5:15]; % [min];
>> Mo = 300; % [g]
>> V = 1000; % [dm^3]
>> M = lsode(@Balance_Masa,Mo,t);
>> plot(t,M/V,'b.-')
```



## 5.2. Problemas de valor inicial para ecuaciones diferenciales ordinarias de orden superior (n>1)

Una ecuación diferencial de orden *n* puede convertirse en *n* ecuaciones acopladas de primer orden y ser resuelta mediante **lsode()**.

### 5.2.1. Ejemplo: EDO 2º Orden. ECUACIÓN DE VAN DER POL

**Ejemplo 49 (OSCILADOR DE VAN DER POL).**

En el ámbito de los sistemas dinámicos, el oscilador de Van Der Pol es un oscilador no conservativo con amortiguamiento no lineal. El modelo dinámico de Van Der Pol se comporta en el tiempo de acuerdo con la siguiente ecuación diferencial de 2º orden:

$$y'' - \alpha(1 - y^2)y' + y = 0$$

esta ecuación es equivalente al sistema diferencial de primer orden:

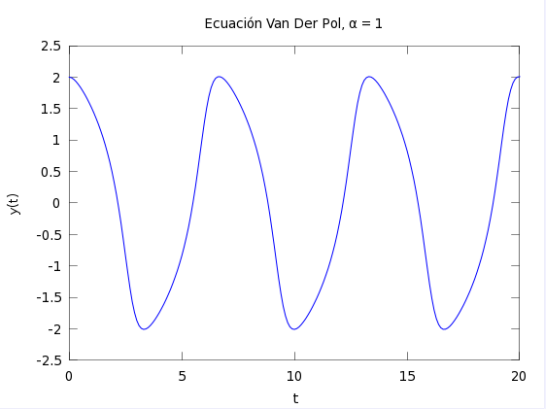
$$\begin{aligned} z_1' &= z_2 \\ z_2' &= \alpha(1 - z_1^2)z_2 - z_1 \end{aligned}$$

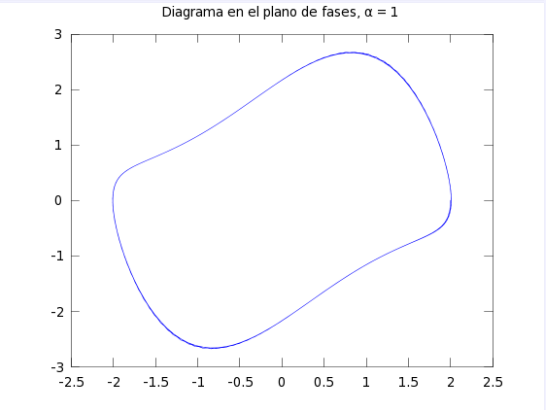
de este modo, si se conoce una solución **(z1(t),z2(t))** del sistema, entonces **y(t)=z1(t)** es una solución de la ecuación.

El problema de valores iniciales es equivalente a:

$$\begin{aligned} z_1' &= z_2 \\ z_2' &= \alpha(1 - z_1^2)z_2 - z_1 \\ z_1(0) &= y_{00} \\ z_2(0) &= y_{10} \end{aligned}$$

```
>> z0=[2; 0];
>> t=[0:0.1:20];
>> alpha = 1;
>> fun=@(z,t) [z(2) ; alpha*(1-z(1).^2).*z(2)-
       z(1) ];
>> y= lsode (fun, z0, t);
>> figure(1)
>> plot (t, y(:,1), 'b-');
>> figure(2)
>> plot(y(:,1),y(:,2))
```



## 5.3. Problema de valor inicial para sistemas de ecuaciones diferenciales ordinarias

En un sistema diferencial ordinario aparecen varias ecuaciones diferenciales y varias incógnitas. En OCTAVE, la orden **lsode()** permite resolver los sistemas de ecuaciones diferenciales de primer orden.

De este modo, el proceso para resolver un sistema de ecuaciones diferenciales en OCTAVE, consta de los siguientes pasos:

1. Definir una función que represente el sistema de ecuaciones diferenciales. El vector **xdot** contiene las ecuaciones diferenciales (en este caso *xdot(1*) y *xdot(2)*). El vector **y** contiene las variables (en este caso *y(1)* y *y(2)*).
2. Definir la condición inicial para cada una de las variables del sistema EDO. (en este caso **y0**: es el vector de condiciones iniciales).
3. Definir un vector que represente los valores que va a tomar la variable independiente $(t)$ (en este caso **t**: es el vector que especifica el intervalo de integración).
4. Utilizar el el comando **lsode** para integrar y resolver el sistema de ecuaciones diferenciales.

### 5.3.1. Ejemplo: Sistema EDO. MODELO DE LOTKA-VOLTERRA

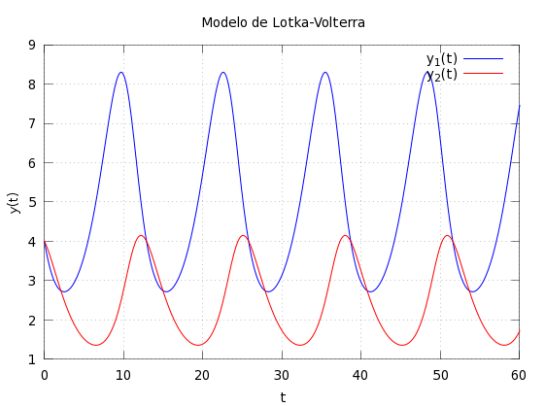
**Ejemplo 50 (MODELO DE LOTKA-VOLTERRA).**

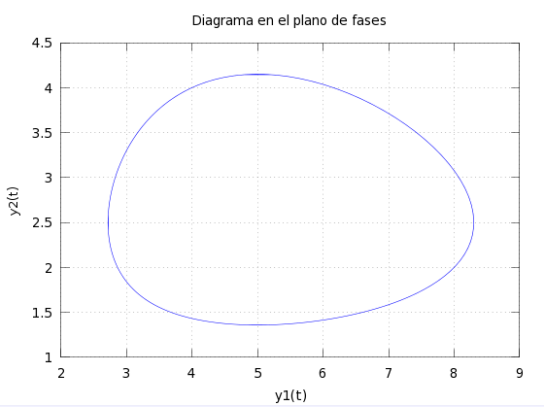
El modelo de Lotka-Volterra, también conocido como modelo de depredador-presa, ya que modeliza la situación en la que hay dos especies que conviven y una de ellas es depredadora de la otra. Si denotamos por $y_1(t)$ el número de presas en el instante $t$ y por $y_2(t)$ el número de depredadores en el instante $t$, el modelo de Lotka-Volterra establece que el número de individuos de cada especie evoluciona en el tiempo de acuerdo con el sistema diferencial:

$$y_1{}' = a\, y_1 - b\, y_1\, y_2$$
$$y_2{}' = a\, y_2 + d\, y_1\, y_2$$

en el que las constantes $a$, $b$, $c$ y $d$ varían de un caso a otro, ya que dependen de la natalidad y agresividad de cada especie. Se pueda observar, que ahora se tienen dos incógnitas y dos ecuaciones. A este sistema habrá que añadir, como en el caso de una sola ecuación, unas condiciones iniciales que indiquen cuál es la situación de partida, es decir, cuántos individuos de cada especie hay en el instante inicial.

```
>> function xdot = fcn(y,t)
>> a=0.5;b=0.2;
>> c=0.5;d=0.1;
>> xdot(1) = a*y(1)−b*y(1)*y(2);
>> xdot(2) = −c*y(2)+d*y(1)*y(2);
>> endfunction
>> y0=[4; 4];
>> t=[0:0.1:60];
>> y=lsode("fcn", y0 , t);
>> figure(1)
>> plot (t, y(:,1), 'r−', t, y(:,2), 'b');
>> legend('y_1(t)','y_2(t)')
>> figure(2)
>> plot(y(:,1),y(:,2))
```



CAPÍTULO 6

# OPTIMIZACIÓN

## 6.1. INVESTIGACIÓN OPERATIVA. Programación lineal

Para resolver problemas de programación lineal :

$$z = [\text{máx } o \text{ mín}]\ \boldsymbol{c'x}$$
$$\boldsymbol{Ax} \begin{bmatrix} \leqslant \\ = \\ \geqslant \end{bmatrix} \boldsymbol{b}$$
$$\boldsymbol{lb} \leqslant \boldsymbol{x} \leqslant \boldsymbol{ub}$$

OCTAVE dispone de la función ***glpk***, cuya utilización más sencilla es:

```
>> [xopt, fopt, status] = glpk (c, A, b, lb, ub, ctype, vartype, s)
```

donde:

- **xopt**: Valor óptimo de las variables de decisión.
- **fopt**: Valor óptimo de la función objetivo.
- **status**: Estado de la optimización ( Si se obtiene un valor de*180*, la solución es óptima).
- **c**: Vector columna que contiene los coeficientes de la función objetivo.
- **A**: Matriz que contiene los coeficientes de las restricciones.
- **b**: Vector columna que contiene el valor de lado derecho de cada restricción en la matriz de restricción.
- **lb**: Vector que contiene el límite inferior de cada una de las variables. El valor predeterminado del límite inferior para las variables es cero.
- **ub**: Vector que contiene el límite superior de cada una de las variables. El valor predeterminado del límite superior es infinito.

- **ctype**: Conjunto de caracteres que contiene el sentido de cada restricción en la matriz de restricción. Cada elemento de la matriz puede tomar uno de los siguientes valores:
  - `"F"`: Una restricción libre (sin límites) (la restricción se ignora).
  - `"U"`: Una restricción de desigualdad con un límite superior ( $A(i,:) \cdot x \leq b(i)$ ).
  - `"S"`: Una restricción de igualdad ( $A(i,:) \cdot x = b(i)$ ).
  - `"L"`: Una desigualdad con un límite inferior ( $A(i,:) \cdot x \geq b(i)$ ).
  - `"D"`: Una restricción de desigualdad con los límites superior e inferior ($A(i,:) \cdot x \geq -b(i)$ y $A(i,:) \cdot x \leq b(i)$).
- **vartype**: Vector columna que contiene los tipos de las variables.
  - `"C"`: Una variable continua.
  - `"I"`: Una variable entera.
- **s**: Sentido de la optimización. Si el sentido es **1**, el problema es de minimización. Si el sentido es **-1**, el problema es de maximización. El valor por defecto es 1.

**Ejemplo 51** (Maximización de la función objetivo).

Resolver, con OCTAVE, el siguiente problema de programación lineal:

$$Max(z) = \begin{bmatrix} 1 & -2 & -3 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$$sujeto\ a:$$

$$\begin{bmatrix} 1 & -1 & -2 & -1 \\ 2 & 0 & 1 & -4 \\ -2 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \leqslant \begin{bmatrix} 4 \\ 2 \\ 1 \end{bmatrix}, \quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \geqslant 0$$

```
>> c = [1; −2; −3; −1];
>> A = [ 1, −1, −2, −1; 2, 0, 1, −4; −2, 1, 0, 1];
>> b = [4, 2, 1];
>> lb = [];
>> ub = [];
>> Tipo_Var = "CCCC";
>> Tipo_Res = "UUU";
>> Max = −1;
>> [xmax, fmax] = glpk (c, A, b, lb, ub, Tipo_Res, Tipo_Var, Max)
```

**Ejemplo 52** (Minimización de la función objetivo).

Resolver, con OCTAVE, el siguiente problema de programación lineal:

$$Min(z) = \begin{bmatrix} 0 & -2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$sujeto\ a:$$

$$\begin{bmatrix} -1 & -2 & 0 \\ 4 & 1 & 7 \\ 2 & -3 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \geqslant \begin{bmatrix} -3 \\ -1 \\ -5 \end{bmatrix}, \quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \geqslant 0$$

```
>> c = [0; −2; 1];
>> A = [ −1, −2, 0; 4, 1, 7; 2, −3, 1];
>> b = [−3; −1; −5];
>> lb = [];
>> ub = [];
>> Tipo_Var = "CCC";
>> Tipo_Res = "LLL";
>> Min = 1;
[xmin, fmin] = glpk (c, A, b, lb, ub, Tipo_Res, Tipo_Var, Min)
```

### 6.1.1. APLICACIONES DE LA PROGRAMACIÓN LINEAL A LA LOGÍSTICA

#### MODELO DE TRANSPORTE

Este modelo consiste, en su versión más básica, en determinar las cantidades a transportar de un producto desde unos centros de producción a unos centros de demanda. Para ello, existen unas limitaciones de producción máxima en los orígenes y unas demandas mínimas en los destinos. En este sentido, la función objetivo a minimizar es la de coste total del transporte, dados los costes unitarios desde cada centro de producción a cada centro de demanda.

La formulación general del problema de transporte es la siguiente:

**función objetivo:**

$$\text{Min (z)} \quad = \quad \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} \cdot x_{ij}$$

**sujeto a:**

*1.Restricciones de disponibilidad(oferta) :*

$$\sum_{i=1}^{n} x_{ij} \leq O_i \quad \forall i \in \{1, 2, \ldots, m\}$$

*2.Restricciones de satisfacción de la demanda:*

$$\sum_{i=1}^{m} x_{ij} \geq D_j \quad \forall j \in \{1, 2, \ldots, n\}$$

*3. No negatividad de las variables:*

$$x_{ij} \geq 0$$

**Ejemplo 53** (Modelo de transporte).

Una empresa tiene 2 plantas de producción (P1 y P2) de cierto artículo que distribuye en 3 ciudades (C1,C2 y C3). Los costes unitarios de transporte, euros por unidad, la demanda semanal de cada ciudad y la producción máxima semanal de cada planta están en la siguiente tabla:

| **Ciudades** | Ciudad 1 | Ciudad 2 | Ciudad 3 | Producción [und.] |
|---|---|---|---|---|
| **Plantas** | | | | |
| Planta 1 | 40 | 60 | 70 | 550 |
| Planta 2 | 90 | 40 | 50 | 350 |
| Demanda [und.] | 400 | 300 | 100 | |

Determinar, con OCTAVE, un plan de distribución para minimizar los costes semanales de transporte.

```
% Matriz de costes
C = [ 40 60 70;
      90 40 50];
c = C'(:);
% Matriz de restricciones
A = [
1 1 1 0 0 0;
0 0 0 1 1 1;
1 0 0 1 0 0;
```

```
0 1 0 0 1 0;
0 0 1 0 0 1];
% Matriz de suministro y demanda
s = [550 350]';
d = [400 300 100]';
b=[s;d];
% Cota inferior de las variables
lb = [0 0 0 0 0 0]';
% Cota superior de las variables
ub = [];
% Tipo de variables
Tipo_Var = "CCCCCC";
% Tipo de restricciones
Tipo_Res = "UULLL";
% Tipo de problema
Min = 1;
% Sol. problema transporte
[xopt, Min_Coste] = glpk (c, A, b, lb, ub, Tipo_Res, Tipo_Var, Min );
```

### 6.1.2. APLICACIONES DE LA PROGRAMACIÓN LINEAL A PROBLEMAS DE BLENDING O MEZCLA.

En el ámbito de los sistemas de producción, los problemas de blending surgen cuando es necesario mezclar varios materiales para obtener un producto final que cumpla una serie de especificaciones. De este modo, el problema consiste en determinar cuál es la mezcla de menor coste que cumple con todos los requerimientos.

De forma genérica, el modelo se puede formular de la siguiente forma: Se dispone de $n$ materiales que pueden entrar en la mezcla, y se tienen $m$ especificaciones de calidad que debe cumplir el producto final.

$$Min(z) = \sum_{j=1}^{n} c_j \cdot x_j$$

$$\sum_{j=1}^{n} a_{ij} \cdot x_j \left[ \begin{array}{c} \leqslant \\ = \\ \geqslant \end{array} \right] b_i \; \forall i \in \{1, 2 \ldots m\}$$

$$\sum_{j=1}^{n} x_j = 1$$

## PRODUCCIÓN DE COMBUSTIBLES

**Ejemplo 54** (Modelo de mezcla de productos).

Una compañía de petróleos produce un tipo de combustible que se obtiene por mezcla de tres calidades de crudo (A,B,C). La mezcla tiene que cumplir una serie de restricciones sobre sus características. El coste de cada producto y su contribución, en términos de cada una de las características, a la calidad de la mezcla, se detallan en la siguiente tabla:

| **Producto** | Crudo A | Crudo B | Crudo C | **Requerimiento** |
|---|---|---|---|---|
| **Característica** | | | | |
| Octanaje | 120 | 100 | 74 | $\geq 94$ |
| Presión de vapor | 60 | 2,6 | 4,1 | $\leq 11$ |
| Volatilidad | 105 | 3 | 12 | $\leq 17$ |
| **Coste/Unidad** | 1000 | 2700 | 1800 | |

De forma específica, se supondrá que cada producto aporta calidad en términos de función(lineal) de su participación en la características de la mezcla. Además, es necesario producir un mínimo de 8.000 unidades de combustible y se dispone, como máximo, de 1.000 unidades de crudo A. Determinar, con OCTAVE, la mezcla que cumple todas las especificaciones y minimiza los costes.

```
% Matriz de costes
c = [1000; 2700; 1800; 0];
% Matriz de restricciones
A = [ 120, 100, 74, −94;
      60, 2.6, 4.1, −11;
      105, 3, 12, −17;
      1, 1, 1, −1;
      1 0 0 0;
      0 0 0 1 ];
% Matriz de disponibilidad recursos
b = [0; 0; 0; 0;1000;8000];
% Cota inferior de las variables
lb = [];
% Cota superior de las variables
ub = [];
% Tipo de variables
Tipo_Var = "CCCC";
% Tipo de restricciones
Tipo_Res = "LUUSUL";
% Tipo de problema (Min=1)
Min = 1;
[xmin, fopt, status ] = glpk (c, A, b, lb, ub, Tipo_Res, Tipo_Var, Min)
```

## GNU-Octave(UPM)

GNU Octave (UPM) está amparado bajo los términos de la GNU (General Public License) de la Fundación del Software Libre (Free Software Foundation) por lo tanto su código fuente se puede distribuir de manera libre. GNU Octave (UPM) se distribuyen como software libre y se pude descargar del siguiente enlace: GNU Octave(UPM).