

WUOLAH



marinamuca01

www.wuolah.com/student/marinamuca01



2723

FORMULAS-MAXIMA.pdf

Recopilación Maxima



1º Cálculo



Grado en Ingeniería Informática



Escuela Técnica Superior de Ingenierías Informática y de
Telecomunicación
Universidad de Granada

¿Quieres **Amazon Prime gratis**?
Entra por nuestro link o QR y consigue **90 días
de Prime gratis** y después **50% de descuento**.

Los recomendados
de **amazon** y **WUOLAH**



<https://amzn.to/33EbAFJ>

LISTAS

nombre:[elementos]; */*los elementos pueden ser puntos, numeros, funciones, etc.*/*
nombre[posicion]; */*para referirse a un elemento que está en dicha posición */*
makelist(regla, contador, valor_inicial, valor_final)

FUNCIONES

DEFINICION:

define (funcion(var), expresion);
funcion(var):=expresion;

GRÁFICAS

PLOT2D:

wxplot2d (expr, [x, min, max], [y, min, max]); */* no hace falta acotar el eje y */*

DRAW2D EXPLICIT → SOLO VARIABLE X

```
wxdraw2d(color=violet,  
          explicit(funcion,x,p1,p2)  
        );
```

DRAW2D IMPLICIT → VARIABLE X, Y

```
wxdraw2d(  
  line_width=5, /* grosor de línea */  
  color=dark-green, /* color de línea */  
  proportional_axes=xy, /* ejes proporcionales */  
  implicit(x + y = a,x,p1,p2,y,p3,p4)  
);
```

La ecuación a introducir debe contener las variables x, y. Ej: $x^2 + y^2 = 7$ // $\sin(x) - 4y = 0$.

PUNTOS

```
wxdraw2d(color=dark-pink,  
  point_size=2, /* tamaño de puntos */  
  point_type=7, /* tipo de puntos */  
  points([ [x1,y1],[x2,y2],[x3,y3] ]),  
  xrange=[p1,p2], /* acotar una variable entre dos puntos */  
  yrange=[p3,p4]  
);
```

CONDICIONALES Y BUCLES

if condición **then** expresion1 **else** expresion2;
for contador:valor_inicial **thru** valor_final **do** expresión;
while condición **do** expresión */* se repite mientras se cumpla condición */*
unless condición **do** expresión */* se repite a no ser que se cumpla condición */*

Formación
Online
Especializada

Clases Online
Prácticas
Becas

Ponle
nombre
a lo que
quieres ser

Jose María Girela
Bim Manager.

*/*Para mostrar resultados intermedios:*/*

print (expr_1, ..., expr_n)

disp (expr_1, expr_2, ...)

display (expr_1, expr_2, ...)

/ muestra y evalúa las expresiones, si la expresión está entre comillas, solo la muestra*/*

ECUACIONES

nombre _ecuación : expresión1 = expresión 2;

lhs (nombre _ecuación); */* miembro izquierdo de la ecuacion */*

rhs (nombre _ecuación); */* miembro derecho de la ecuacion */*

solve ([expresion/es], var);

to_poly_solve (expr, var);

find_root (funcion, var, min, max);

elipse: $x^2/a+y^2/b=1$;

circunferencia: $x^2+y^2=r^2$;

ERROR

ABSOLUTO

| valor_exacto - valor_aprox |

RELATIVO

| valor_exacto - valor_aprox | / valor_exacto

SUSTTUIR VARIABLES

subst (y, x, f(x)) */*Sustituye en f(x) x por y */*

subst ([x = y], f(x)) */*Sustituye en f(x) x por y */*

at (función (x, y),(x = num1, y = num2)) */*evalúa la función en x e y*/*

BISECCIÓN

biseccion(expr,var,ext_inf,ext_sup):=

block(

[a,b,c,

fa,fb,fc,

contador:0,

tolx: 10^{-6} ,tolfx: 10^{-6})

],

local(f),

define(f(x),subst(x,var,expr)),

a:float(min(ext_inf,ext_sup)),

b:float(max(ext_inf,ext_sup)),

c:(a+b)*0.5,

fa:f(a),

fb:f(b),

/ extremos y punto medio */*

/ valores de la función en dichos puntos */*

/ número de repeticiones */*

/ error permitido */*

```

fc:f(c),
if abs(fc)<tolfx then return([c,1,(b-a)*0.5,fc]),
if sign(fa)=sign(fb) then error("la función no cambia de signo en los extremos"),
while ((b-a)>tolx and abs(fc)>tolfx)
do(
    contador:contador+1,
    c:float((a+b)/2),
    fc:f(c),
    if abs(fc)<tolfx then return(),
    if sign(fa) = sign(fc) then (a:c,fa:fc) else (b:c,fb:fc)
),
[c,contador,(b-a)*0.5,f(c)]
);

```

TRISECCIÓN

```

triseccion(exp,var,ext_inf,ext_sup):=block(
    [a:ext_inf,b:ext_sup,c,d,solucion,numer:true],
    local(f(x),subst(x,var,expr)),
    while (b-a)/3>=10^(-5) do(
        c:a+(b-a)/3,
        if f(c)=0 then (solucion:c,return()),
        d:a+2*(b-a)/3,
        if f(d)=0 then (solucion:d,return()),
        /*miro en cual de los dos puntos vale menos*/
        if f(a)*f(c)<0 then b:c else
            if f(c)*f(d)<0 then (a:c,b:d) else a:d,
        solucion:(a+b)/2
    ),
    solucion
);

```

MÉTODO NEWTON-RAPHSON

```

nr(exp,var,ini):=block(
    [x0:ini,x1,dfx0,j,tol:10^(-10)],
    local(f,df),
    define(f(x),subst(x,var,expr)),
    define(df(x),diff(f(x),x)),
    for i:1 thru 15 do (
        j:i,
        dfx0:df(x0),
        if abs(df(x0))<10^(-5) then
            x1:x0-f(x0)/dfx0,
            if abs(x0-x1)<tol then return(),
            x0:x1
        ),
        if j=15 then error("elige otro valor inicial") else x1
    )$
);

```

VERSIÓN CON ERRORES ABSOLUTO Y RELATIVO

```
nr1(expr,var,ini,errab,errel):=block(
  [x0:ini,x1:dfx0,j,tol:10^(-10)],
  local(f,df),
  define(f(x),subst(x,var,expr)),
  define(df(x),diff(f(x),x)),
  for i:1 thru 15 do (
    j:i,
    dfx0:df(x0),
    if abs(df(x0))<10^(-5) then
      x1:x0-f(x0)/dfx0,
      if abs(x0-x1)<errab then return(),
      if abs(x0-x1)/abs(x1)<errel then return(),
      x0:x1
    ),
    if j=15 then error("elige otro valor inicial") else x1
  );
```

MÉTODO REGULA FALSI

```
define(f(x),x^3-5)$          /* función */
a:0.0;b:4.0$                 /* extremos a y b del intervalo */
err_a:10^(-3)$               /* error absoluto */
contador:0$
maxiter:30$
while abs(b-a)/2 > err_a and contador < maxiter
do(
  contador:contador+1,
  c:(f(b)*a-f(a)*b)/(f(b)-f(a)),
  if f(c)=0 then return(c),
  if f(a)*f(c)<0 then b:c else a:c
);
[c,contador];
```

MÉTODO DE LA SECANTE

```
secante(expr,var,ini,fin,errab,errel):=block(
  [x0:ini,x1:fin,x2,j,control:10^(-5)],
  local(f),
  define(f(x),subst(x,var,expr)),
  for i:1 thru 15 do (
    j:i,
    if abs(x0-x1)<erra then return(),
    if abs(x1-x0)< errel*abs(x1) then return(),
    if abs(f(x0)-f(x1))<control then return(),
    x2:(x0*f(x1)-x1*f(x0))/(f(x1)-f(x0)),
    x0:x1,x1:x2
  ),
  if j=15 then error("elige otros valores iniciales") else [x2,j]
);
```