SLUD 2006

Resolver problemas con Máxima

In memoriam William Schelter

Robert Dodier
Proyecto Máxima

Libero este documento por el
GNU General Public License version 2

Características generales de Máxima

Toda cosa es una expresión (casi todas)

Máxima = colección de funciones para trasformar expresiones

Soluciones perezosas y la actitud “laissez faire”

Distinguir entre evaluación y simplificación

Máxima como calculador perezoso

```
(%i1) F : m * a;
(%o1)                                   a m
(%i2) a : 9.81;
(%o2)                                  9.81
(%i3) F;
(%o3)                                   a m
(%i4) ''F;
(%o4)                                 9.81 m
(%i5) S : sum (g(i, x), i, 0, 6);
(%o5) g(6, x) + g(5, x) + g(4, x) + g(3, x) + g(2, x) + g(1, x) + g(0, x)
(%i6) g(i, x) := x^i / i!;
                                                i
                                               x
(%o6)                              g(i, x) := --
                                               i!
```

```
(%i7) ''S;
                          6     5     4    3    2
                         x     x     x    x    x
(%o7)                    --- + --- + -- + -- + -- + x + 1
                         720   120   24   6    2
(%i8) x : 1;
(%o8)                                1
(%i9) ''S;
                                   1957
(%o9)                              ----
                                   720
(%i10) ''%, numer;
(%o10)                       2.718055555555555
(%i11) %e, numer;
(%o11)                       2.718281828459045
```

Definición de funciones

```
(%i1) F (m, a) := m * a;
(%o1)                          F(m, a) := m a
(%i2) F (100, 9.81);
(%o2)                               981.0
(%i3) F (100, g);
(%o3)                               100 g
(%i4) apply (F, [100, g]);
(%o4)                               100 g
(%i5) L1 : [100, 200, 300];
(%o5)                          [100, 200, 300]
(%i6) L2 : [a, b, c];
(%o6)                             [a, b, c]
(%i7) map (F, L1, L2);
(%o7)                      [100 a, 200 b, 300 c]
```

Función sin nombre: **lambda** — Máxima reconoce a funciones sin nombre en algunos contextos en los que se espere una función

```
(%i1) L1 : [100, 200, 300];
(%o1)                            [100, 200, 300]
(%i2) L2 : [a, b, c];
(%o2)                               [a, b, c]
(%i3) map (lambda ([m, a], m * a), L1, L2);
(%o3)                         [100 a, 200 b, 300 c]
(%i4) apply (lambda ([m, a], m * a), [1234, x + u]);
(%o4)                             1234 (x + u)
```

Gráficos

Máxima puede hacer gráficos de 2 o 3 dimensiones

plot2d($expr, [x, a, b]$) — hacer un gráfico de *expr* de variable x entre limites a y b

plot2d([**discrete**$, x, y$]) — hacer un gráfico de datos discretos, en las listas x y y

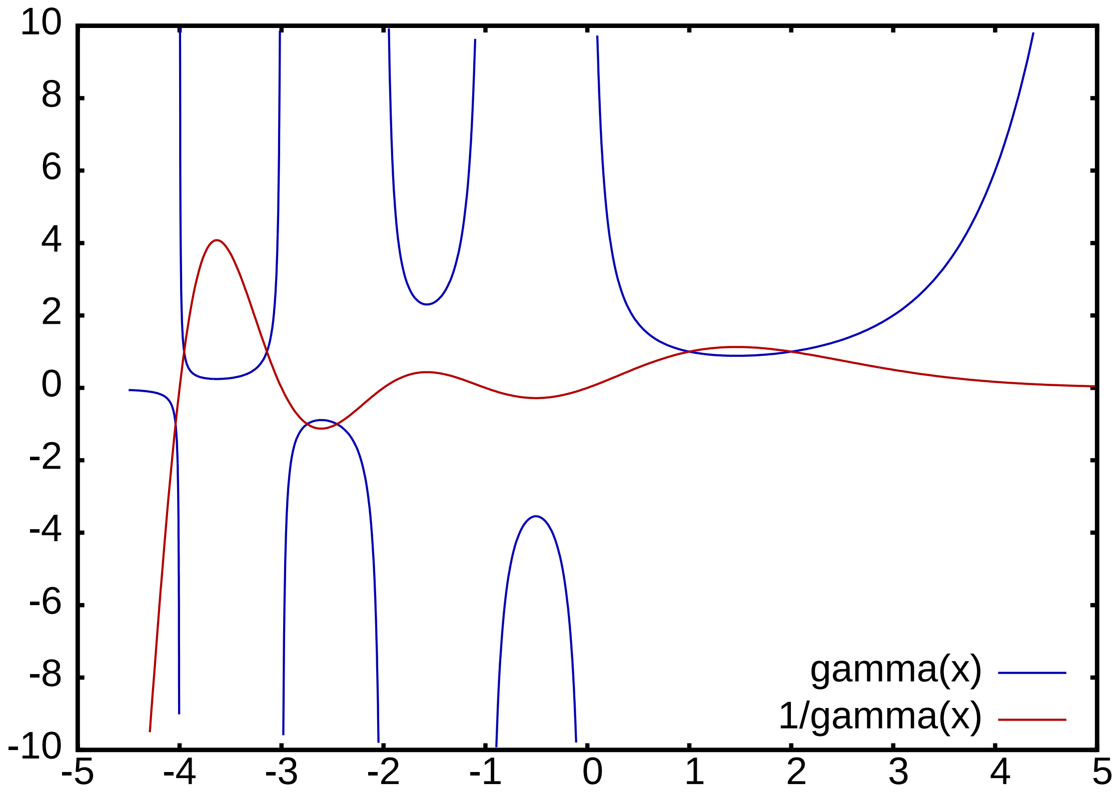
plot3d — gráficos de 3 dimensiones

Se puede eligir entre formatos — Gnuplot o Openmath (Tcl/Tk)

Para hacer gráficos, hay varias opciones arcanas. Se refiere a la documentación.

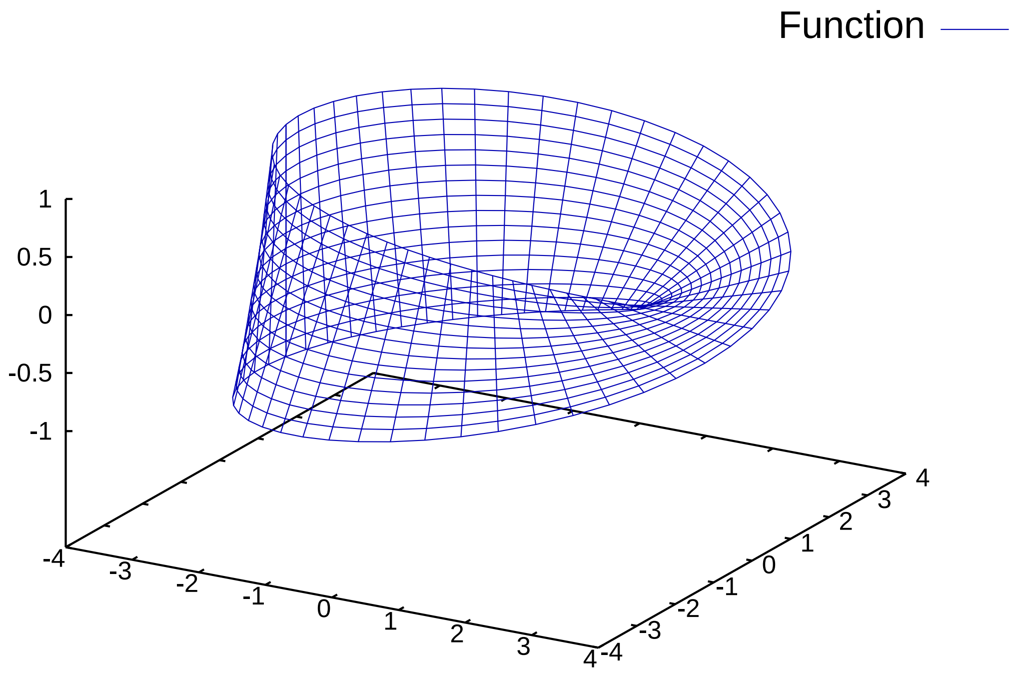
Ejemplo de gráfico 2-dimensional

```
plot2d ([gamma(x), 1/gamma(x)], [x, -4.5, 5], [y, -10, 10]);
```



Ejemplo de gráfico 3-dimensional

```
plot3d ([cos(x)*(3 + y*cos(x/2)), sin(x)*(3 + y*cos(x/2)),
   y*sin(x/2)], [x, -%pi, %pi], [y, -1, 1], ['grid, 50, 15]);
```



Números

Números enteros. Precisión no fija. Computación rapida con funciones especializadas GMP

```
(%i1) 2^100;
(%o1)            1267650600228229401496703205376
(%i2) 100!;
(%o2) 9332621544394415268169923885626670049071596826438162146859\
2963895217599993229915608941463976156518286253697920827223758251\
185210916864000000000000000000000000
(%i3) is (100! > 2^100);
(%o3)                         true
(%i4) primep (2^100 + 1);
(%o4)                         false
(%i5) ifactors (2^100 + 1);
(%o5) [[17, 1], [401, 1], [61681, 1], [340801, 1], [2787601, 1],
                                                [3173389601, 1]]
```

Números racionales. Raciones de números enteros

```
(%i1) 1/4 - 1/5;
                                         1
(%o1)                                    --
                                         20
(%i2) 2^100 / 100!;
(%o2) 8/588971222367687651371627846346807888288472382883312574253\
249804256440585603406374176100610302040933304083276457607746124\
267578125
(%i3) 17 * 2^100 / 100! - 29 * 2^101 / 101!;
(%o3) 8/35856596419009314519912243810143216827688794859080512356\
58723943936136175162389619758056759524179280513104938048124313354\
44921875
```

Números de coma flotante y precisión fija (IEEE 754). Máxima usa solamente la precisión doble (64 bits)

```
(%i1) 0.25 - 1/5;
(%o1)                         .04999999999999999
(%i2) float (1/4 - 1/5);
(%o2)                                0.05
(%i3) sin (1);
(%o3)                               sin(1)
(%i4) sin (1.0);
(%o4)                          .8414709848078965
```

Números de coma flotante y precisión no fija ("bigfloat")

```
(%i1) fpprec : 50;
(%o1)                                 50
(%i2) fpprintprec : 50;
(%o2)                                 50
(%i3) bfloat (%pi);
(%o3)       3.1415926535897932384626433832795028841971693993751b0
(%i4) sin (1b0);
(%o4)      8.4147098480789650665250232163029899962256306079837b-1
```

Números complejos. Máxima no reconoce como tipo distinto, sino como expresión de forma $a + b \times i$

```
(%i1) c : a + b * %i;
(%o1)                              %i b + a
(%i2) realpart (c);
(%o2)                                  a
(%i3) imagpart (c);
(%o3)                                  b
(%i4) exp (c);
                                    %i b + a
(%o4)                             %e
(%i5) demoivre (exp (c));
                               a
(%o5)                        %e  (%i sin(b) + cos(b))
```

Polinomios

```
(%i1) P : (x - 1) * (x - 2) * (x + 3);
(%o1)                    (x - 2) (x - 1) (x + 3)
(%i2) Q : expand (P);
                                3
(%o2)                          x  - 7 x + 6
(%i3) factor (Q);
(%o3)                    (x - 2) (x - 1) (x + 3)
(%i4) solve (Q = 0, x);
(%o4)                    [x = 1, x = 2, x = - 3]
(%i5) subst (first (%), Q);
(%o5)                                0
```

```
(%i6) solve (Q = 1, x);
                                - 3/2
              sqrt(3) %i   1   3      sqrt(697) %i   5 1/3
(%o6) [x = (- ---------- - -) (------------------- - -)
                  2        2             2            2
            sqrt(3) %i   1
         7 (---------- - -)
                2        2
 + ------------------------------, x =
       - 3/2
      3      sqrt(697) %i   5 1/3
   3 (------------------- - -)
               2            2
                   - 3/2
 sqrt(3) %i   1   3      sqrt(697) %i   5 1/3
(---------- - -) (------------------- - -)
     2        2             2            2
             sqrt(3) %i   1
        7 (- ---------- - -)
                 2        2
```

```
 + ------------------------------, x = 
         - 3/2
        3       sqrt(697) %i    5 1/3
     3 (------------------- - -)
                  2             2
   - 3/2
  3       sqrt(697) %i    5 1/3                    7
(------------------- - -)     + ------------------------------]
           2             2             - 3/2
                                        3       sqrt(697) %i    5 1/3
                                     3 (------------------- - -)
                                                  2             2
(%i7) allroots (Q - 1);
(%o7) [x = .7828156786641545, x = 2.166012679457936, 
                                            x = - 2.948828358122091]
```

Funciones matemáticas

Máxima tiene una gran collección de funciones básicas. Se muestran algunos debajo. También tiene paquetes de polinómios ortogonales y funciones hipergeometricas y muchas otras funciones.

```
(%i1) sin (u);
(%o1)                            sin(u)
(%i2) sin (%pi * u - v / w);
                                   v
(%o2)                        - sin(- - %pi u)
                                   w
(%i3) sin (0);
(%o3)                              0
(%i4) sin (%pi);
(%o4)                              0
(%i5) sin (1);
(%o5)                            sin(1)
```

```
(%i6) sin (1/2);
                                        1
(%o6)                               sin(-)
                                        2
(%i7) sin (0.5);
(%o7)                           0.479425538604203
(%i8) sin (1) + cos (1);
(%o8)                            sin(1) + cos(1)
(%i9) ''%, numer;
(%o9)                           1.381773290676036
```

Unos ejemplos de gráficos que se pueden hacer.

```
plot2d ([sin, cos, tan], [x, -0.5, 7], [y, -3, 3]);
plot2d ([sinh, cosh, tanh], [x, -2, 2], [y, -10, 10]);
plot2d ([bessel_j (0, x), bessel_j (1, x)], [x, -0.5, 10], [y, -3, 3]);
```

Listas, conjuntos, y matrices

map, **apply**

```
(%i1) L : [1, 1, 2, 2, a, a, b, b];
(%o1)                      [1, 1, 2, 2, a, a, b, b]
(%i2) length (L);
(%o2)                                  8
(%i3) first (L);
(%o3)                                  1
(%i4) last (L);
(%o4)                                  b
(%i5) [length (L), first (L), last (L)];
(%o5)                               [8, 1, b]
(%i6) makelist (sin (i), i, 1, 5);
(%o6)              [sin(1), sin(2), sin(3), sin(4), sin(5)]
```

```
(%i1) L : [a, b, x, 0, 1, 2];
(%o1)                       [a, b, x, 0, 1, 2]
(%i2) map (F, L);
(%o2)             [F(a), F(b), F(x), F(0), F(1), F(2)]
(%i3) map (sin, L);
(%o3)          [sin(a), sin(b), sin(x), 0, sin(1), sin(2)]
(%i4) map (lambda ([e], e + 1), L);
(%o4)                [a + 1, b + 1, x + 1, 1, 2, 3]
(%i5) apply (concat, L);
(%o5)                             abx012
(%i6) apply (max, L);
(%o6)                        max(x, b, a, 2)
```

```
(%i1) C : {1, 1, 2, 2, a, a, b, b};
(%o1)                       {1, 2, a, b}
(%i2) cardinality (C);
(%o2)                             4
(%i3) subsetp ({1, 2, a}, C);
(%o3)                           true
(%i4) powerset (C);
(%o4) {{}, {1}, {1, 2}, {1, 2, a}, {1, 2, a, b}, {1, 2, b},
{1, a}, {1, a, b}, {1, b}, {2}, {2, a}, {2, a, b}, {2, b}, {a},
{a, b}, {b}}
(%i5) permutations (C);
(%o5) {[1, 2, a, b], [1, 2, b, a], [1, a, 2, b], [1, a, b, 2],
[1, b, 2, a], [1, b, a, 2], [2, 1, a, b], [2, 1, b, a],
[2, a, 1, b], [2, a, b, 1], [2, b, 1, a], [2, b, a, 1],
[a, 1, 2, b], [a, 1, b, 2], [a, 2, 1, b], [a, 2, b, 1],
[a, b, 1, 2], [a, b, 2, 1], [b, 1, 2, a], [b, 1, a, 2],
[b, 2, 1, a], [b, 2, a, 1], [b, a, 1, 2], [b, a, 2, 1]}
```

```
(%i1) M : matrix ([3, -1/5], [-1/5, %pi]);
                          [          1 ]
                          [  3     - - ]
                          [          5 ]
(%o1)                     [            ]
                          [   1        ]
                          [ - -    %pi ]
                          [   5        ]
(%i2) transpose (M);
                          [          1 ]
                          [  3     - - ]
                          [          5 ]
(%o2)                     [            ]
                          [   1        ]
                          [ - -    %pi ]
                          [   5        ]
(%i3) N : transpose (M) . M;
                    [    226        %pi   3 ]
                    [    ---      - --- - - ]
                    [    25          5    5 ]
```

```
(%o3)                  [                      ]
                       [   %pi   3      2   1 ]
                       [ - --- - -  %pi   + -- ]
                       [    5    5          25 ]
(%i4) invert (N);
                  [                   2   1               ]
                  [                %pi  + --              ]
                  [                       25              ]
                  [ ------------------------------------- ]
                  [          2   1                        ]
                  [ 226 (%pi  + --)                       ]
                  [              25        %pi   3   %pi   3 ]
                  [ -------------- + (- --- - -) (--- + -) ]
                  [       25               5    5    5    5 ]
(%o4)  Col 1 =    [                                       ]
                  [                 %pi   3               ]
                  [                 --- + -               ]
                  [                  5    5               ]
                  [ ------------------------------------- ]
                  [          2   1                        ]
```

```
      [ 226 (%pi  + --)                         ]
      [             25       %pi   3   %pi   3  ]
      [ --------------- + (- --- - -) (--- + -) ]
      [       25              5    5    5    5  ]
        [                   %pi   3                    ]
        [                   --- + -                    ]
        [                    5    5                    ]
        [   ---------------------------------------    ]
        [           2   1                              ]
        [   226 (%pi  + --)                            ]
        [               25       %pi   3   %pi   3     ]
        [   --------------- + (- --- - -) (--- + -)    ]
Col 2 = [         25              5    5    5    5     ]
        [                                              ]
        [                     226                      ]
        [ -------------------------------------------- ]
        [             2   1                            ]
        [     226 (%pi  + --)                          ]
        [                 25       %pi   3   %pi   3   ]
        [ 25 (--------------- + (- --- - -) (--- + -)) ]
```

```
                  [            25                5     5     5     5   ]
(%i5) ratsimp (N);
                    [     226          %pi + 3  ]
                    [     ---        - -------  ]
                    [     25              5     ]
(%o5)               [                           ]
                    [                     2     ]
                    [    %pi + 3  25 %pi   + 1 ]
                    [  - -------  ----------- ]
                    [       5          25      ]
(%i6) vcm : eigenvalues (M);
                  2
          sqrt(25 %pi  - 150 %pi + 229) - 5 %pi - 15
(%o6) [[- ------------------------------------------,
                              10
                     2
             sqrt(25 %pi  - 150 %pi + 229) + 5 %pi + 15
             ------------------------------------------], [1, 1]]
                                 10
(%i7) vcn : eigenvalues (N);
```

```
(%o7)
                           2                          2
    (5 %pi + 15) sqrt(25 %pi  - 150 %pi + 229) - 25 %pi  - 227
[[- ---------------------------------------------------------,
                                50
                       2                          2
(5 %pi + 15) sqrt(25 %pi  - 150 %pi + 229) + 25 %pi  + 227
---------------------------------------------------------],
                            50
[1, 1]]
(%i8) vcm, numer;
(%o8)   [[2.858635727513009, 3.282956926076785], [1, 1]]
(%i9) vcm^2, numer;
(%o9)   [[8.171798222613827, 10.77780617847553], [1, 1]]
(%i10) vcn, numer;
(%o10)  [[8.171798222613827, 10.77780617847553], [1, 1]]
```

Integrales símbolicos

Máxima tiene una implementación del algoritmo Risch y otros para integrales símbolicos

```
(%i1) I : exp(x) * sin(2 * x) * cos(3 * x);
                                x
(%o1)                         %e  sin(2 x) cos(3 x)
(%i2) integrate (I, x);
             x                              x
           %e  (sin(5 x) - 5 cos(5 x))   %e  (sin(x) - cos(x))
(%o2)      --------------------------- - ---------------------
                       52                          4
(%i3) integrate (I, x, 0, 1);
          %e sin(5)   5 %e cos(5)   %e sin(1)   %e cos(1)   2
(%o3)     --------- - ----------- - --------- + --------- - --
             52           52            4           4       13
```

Integrales númericos

También podemos realizar aproximaciones numéricas por funciones de QUADPACK (originalmente de Fortran)

```
(%i1) I : exp(x) * sin(2 * x) * cos(3 * x);
                                x
(%o1)                         %e  sin(2 x) cos(3 x)
(%i2) quad_qags (I, x, 0, 1);
(%o2)         [- .4827806301140743, 8.870075205972286E-15, 21, 0]
(%i3) ''(integrate (I, x, 0, 1)), numer;
(%o3)                         - .4827806201250382
```

Ecuaciones diferenciales

ode2 para resolver una ecuación diferencial

ic1 y **ic2** para problemas de valores inicial

bc2 para problemas de valores de frontera

```
(%i1) x^2 * 'diff(y, x) + 3 * y * x = sin(x)/x;
                           2 dy            sin(x)
(%o1)                     x  -- + 3 x y = ------
                             dx              x
(%i2) ode2 (%, y, x);
                                 %c - cos(x)
(%o2)                        y = -----------
                                      3
                                     x
(%i3) ic1 (%, x = %pi, y = 0);
                                  cos(x) + 1
(%o3)                       y = - ----------
                                       3
                                      x
```

```
(%i1) 'diff(y, x, 2) + y * 'diff(y, x)^3 = 0;
                         2
                        d y        dy 3
(%o1)                   --- + y (--)  = 0
                          2        dx
                        dx
(%i2) soln : ode2 (%, y, x);
                     3
                    y  + 6 %k1 y
(%o2)               ------------ = x + %k2
                         6
```

```
(%i1) 'diff(y, x, 2) + y * 'diff(y, x)^3 = 0;
                         2
                        d y       dy 3
(%o1)                   --- + y (--)  = 0
                          2       dx
                        dx
(%i2) soln : ode2 (%, y, x);
                     3
                    y  + 6 %k1 y
(%o2)               ------------ = x + %k2
                         6
(%i3) ratsimp (ic2 (soln, x = 0, y = 0,
        'diff(y, x) = 2));
                            3
                         2 y  - 3 y
(%o3)                  - ---------- = x
                             6
```

```
(%i4) bc2 (soln, x = 0, y = 1, x = 1, y = 3);
                                 3
                                y  - 10 y       3
(%o4)                           --------- = x - -
                                    6           2
```

Álgebra lineal

Por lo largo de la historia de Máxima, habían sido varios paquetes de funciones de álgebra lineal

Lo más reciente se llama **linearalgebra**, que tiene funciones de LU descomposición, computación del espacio nulo y su complemento, y otros

```
(%i1) M : matrix ([a, 1], [2, b]);
                                   [ a  1 ]
(%o1)                              [      ]
                                   [ 2  b ]
(%i2) load (linearalgebra);
(%o2)   /usr/share/maxima/5.10.0/share/linearalgebra/linearalgebra.mac
```

```
(%i3) LU : lu_factor (M);
                         [ a    1   ]
                         [          ]
(%o3)                   [[ 2      2 ], [1, 2], generalring]
                         [ -  b - - ]
                         [ a      a ]
(%i4) get_lu_factors (LU);
                                   [ 1  0 ]  [ a    1   ]
                         [ 1  0 ]  [      ]  [          ]
(%o4)                   [[      ], [ 2    ], [        2 ]]
                         [ 0  1 ]  [ -  1 ]  [ 0  b - - ]
                                   [ a    ]  [        a ]
(%i5) lu_backsub (LU, matrix ([3], [4]));
                                   [         6 ]
                                   [     4 - - ]
                                   [         a ]
                                   [ 3 - ----- ]
                                   [         2 ]
                                   [     b - - ]
                                   [         a ]
```

```
                                        [ --------- ]
(%o5)                                   [     a     ]
                                        [           ]
                                        [       6   ]
                                        [   4 - -   ]
                                        [       a   ]
                                        [   -----   ]
                                        [       2   ]
                                        [   b - -   ]
                                        [       a   ]
(%i6) ratsimp (%);
                                         [ 3 b - 4 ]
                                         [ ------- ]
                                         [ a b - 2 ]
(%o6)                                    [         ]
                                         [ 4 a - 6 ]
                                         [ ------- ]
                                         [ a b - 2 ]
```

```
(%i2) M2 : matrix ([x, - 1/2], [1/3, y]);
                                       [      1 ]
                                       [ x  - - ]
                                       [      2 ]
(%o2)                                  [        ]
                                       [ 1      ]
                                       [ -    y ]
                                       [ 3      ]
(%i3) nullspace (M2);
                        1
Proviso:  {x # 0, y + --- # 0}
                       6 x
(%o3)                                   span()
```

```
(%i4) M3 : matrix ([x, - 1/2], [- 2 * x, 1]);
                                  [           1 ]
                                  [   x     - - ]
(%o4)                             [           2 ]
                                  [             ]
                                  [ - 2 x   1   ]
(%i5) nullspace (M3);
Proviso:  {x # 0}
                                        [  1  ]
(%o5)                              span([     ])
                                        [ 2 x ]
(%i6) columnspace (M3);
Proviso:  {x # 0}
                                       [   x   ]
(%o6)                             span([       ])
                                       [ - 2 x ]
```

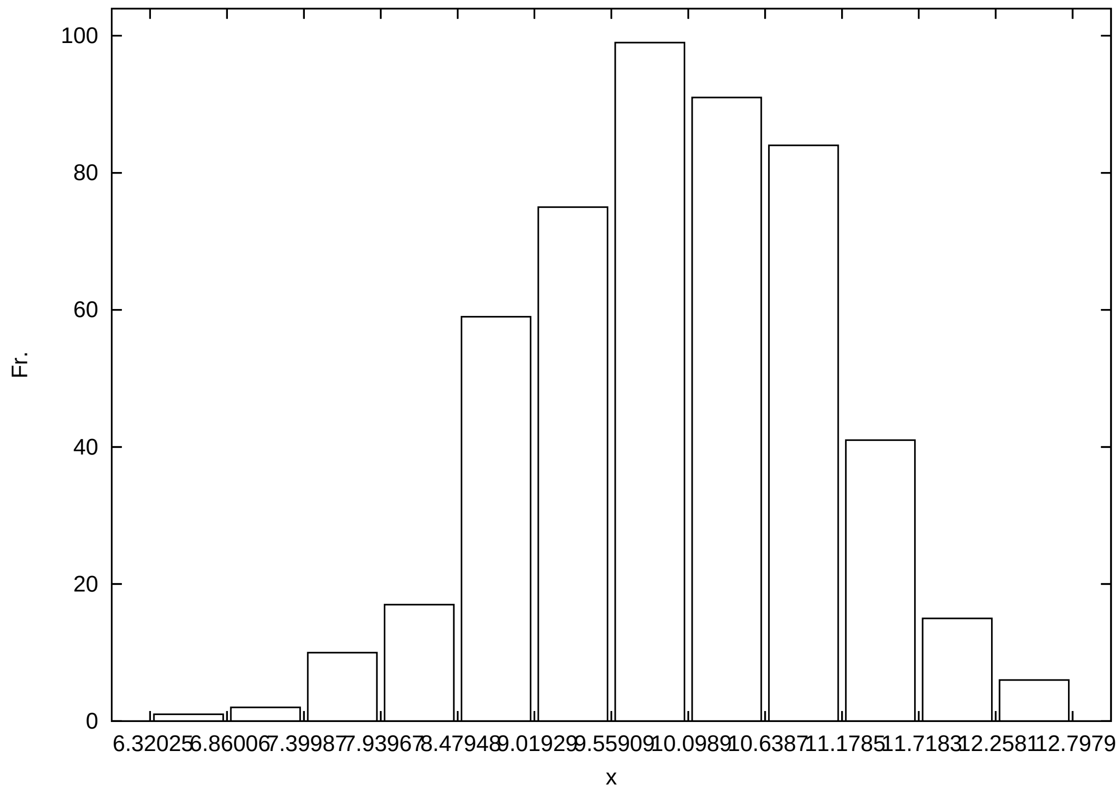
Estadística y tratamiento de datos

Para funciones de probabilidad y estadística, paquetes adicionales **distrib** y **descriptive**

```
(%i1) load (descriptive);
(%o1) /usr/share/maxima/5.10.0/share/contrib/descriptive/descriptive.mac
(%i2) X : [1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7];
(%o2)                [1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7]
(%i3) mean (X);
(%o3)                               4.4
(%i4) std (X);
(%o4)                               2.2
(%i5) load (distrib);
(%o5)    /usr/share/maxima/5.10.0/share/contrib/distrib/distrib.mac
(%i6) quantile_normal (0.95, mean(X), std(X));
(%o6)                         8.018677979293237
```

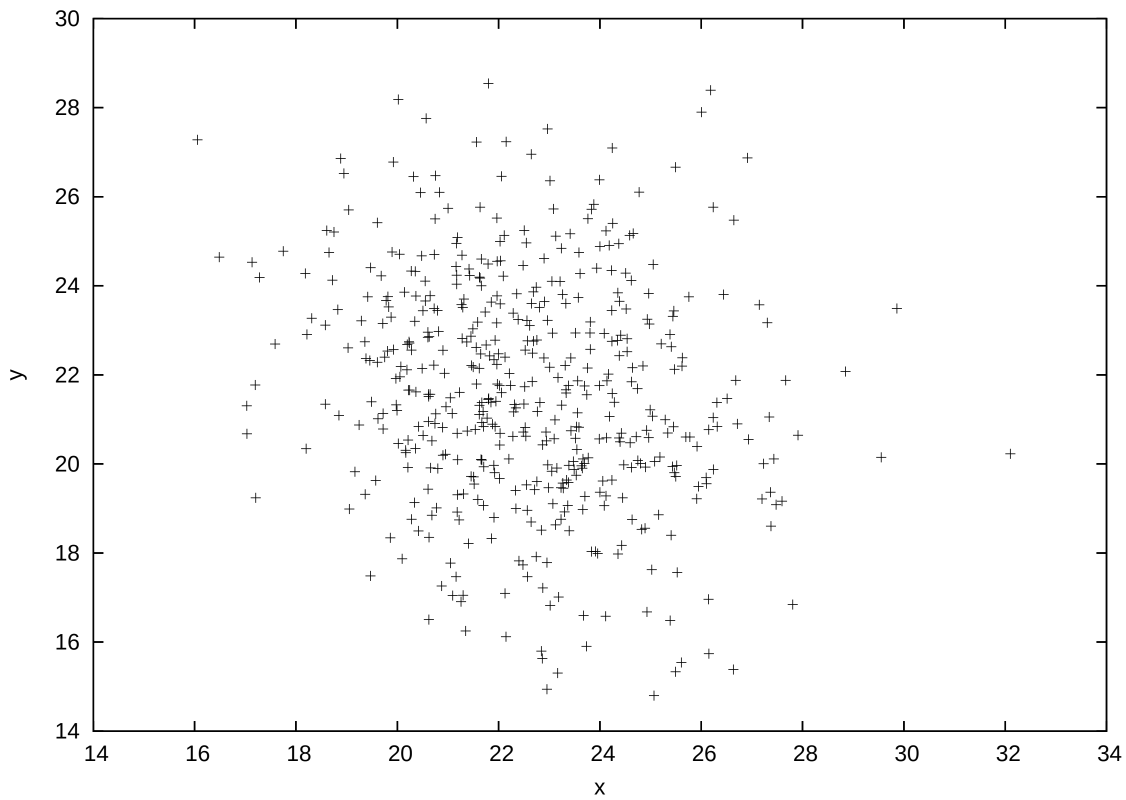
Gráficas estadísticas. Ejemplo: histograma de ejemplares de una distribución normal

```
(%i1) load (descriptive);
(%o1) /usr/share/maxima/5.10.0/share/contrib/descriptive/descriptive.mac
(%i2) load (distrib);
(%o2)     /usr/share/maxima/5.10.0/share/contrib/distrib/distrib.mac
(%i3) X : random_normal (10, 1, 100)$
(%i4) histogram (X, nclasses = 12);
(%o4)                                   0
```


100
80
60
40
20
0
Fr.
6.32025 6.86006 7.39987 7.93967 8.47948 9.01929 9.55909 10.0989 10.6387 11.1785 11.7183 12.2581 12.7979
x

A continuación. Un diagrama de dispersión de ejemplares de una distribución normal bidimensional

```
(%i5) Y : random_normal (10, 1, 100)$
(%i6) S : matrix ([5, - 1], [- 1, 7]);
                                    [  5   - 1 ]
(%o6)                               [          ]
                                    [ - 1   7  ]
(%i7) load (cholesky);
Warning - you are redefining the Maxima function cholesky
(%o7)        /usr/share/maxima/5.10.0/share/contrib/cholesky.mac
(%i8) L : cholesky (S), numer;
               [  2.23606797749979           0         ]
(%o8)          [                                       ]
               [ - .4472135954999579  2.60768096208106 ]
(%i9) XY : transpose (matrix (X, Y))$
(%i10) dataplot (XY . transpose (L));
(%o10)                            0
```



30
28
26
24
22
20
18
16
14
y
14
16
18
20
22
24
26
28
30
32
34
x

Derivación del estimador de probabilidad máxima

```
(%i1) pdf[i] := exp (- (1/2) * (x[i] - mu)^2 / sigma2)
           / sqrt(sigma2) / sqrt(2 * %pi);
                                 1           2
                              (- -) (x  - mu)
                                 2    i
                          exp(----------------)
                                   sigma2
                          ---------------------
                             sqrt(sigma2)
(%o1)              pdf  := ---------------------
                      i         sqrt(2 %pi)
(%i2) logexpand : all;
(%o2)                           all
(%i3) declare (sum, linear);
(%o3)                          done
```

```
(%i4) FPL : sum (log (pdf[i]), i, 1, n);
                          n
                         ====
                         \              2
                          >    (x  - mu)
                         /       i
                         ====
       n log(sigma2)     i = 1              log(%pi) n   log(2) n
(%o4) - ------------- - ---------------- - ---------- - --------
             2              2 sigma2             2           2
(%i5) dmu : diff (FPL, mu);
                          n
                         ====
                         \
                          >     x  - mu n
                         /       i
                         ====
                         i = 1
(%o5)                    ---------------
                             sigma2
```

```
(%i6) dsigma2 : diff (FPL, sigma2);
                  n
                 ====
                 \              2
                  >    (x  - mu)
                 /       i
                 ====
                 i = 1                 n
(%o6)            ---------------- - --------
                            2       2 sigma2
                    2 sigma2
```

```
(%i7) solve (dmu = 0, mu);
                                 n
                                ====
                                \
                                 >    x
                                /      i
                                ====
                                i = 1
(%o7)                     [mu = --------]
                                  n
(%i8) solve (dsigma2 = 0, sigma2);
                               n
                              ====
                              \              2
                               >    (x  - mu)
                              /       i
                              ====
                              i = 1
(%o8)               [sigma2 = ----------------]
                                    n
```