Menggambar Grafik 2D dengan EMT

Notebook ini menjelaskan tentang cara menggambar berbagaikurva dan grafik 2D dengan software EMT. EMT menyediakan fungsi plot2d() untuk menggambar berbagai kurva dan grafik dua dimensi (2D).

Plot Dasar

Ada fungsi dasar dari plot. Terdapat koordinat layar, yang selalu berada pada rentang 0 hingga 1024 pada masing-masing sumbu, tidak peduli apakah layar berbentuk persegi atau tidak. Dan ada juga koordinat plot, yang dapat diatur dengan setplot(). Pemetaan antara koordinat bergantung pada jendela plot yang sedang digunakan. Misalnya, fungsi bawaan shrinkwindow() menyisakan ruang untuk label sumbu dan judul plot.

Dalam contoh, kita hanya menggambar beberapa garis acak dengan berbagai warna. Untuk detail lebih lanjut mengenai fungsi-fungsi ini, pelajari fungsi inti dari EMT.

```
>clg; // clear screen
>window(0,0,1024,1024); // use all of the window
>setplot(0,1,0,1); // set plot coordinates
>hold on; // start overwrite mode
>n=100; X=random(n,2); Y=random(n,2);  // get random points
>colors=rgb(random(n),random(n),random(n)); // get random colors
>loop 1 to n; color(colors[#]); plot(X[#],Y[#]); end; // plot
>hold off; // end overwrite mode
>insimg; // insert to notebook
```

```
>reset;
```

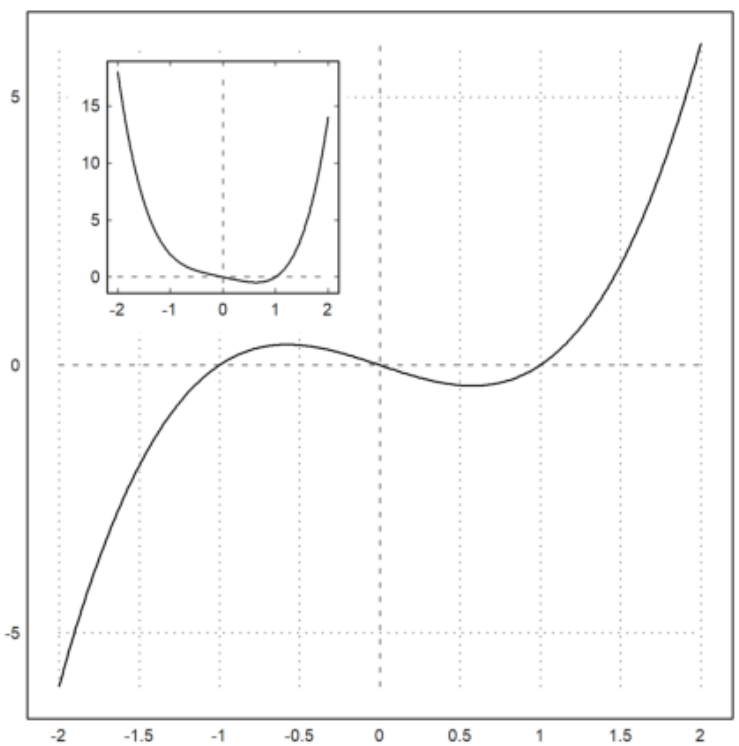
Perlu menahan grafik, karena perintah plot() akan menghapus jendela plot.

Untuk menghapus semua yang telah kita buat, gunakan reset().

Untuk menampilkan gambar hasil plot di layar notebook, perintah plot2d() dapat diakhiri dengan titik dua (:). Cara lain adalah perintah plot2d() diakhiri dengan titik koma (;), kemudian menggunakan perintah insimg() untuk menampilkan gambar hasil plot.

Sebagai contoh lain, kita menggambar sebuah plot sebagai sisipan (inset) di dalam plot lain. Hal ini dilakukan dengan mendefinisikan jendela plot yang lebih kecil. Perlu dicatat bahwa jendela ini tidak menyediakan ruang untuk label sumbu di luar jendela plot. Kita perlu menambahkan margin seperlunya. Perhatikan juga bahwa kita menyimpan dan mengembalikan jendela penuh, serta menahan plot yang ada saat kita menggambar sisipan.

```
>plot2d("x^3-x");
>xw=200; yw=100; ww=300; hw=300;
>ow=window();
>window(xw,yw,xw+ww,yw+hw);
>hold on;
>barclear(xw-50,yw-10,ww+60,ww+60);
>plot2d("x^4-x",grid=6):
```

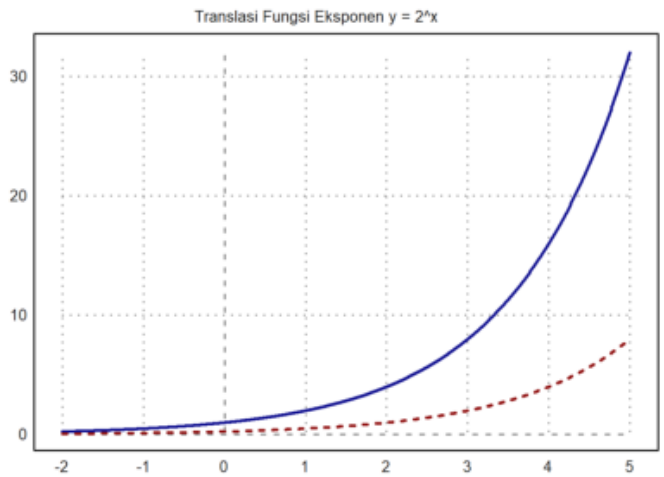



```
>hold off;
>// Fungsi eksponen asli: y = 2^x
>function f(x) := 2^x
>// Fungsi setelah translasi ke kanan 2 satuan: y = 2^(x-2)
>function g(x) := 2^(x-2)
>// Membuat plot perbandingan
>x = linspace(-2,5,1000);
>plot2d(x,f(x),color=blue,thickness=2,title="Translasi Fungsi Eksponen y = 2^x");
```

```
>plot2d(x,g(x),>add,color=red,style="--",thickness=2);
>labelbox(["y = 2^x (Asli)","y = 2^{x-2} (Translasi Kanan 2 satuan)"],
```

```
  Syntax error in expression, or unfinished expression!
  Error in:
  ... Asli)","y = 2^{x-2} (Translasi Kanan 2 satuan)"], ...
                                                       ^
```

```
>insimg;
```




```
>// Membuat inset untuk menunjukkan pergeseran secara detail
>xw=200; yw=100; ww=300; hw=300;
>ow=window();
>window(xw,yw,xw+ww,yw+hw);
>hold on;
>barclear(xw-50,yw-10,ww+60,ww+60);
>plot2d(x,f(x),color=blue,thickness=1,grid=6);
>plot2d(x,g(x),>add,color=red,style="--",thickness=1);
>title("Detail Translasi");
>hold off;
>window(ow);
>// Verifikasi dengan titik tertentu
>printf("Pada x = 2:\n");
```

```
  Built-in function printf needs 2 arguments (got 1)!
  Error in:
  printf("Pada x = 2:\n"); ...
                          ^
```

```
>printf("f(2) = 2^2 = %g\n", f(2));
>printf("g(2) = 2^(2-2) = 2^0 = %g\n", g(2));
>printf("f(0) = 2^0 = %g\n", f(0));
>printf("\nPada x = 4:\n");
```

```
  Built-in function printf needs 2 arguments (got 1)!
  Error in:
  printf("\nPada x = 4:\n"); ...
                           ^
```

```
>printf("f(4) = 2^4 = %g\n", f(4));
>printf("g(4) = 2^(4-2) = 2^2 = %g\n", g(4));
>printf("f(2) = 2^2 = %g\n", f(2));
>window(ow);
```

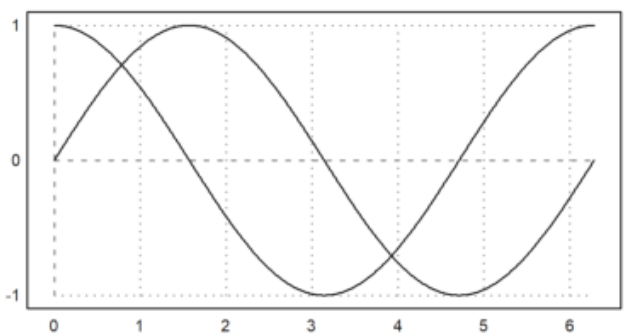
Sebuah plot dengan beberapa gambar dapat dibuat dengan cara yang sama. Untuk itu tersedia fungsi utilitas figure().

Aspek Plot

Secara bawaan, plot menggunakan jendela berbentuk persegi. Anda dapat mengubahnya dengan fungsi aspect(). Jangan lupa untuk mengatur ulang aspek setelahnya. Anda juga dapat mengubah pengaturan bawaan ini melalui menu "Set Aspect" ke rasio aspek tertentu atau menyesuaikan dengan ukuran jendela grafik saat ini.

Namun, Anda juga bisa mengubahnya hanya untuk satu plot. Untuk itu, ukuran area plot saat ini diubah, dan jendela diatur agar label memiliki cukup ruang.

```
>aspect(2); // rasio panjang dan lebar 2:1
>plot2d(["sin(x)","cos(x)"],0,2pi):
```




```
>aspect();
>reset;
```

The reset() function restores plot defaults including the aspect ratio.

Plot 2D di Euler

EMT Math Toolbox memiliki plot 2D, baik untuk data maupun fungsi. EMT menggunakan fungsi plot2d. Fungsi ini dapat memplot fungsi maupun data.

Dimungkinkan juga untuk memplot di Maxima menggunakan Gnuplot atau di Python menggunakan Matplotlib.

Euler dapat memplot grafik 2D dari:

- ekspresi
- fungsi, variabel, atau kurva parametrik,
- vektor nilai x-y,
- kumpulan titik (clouds of points) pada bidang,
- kurva implisit dengan level atau daerah level,
- fungsi kompleks.

Gaya plot mencakup berbagai gaya untuk garis dan titik, plot batang (bar plots), serta plot berarsir (shaded plots).

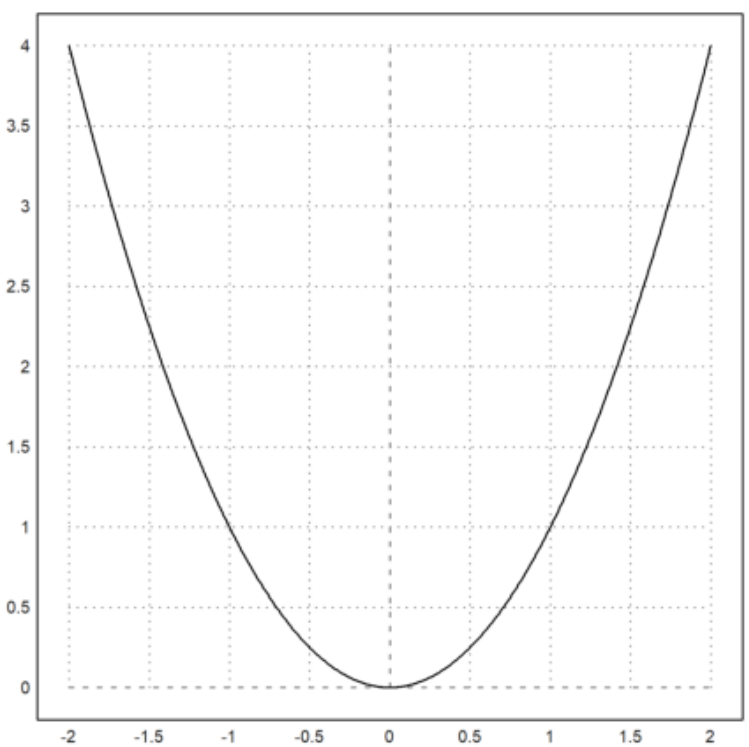
Plot Ekspresi atau Variabel

Sebuah ekspresi tunggal dalam "x" (misalnya "4*x^2") atau nama suatu fungsi (misalnya "f") akan menghasilkan grafik fungsi tersebut.

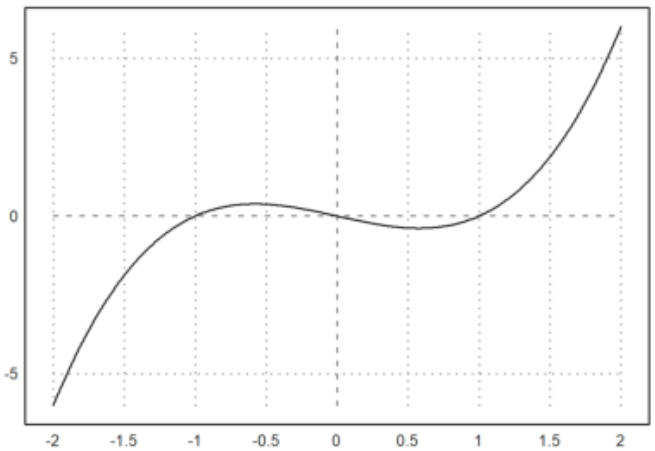
Berikut contoh paling dasar, yang menggunakan rentang default dan mengatur rentang y yang sesuai agar grafik fungsi dapat ditampilkan dengan baik.

Catatan: Jika Anda mengakhiri baris perintah dengan titik dua ":", plot akan dimasukkan ke dalam jendela teks. Jika tidak, tekan TAB untuk melihat plot jika jendela plot tertutup.

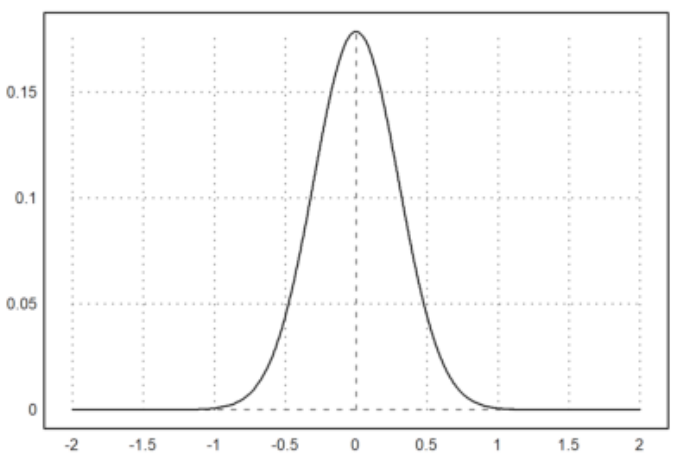
```
>plot2d("x^2"):
```



```
>aspect(1.5); plot2d("x^3-x"):
```




```
>a:=5.6; plot2d("exp(-a*x^2)/a"); insimg(30); // menampilkan gambar hasil plot setinggi 25
```

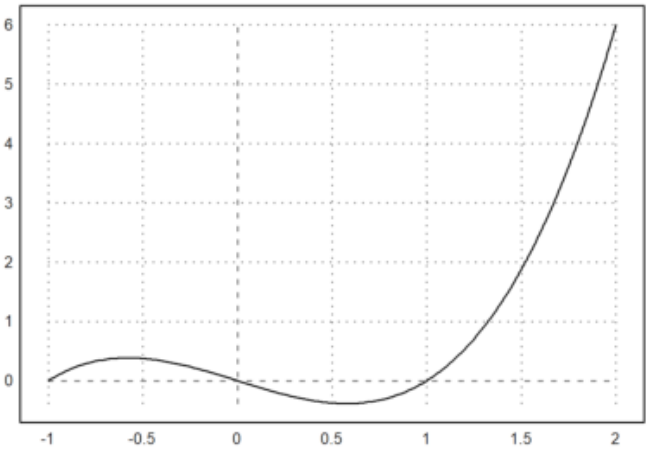


Dari beberapa contoh sebelumnya Anda dapat melihat bahwa aslinya gambar plot menggunakan sumbu X dengan rentang nilai dari -2 sampai dengan 2. Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan nilai-nilai batas X (dan Y) di belakang ekspresi yang digambar.

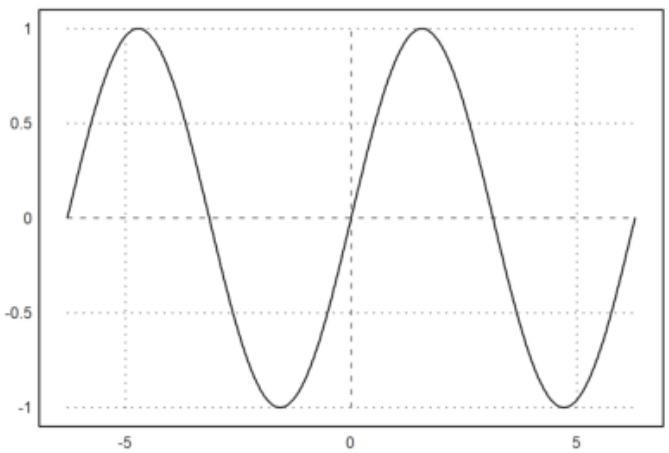
Rentang plot diatur dengan parameter berikut:

- a, b: rentang x (default -2, 2)
- c, d: rentang y (default: menyesuaikan dengan nilai)
- r: alternatif berupa radius di sekitar pusat plot
- cx, cy: koordinat pusat plot (default 0,0)

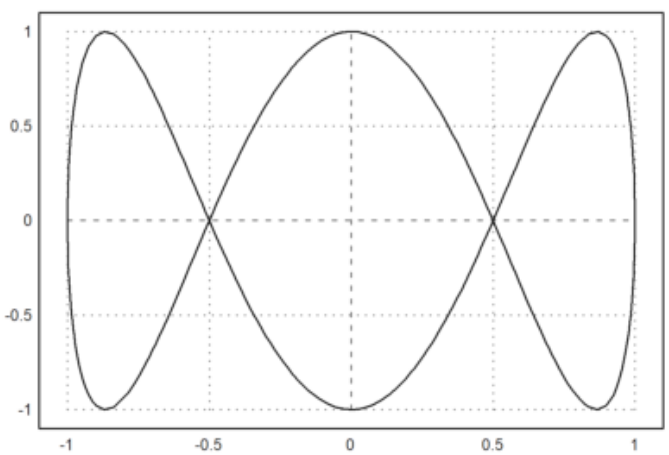
```
>plot2d("x^3-x",-1,2):
```




```
>plot2d("sin(x)",-2*pi,2*pi): // plot sin(x) pada interval [-2pi, 2pi]
```




```
>plot2d("cos(x)","sin(3*x)",xmin=0,xmax=2pi):
```



Alternatif dari penggunaan titik dua adalah perintah insimg(lines), yang menyisipkan plot dengan ukuran sesuai jumlah baris teks yang ditentukan.

Dalam opsi, plot dapat diatur agar muncul:

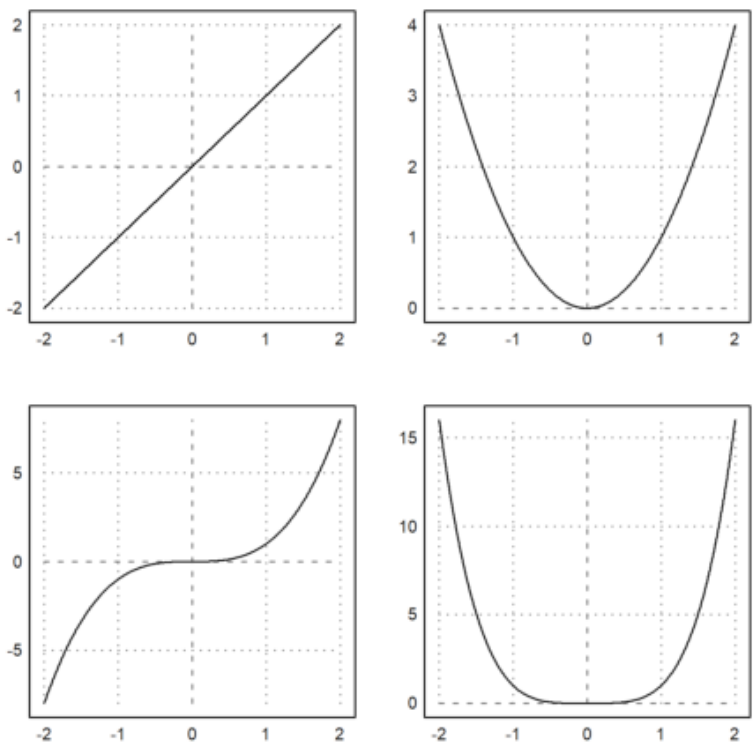
- di jendela terpisah yang dapat diubah ukurannya,
- di dalam jendela notebook.in the notebook window.

Lebih banyak gaya dapat dicapai dengan perintah plot khusus.

Dalam kondisi apa pun, tekan tombol Tab untuk melihat plot jika tertutup.

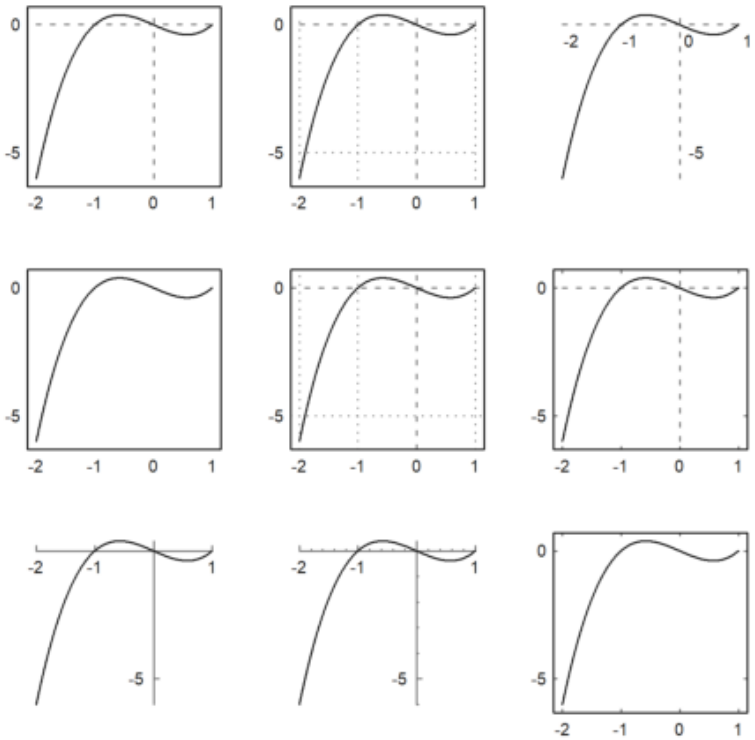
Untuk membagi jendela menjadi beberapa bagian, gunakan perintah figure(). Dalam contoh ini, kita menggambar x^1 hingga x^4 ke dalam 4 bagian jendela. figure(0) akan mengembalikan jendela ke pengaturan default.

```
>reset;
>figure(2,2); ...
>for n=1 to 4; figure(n); plot2d("x^"+n); end; ...
>figure(0):
```



Dalam plot2d(), terdapat beberapa gaya alternatif yang tersedia dengan grid=x. Sebagai gambaran, berbagai gaya grid ditampilkan dalam satu gambar (lihat perintah figure() di bawah). Gaya grid=0 tidak disertakan, karena tidak menampilkan grid maupun bingkai.

```
>figure(3,3); ...
>for k=1:9; figure(k); plot2d("x^3-x",-2,1,grid=k); end; ...
>figure(0):
```

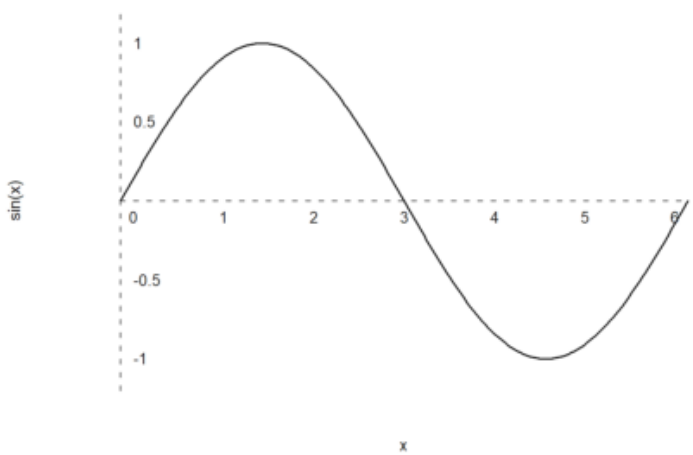


Jika argumen pada plot2d() berupa sebuah ekspresi yang diikuti oleh empat angka, maka angka-angka tersebut adalah rentang x dan y untuk plot.

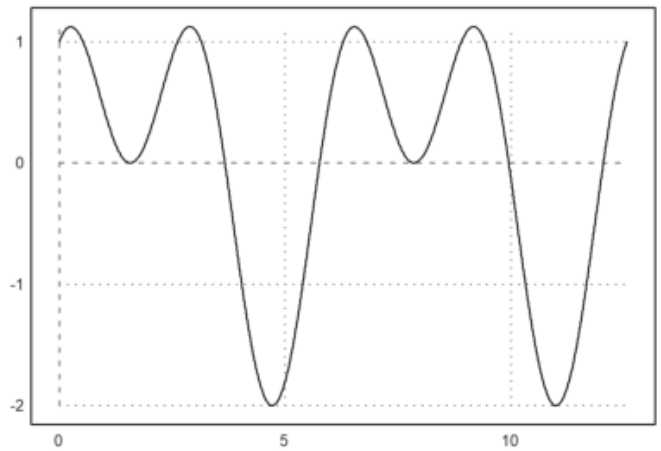
Sebagai alternatif, a, b, c, d dapat ditentukan sebagai parameter yang diberikan seperti a=... dan seterusnya.

Pada contoh berikut, kita mengubah gaya grid, menambahkan label, dan menggunakan label vertikal untuk sumbu y.

```
>aspect(1.5); plot2d("sin(x)",0,2pi,-1.2,1.2,grid=3,xl="x",yl="sin(x)"):
```




```
>plot2d("sin(x)+cos(2*x)",0,4pi):
```

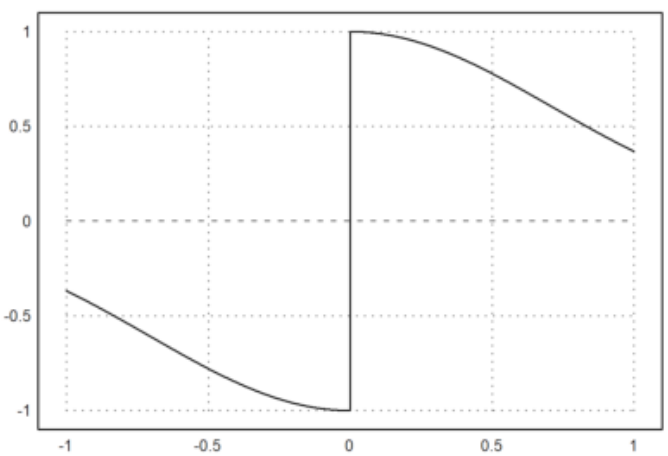


Gambar yang dihasilkan dengan menyisipkan plot ke dalam jendela teks disimpan di direktori yang sama dengan notebook, secara default di dalam subdirektori bernama "images". Gambar tersebut juga digunakan oleh ekspor HTML.

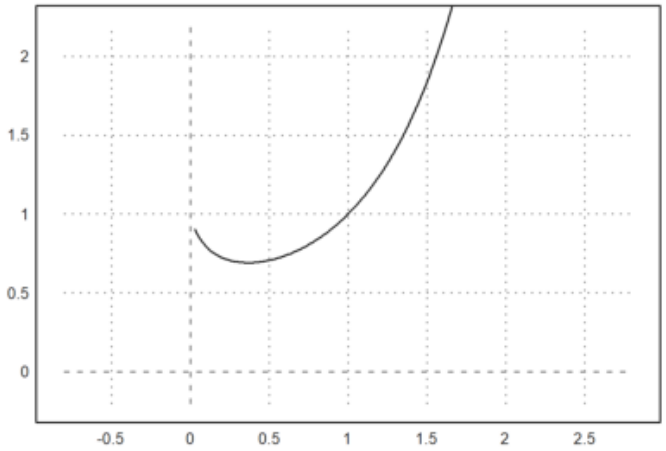
Anda bisa langsung menandai gambar apa pun dan menyalinnya ke clipboard dengan Ctrl-C. Tentu saja, Anda juga dapat mengekspor grafik yang sedang aktif dengan fungsi yang ada di menu File.

Fungsi atau ekspresi dalam plot2d dievaluasi secara adaptif. Untuk mendapatkan kecepatan lebih tinggi, matikan plot adaptif dengan <adaptive dan tentukan jumlah subinterval dengan n=.... Hal ini biasanya hanya diperlukan dalam kasus yang jarang.

```
>plot2d("sign(x)*exp(-x^2)",-1,1,<adaptive,n=10000):
```

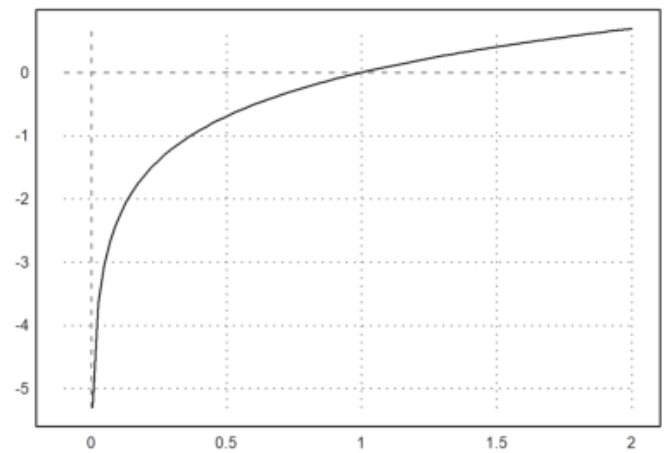



```
>plot2d("x^x",r=1.2,cx=1,cy=1):
```



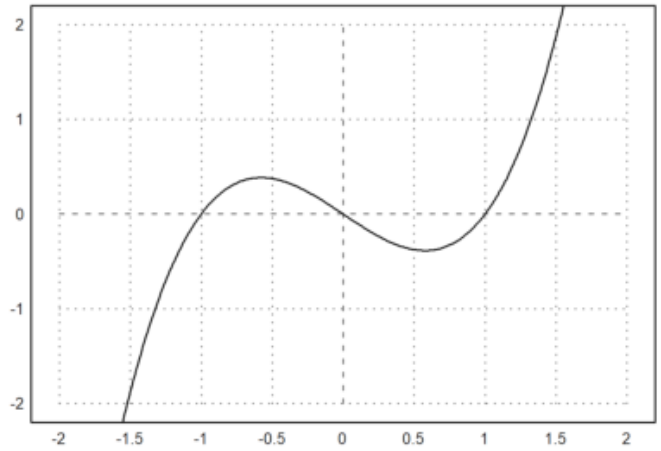
Perlu dicatat bahwa x^x tidak terdefinisi untuk x = 0. Fungsi plot2d akan menangkap error ini, dan mulai memplot segera setelah fungsi tersebut terdefinisi. Hal ini berlaku untuk semua fungsi yang menghasilkan NAN di luar domain (daerah definisi)-nya.

```
>plot2d("log(x)",-0.1,2):
```

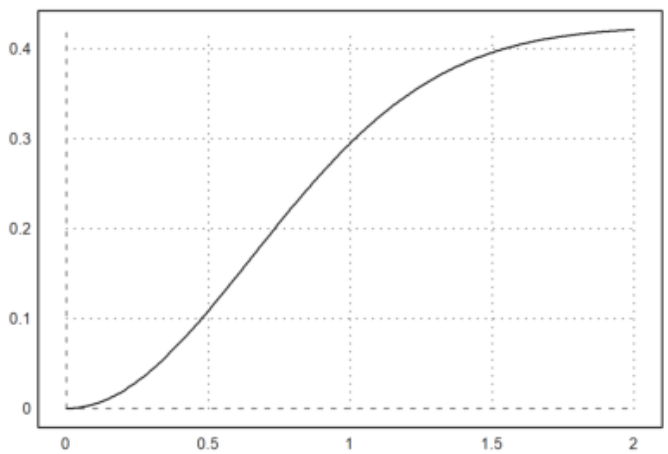


Parameter square=true (atau >square) secara otomatis memilih rentang y sehingga hasilnya berupa jendela plot berbentuk persegi. Perlu dicatat bahwa secara default, Euler menggunakan ruang berbentuk persegi di dalam jendela plot.

```
>plot2d("x^3-x",>square):
```

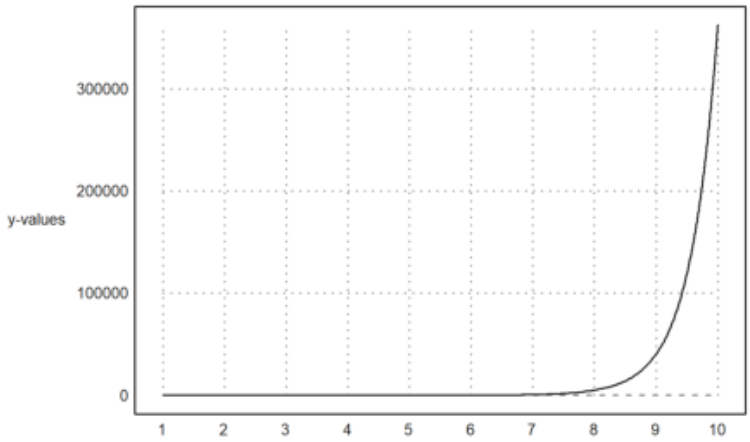



```
>plot2d(''integrate("sin(x)*exp(-x^2)",0,x)'',0,2): // plot integral
```



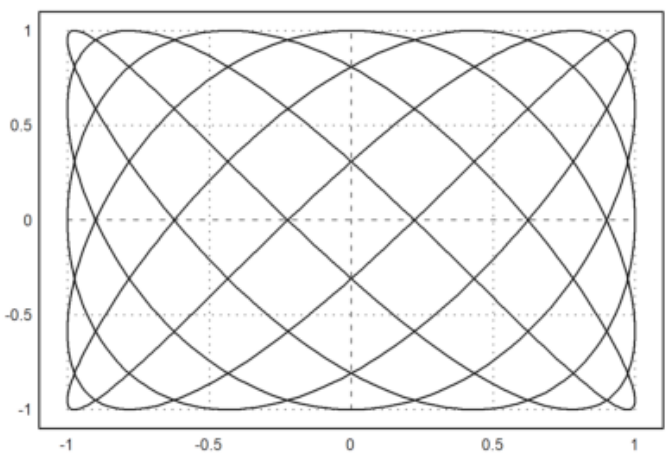
Jika Anda memerlukan lebih banyak ruang untuk label sumbu-y, panggil shrinkwindow() dengan parameter smaller, atau atur nilai positif untuk "smaller" di plot2d().

```
>plot2d("gamma(x)",1,10,yl="y-values",smaller=6,<vertical):
```

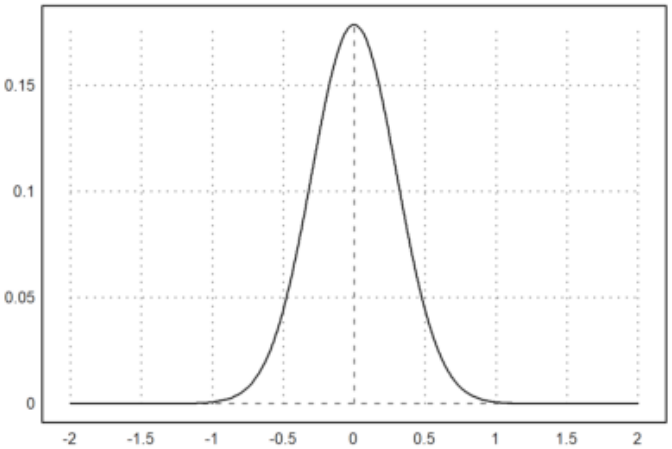


Ekspresi simbolik juga dapat digunakan, karena disimpan sebagai ekspresi string sederhana.

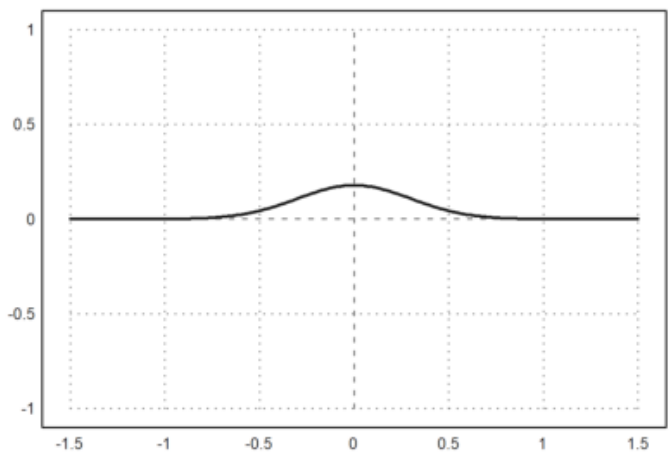
```
>x=linspace(0,2pi,1000); plot2d(sin(5x),cos(7x)):
```



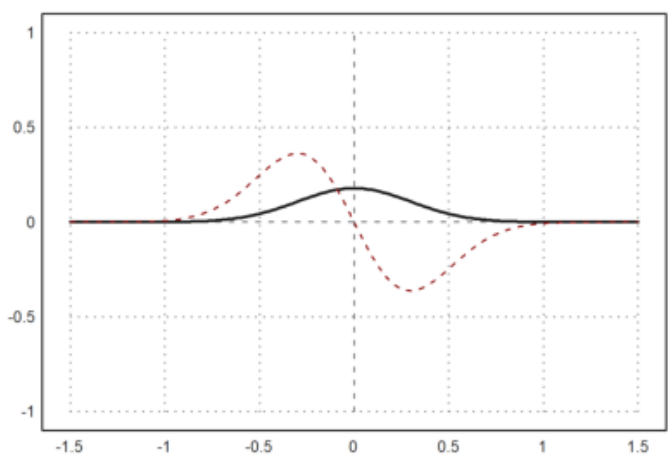

```
>a:=5.6; expr &= exp(-a*x^2)/a; // define expression
>plot2d(expr,-2,2): // plot from -2 to 2
```



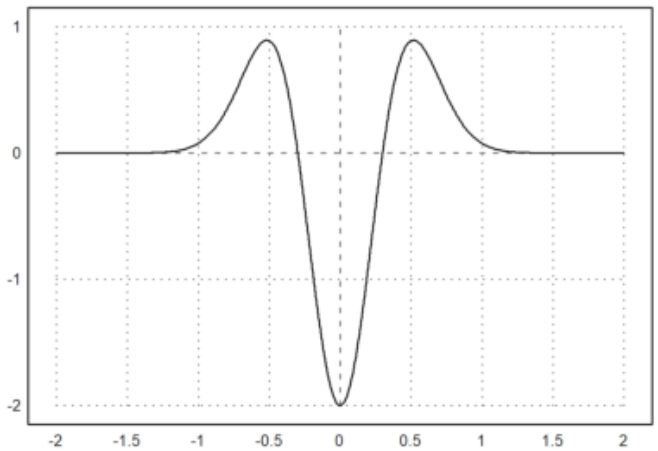

```
>plot2d(expr,r=1,thickness=2): // plot in a square around (0,0)
```



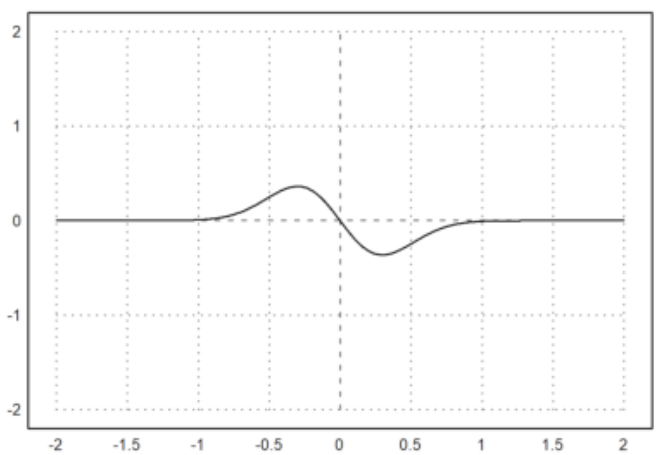

```
>plot2d(&diff(expr,x),>add,style="--",color=red): // add another plot
```



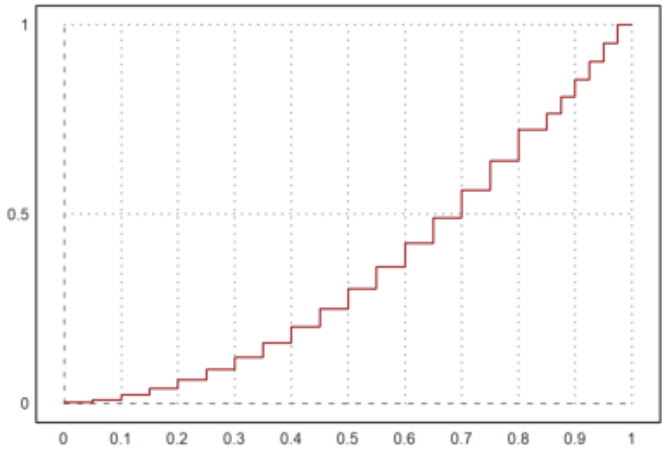
```
>plot2d(&diff(expr,x,2),a=-2,b=2,c=-2,d=1): // plot in rectangle
```



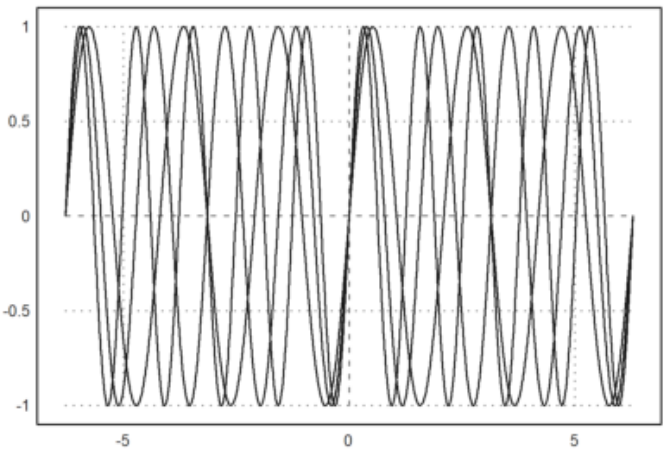

```
>plot2d(&diff(expr,x),a=-2,b=2,>square): // keep plot square
```



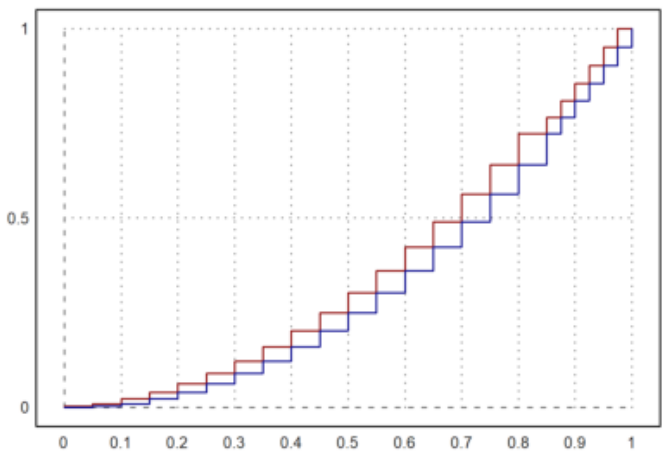

```
>plot2d("x^2",0,1,steps=1,color=red,n=10):
```




```
>plot2d(["sin(3*x)", "sin(4*x)", "sin(5*x)"], -2*pi, 2*pi): // Plot sin(nx) untuk n = 3, 4
```




```
>plot2d("x^2",>add,steps=2,color=blue,n=10):
```

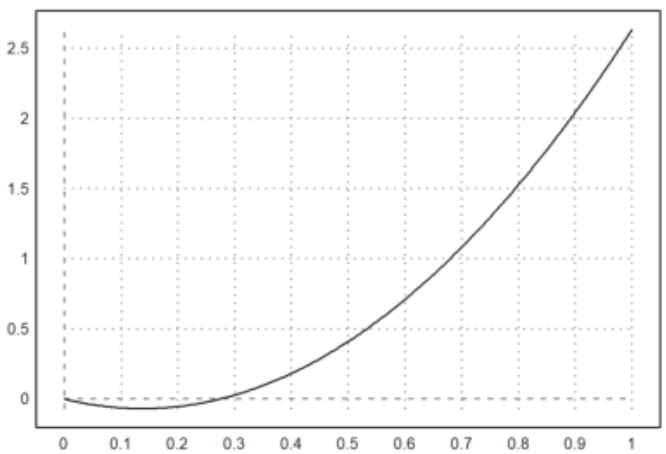


Fungsi dengan Satu Parameter

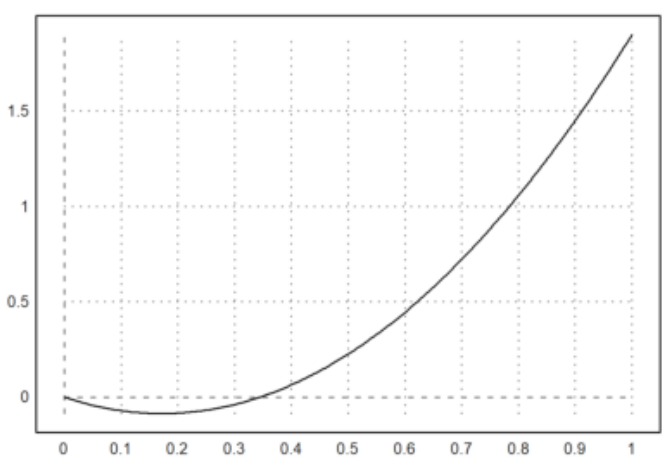
Fungsi plotting terpenting untuk plot bidang datar adalah plot2d(). Fungsi ini diimplementasikan dalam bahasa Euler pada file "plot.e", yang dimuat saat program dijalankan.

Berikut beberapa contoh penggunaan fungsi. Seperti biasa di EMT, fungsi-fungsi yang bekerja untuk fungsi atau ekspresi lain dapat menerima parameter tambahan (selain x) yang bukan variabel global, dengan menggunakan parameter titik koma atau dengan call collection.

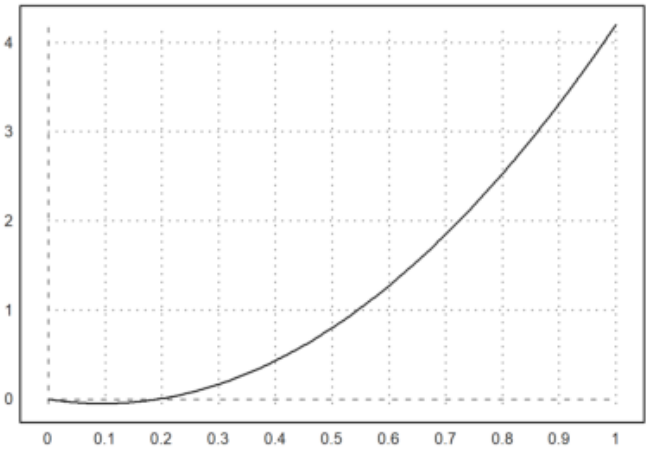
```
>function f(x,a) := x^2/a+a*x^2-x; // define a function
>a=0.3; plot2d("f",0,1;a): // plot with a=0.3
```



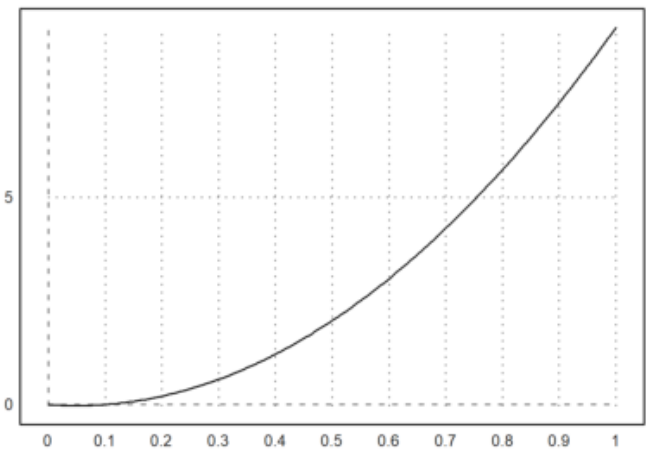

```
>plot2d("f",0,1;0.4): // plot with a=0.4
```



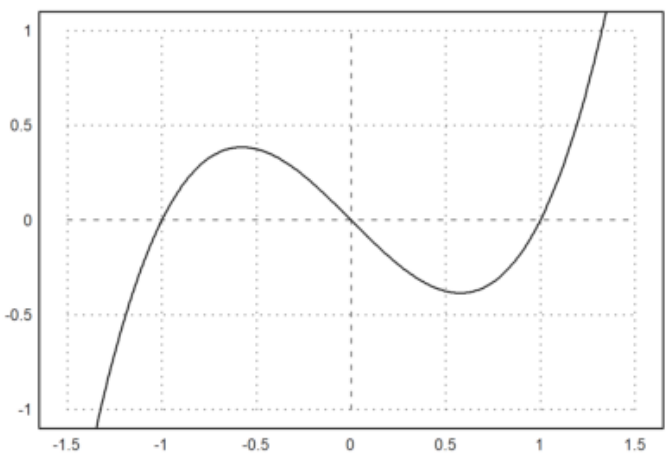

```
>plot2d({{"f",0.2}},0,1): // plot with a=0.2
```




```
>plot2d({{"f(x,b)",b=0.1}},0,1): // plot with 0.1
```




```
>function f(x) := x^3-x; ...
>plot2d("f",r=1):
```



Berikut adalah ringkasan fungsi yang diterima:

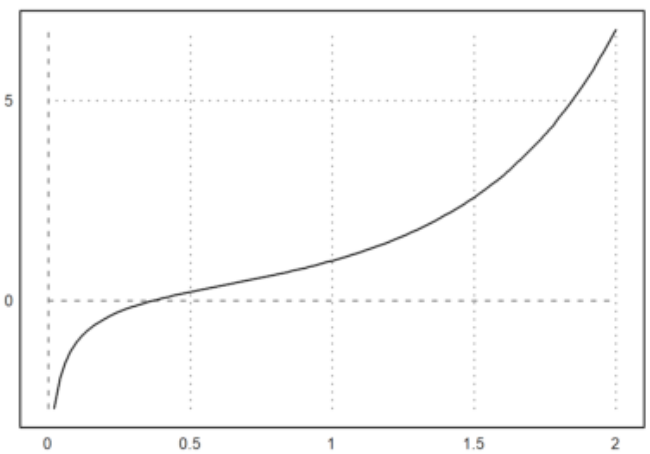
- ekspresi atau ekspresi simbolik dalam x
- fungsi atau fungsi simbolik dengan nama seperti "f"
- fungsi simbolik cukup dengan nama f

Fungsi plot2d() juga menerima fungsi simbolik. Untuk fungsi simbolik, cukup dengan menggunakan namanya saja.

```
>function f(x) &= diff(x^x,x)
```

$$x^{x} \, (\log(x) + 1)$$

```
>plot2d(f,0,2):
```

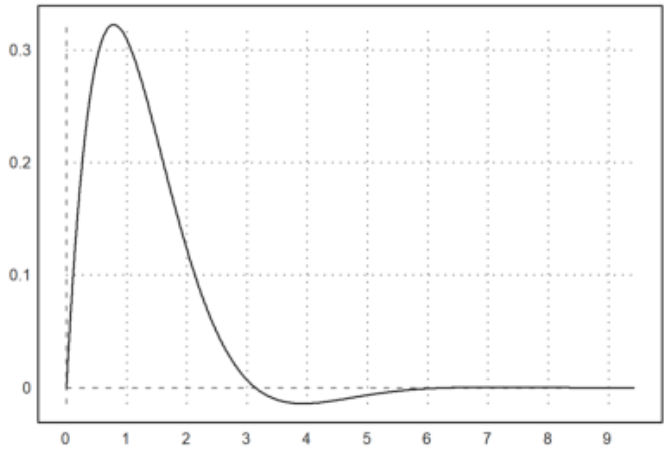


Tentu saja, untuk ekspresi atau ekspresi simbolik, nama variabel saja sudah cukup untuk memplotnya.

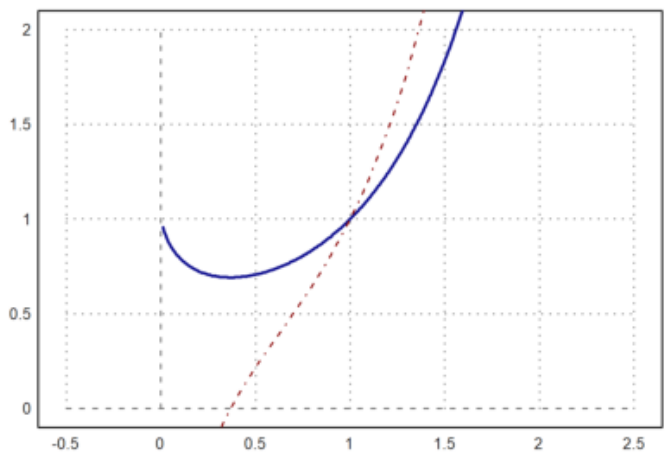
```
>expr &= sin(x)*exp(-x)
```

$$E^{-x} \, \sin(x)$$

```
>plot2d(expr,0,3pi):
```




```
>function f(x) &= x^x;
>plot2d(f,r=1,cx=1,cy=1,color=blue,thickness=2);
>plot2d(&diff(f(x),x),>add,color=red,style="-.-"):
```



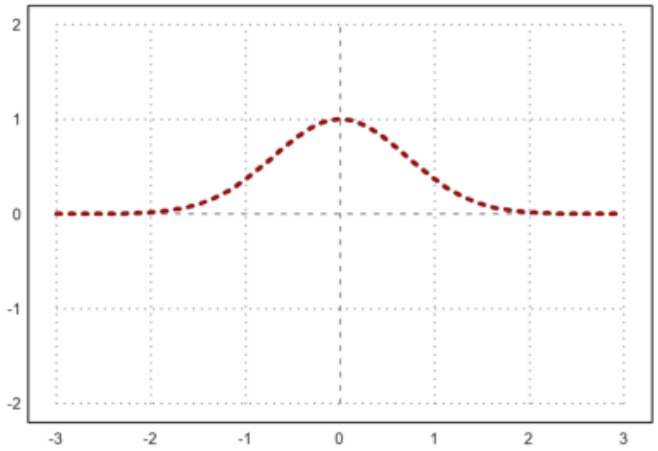
Untuk gaya garis tersedia berbagai opsi.

- style="...": pilih dari "-", "–", "-.", ".", ".-.", "-.-".
- color: lihat daftar warna di bawah.
- thickness: default adalah 1.

Warna dapat dipilih sebagai salah satu warna bawaan, atau sebagai warna RGB.

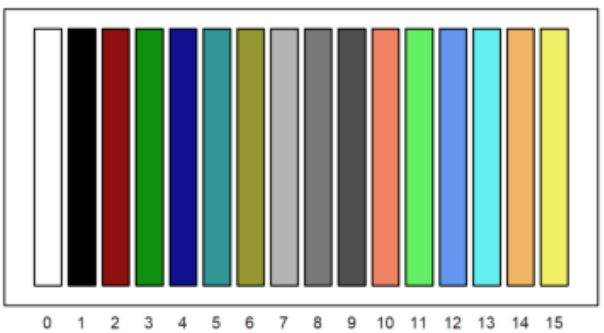
- 0..15: indeks warna bawaan.
- konstanta warna: white, black, red, green, blue, cyan, olive, lightgray, gray, darkgray, orange, lightgreen, turquoise, lightblue, lightorange, yellow
- rgb(red,green,blue): parameternya berupa bilangan real dalam [0,1].

```
>plot2d("exp(-x^2)",r=2,color=red,thickness=3,style="--"):
```

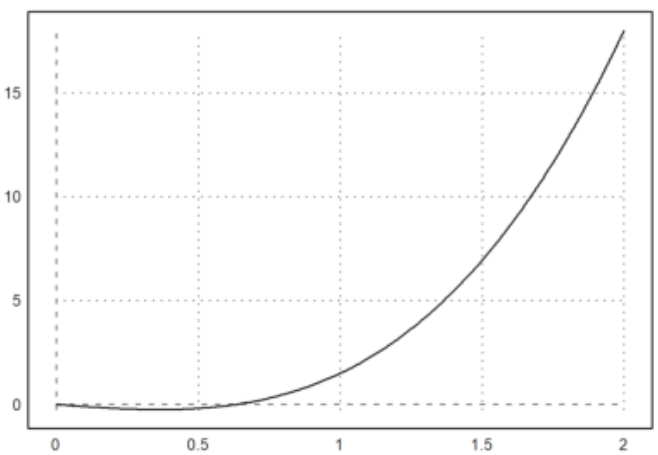


Berikut adalah tampilan warna-warna bawaan (predefined) dari EMT.

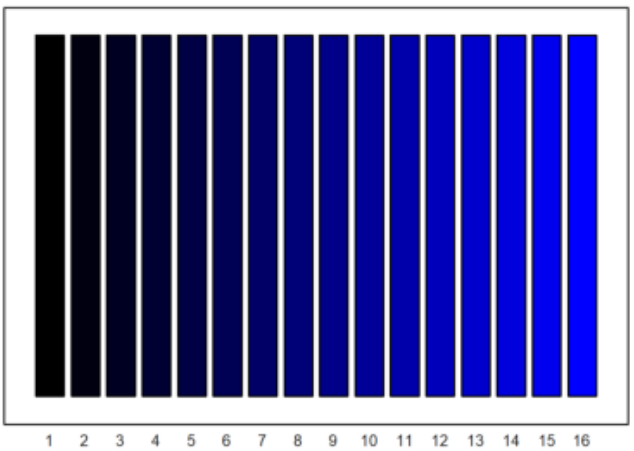
```
>aspect(2); columnsplot(ones(1,16),lab=0:15,grid=0,color=0:15):
```




```
>function f(x,a) := x^3/a+a*x^3-x; // define a function
>a=0.5; plot2d("f",0,2;a): // plot with a=0.5
```



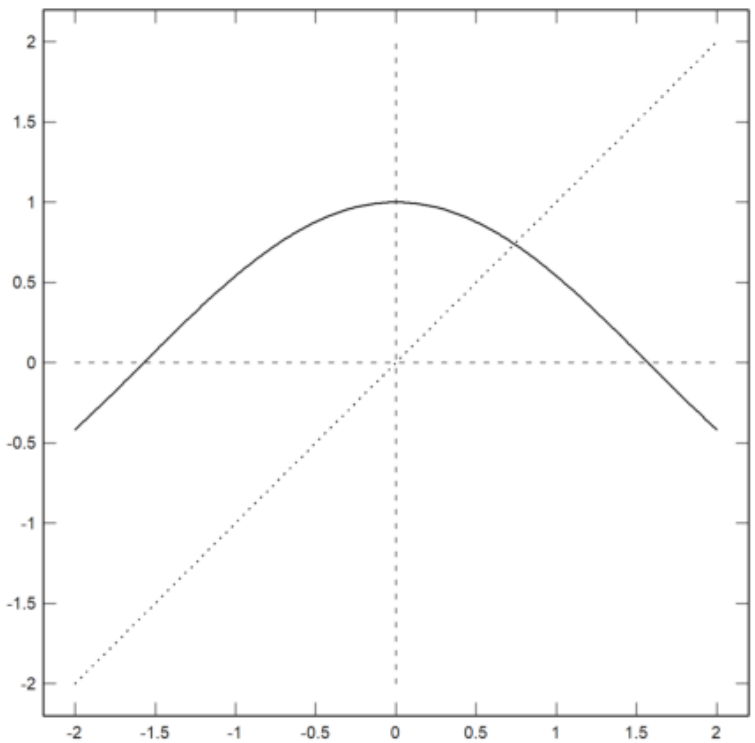
```
>
>columnsplot(ones(1,16),grid=0,color=rgb(0,0,linspace(0,1,15))):
```



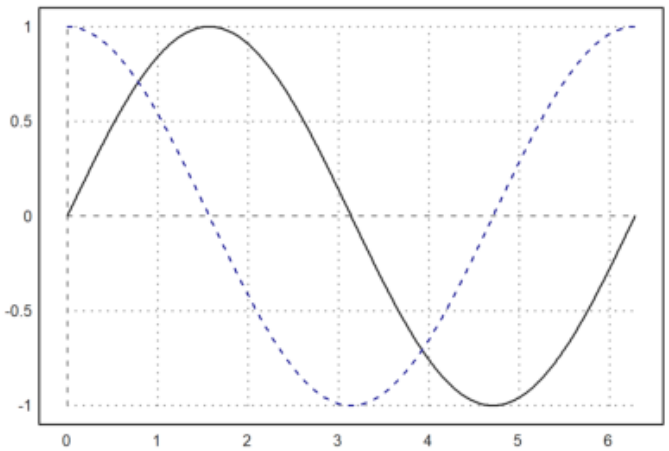
Menggambar Beberapa Kurva pada bidang koordinat yang sama

Memplot lebih dari satu fungsi (multiple functions) dalam satu jendela dapat dilakukan dengan beberapa cara. Salah satu caranya adalah menggunakan >add untuk beberapa pemanggilan plot2d, kecuali pada pemanggilan pertama. Fitur ini sudah kita gunakan pada contoh-contoh di atas.

```
>aspect(); plot2d("cos(x)",r=2,grid=6); plot2d("x",style=".",>add):
```

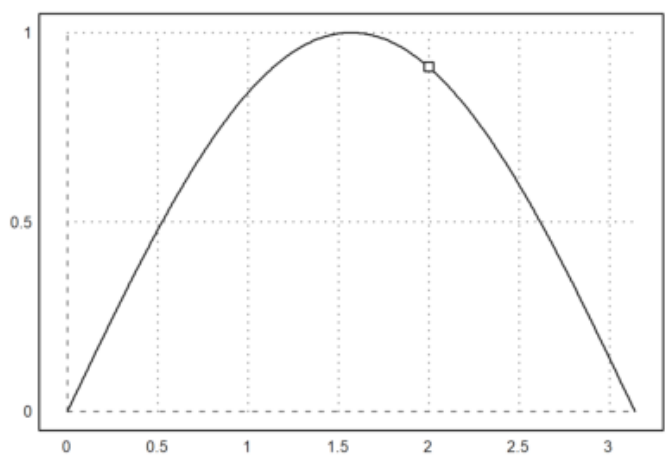


```
>aspect(1.5); plot2d("sin(x)",0,2pi); plot2d("cos(x)",color=blue,style="--",>add):
```



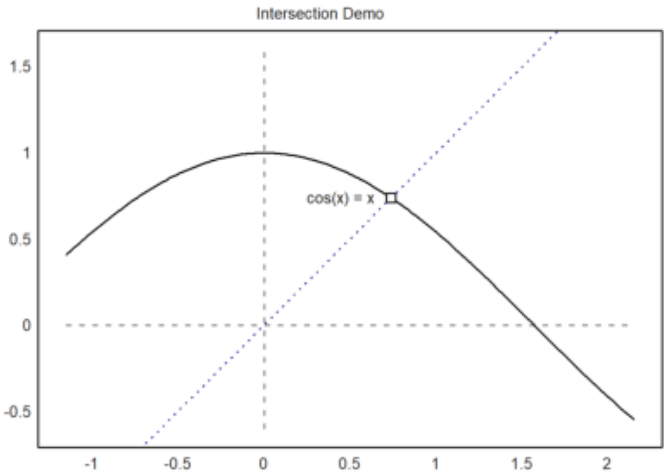
Salah satu kegunaan >add adalah untuk menambahkan titik pada kurva.

```
>plot2d("sin(x)",0,pi); plot2d(2,sin(2),>points,>add):
```



Kita menambahkan titik potong dengan sebuah label (pada posisi "cl" untuk center left), dan menyisipkan hasilnya ke dalam notebook. Kita juga menambahkan judul pada plot.

```
>plot2d(["cos(x)","x"],r=1.1,cx=0.5,cy=0.5, ...
>  color=[black,blue],style=["-","."], ...
>  grid=1);
>x0=solve("cos(x)-x",1);  ...
>  plot2d(x0,x0,>points,>add,title="Intersection Demo");  ...
>  label("cos(x) = x",x0,x0,pos="cl",offset=20):
```

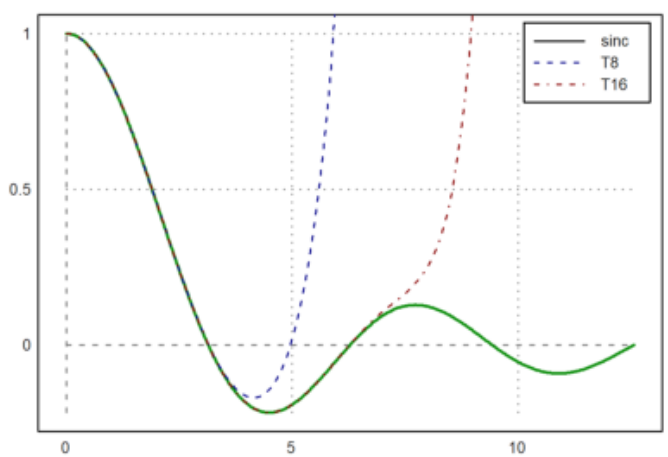


Dalam demo berikut, kita memplot fungsi sinc(x) = sin(x)/x serta ekspansi Taylor ke-8 dan ke-16. Ekspansi ini dihitung menggunakan Maxima melalui ekspresi simbolik.
Plot ini dibuat dengan perintah multi-baris berikut yang memanggil plot2d() sebanyak tiga kali. Pemanggilan kedua dan ketiga menggunakan flag >add, yang membuat plot tersebut menggunakan rentang sebelumnya.
Kita juga menambahkan kotak label untuk menjelaskan fungsi-fungsi tersebut.

```
>$taylor(sin(x)/x,x,0,4)
```

$$\frac{x^4}{120}-\frac{x^2}{6}+1$$

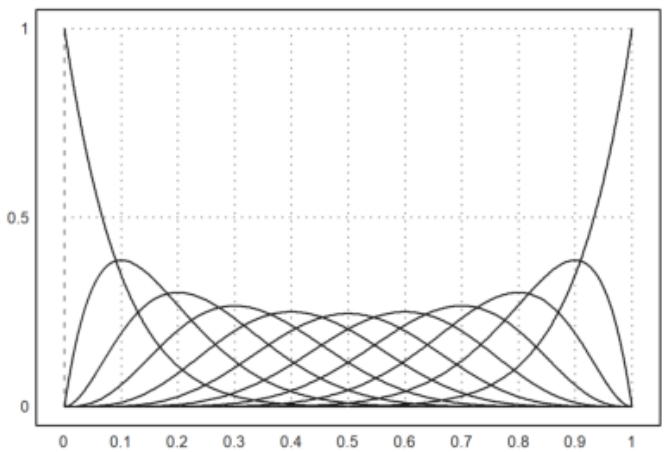
```
>plot2d("sinc(x)",0,4pi,color=green,thickness=2); ...
>  plot2d(&taylor(sin(x)/x,x,0,8),>add,color=blue,style="--"); ...
>  plot2d(&taylor(sin(x)/x,x,0,16),>add,color=red,style="-.-"); ...
>  labelbox(["sinc","T8","T16"],styles=["-","--","-.-"], ...
>    colors=[black,blue,red]):
```



Pada contoh berikut, kita menghasilkan Polinomial Bernstein.

$$B_i(x) = \binom{n}{i} x^i (1-x)^{n-i}$$

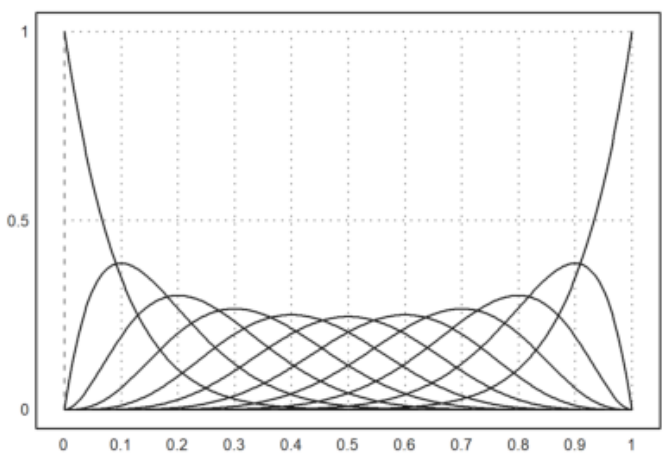
```
>plot2d("(1-x)^10",0,1); // plot first function
>for i=1 to 10; plot2d("bin(10,i)*x^i*(1-x)^(10-i)",>add); end;
>insimg;
```



Metode kedua adalah menggunakan sepasang matriks, yaitu matriks nilai-x dan matriks nilai-y dengan ukuran yang sama.

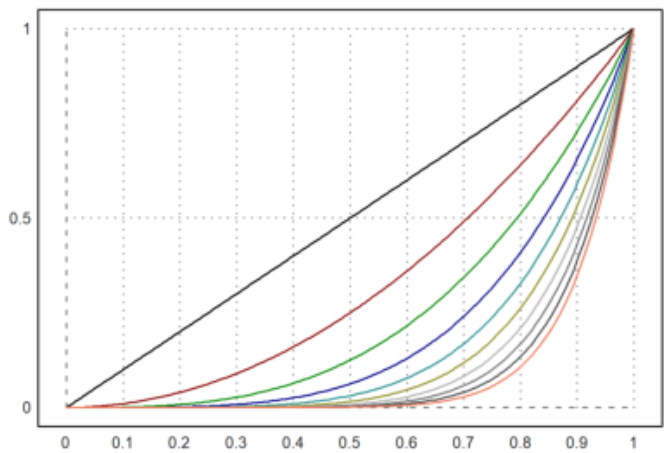
Kita menghasilkan sebuah matriks nilai dengan satu Polinomial Bernstein pada setiap baris. Untuk itu, kita cukup menggunakan sebuah vektor kolom dari i. Silakan lihat bagian pendahuluan tentang bahasa matriks untuk mempelajari lebih detail.

```
>x=linspace(0,1,500);
>n=10; k=(0:n)'; // n is row vector, k is column vector
>y=bin(n,k)*x^k*(1-x)^(n-k); // y is a matrix then
>plot2d(x,y):
```



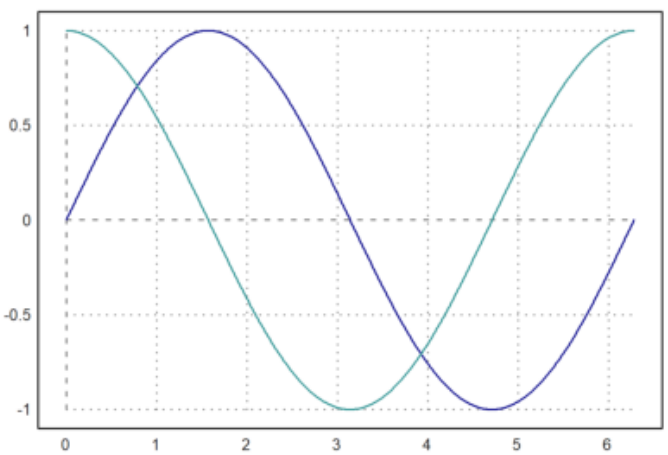
Perlu dicatat bahwa parameter color dapat berupa sebuah vektor. Maka, setiap warna akan digunakan untuk setiap baris dari matriks.

```
>x=linspace(0,1,200); y=x^(1:10)'; plot2d(x,y,color=1:10):
```

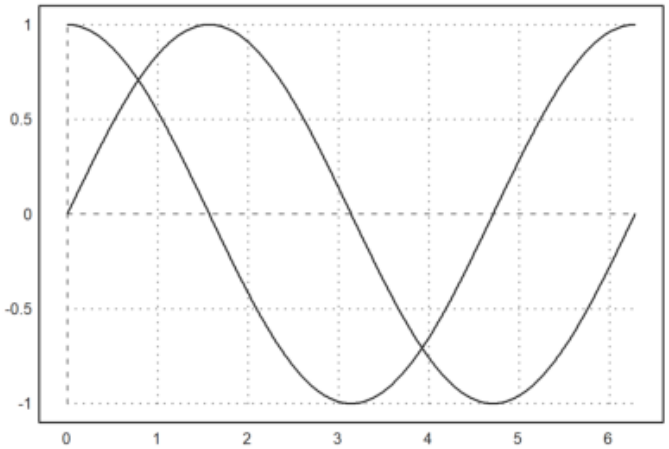


Metode lain adalah menggunakan sebuah vektor ekspresi (string). Kemudian, Anda dapat menggunakan sebuah array warna, array gaya, dan array ketebalan dengan panjang yang sama.

```
>plot2d(["sin(x)","cos(x)"],0,2pi,color=4:5):
```




```
>plot2d(["sin(x)","cos(x)"],0,2pi): // plot vector of expressions
```



Kita dapat memperoleh vektor seperti itu dari Maxima dengan menggunakan makelist() dan mxm2str().

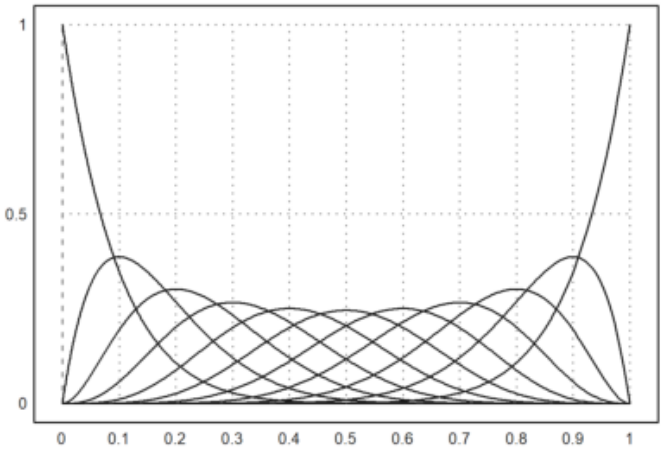
```
>v &= makelist(binomial(10,i)*x^i*(1-x)^(10-i),i,0,10) // make list
```

$$\left[(1-x)^{10},\ 10\,(1-x)^{9}\,x,\ 45\,(1-x)^{8}\,x^{2},\ 120\,(1-x)^{7}\,x^{3},\ 210\,(1-x)^{6}\,x^{4},\ 252\,(1-x)^{5}\,x^{5},\ 210\,(1-x)^{4}\,x^{6},\ 120\,(1-x)^{3}\,x^{7},\ 45\,(1-x)^{2}\,x^{8},\ 10\,(1-x)\,x^{9},\ x^{10}\right]$$

```
>mxm2str(v) // get a vector of strings from the symbolic vector
```

```
(1-x)^10
10*(1-x)^9*x
45*(1-x)^8*x^2
120*(1-x)^7*x^3
210*(1-x)^6*x^4
252*(1-x)^5*x^5
210*(1-x)^4*x^6
120*(1-x)^3*x^7
45*(1-x)^2*x^8
10*(1-x)*x^9
x^10
```

```
>plot2d(mxm2str(v),0,1): // plot functions
```

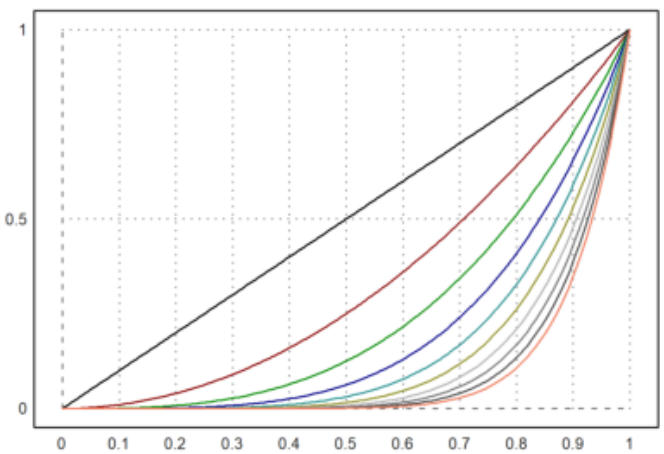


Alternatif lain adalah menggunakan bahasa matriks dari Euler.

Jika sebuah ekspresi menghasilkan matriks fungsi, dengan satu fungsi di setiap baris, maka semua fungsi tersebut akan diplot dalam satu plot.

Untuk itu, gunakan sebuah vektor parameter dalam bentuk vektor kolom. Jika sebuah array warna ditambahkan, maka array tersebut akan digunakan untuk setiap baris plot.

```
>n=(1:10)'; plot2d("x^n",0,1,color=1:10):
```

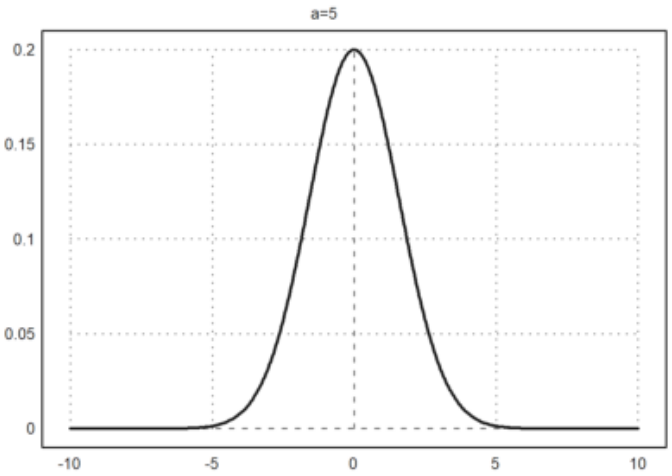


Expressions and one-line functions can see global variables.

If you cannot use a global variable, you need to use a function with an extra parameter, and pass this parameter as a semicolon parameter.

Take care, to put all assigned parameters to the end of the plot2d command. In the example we pass a=5 to the function f, which we plot from -10 to 10.

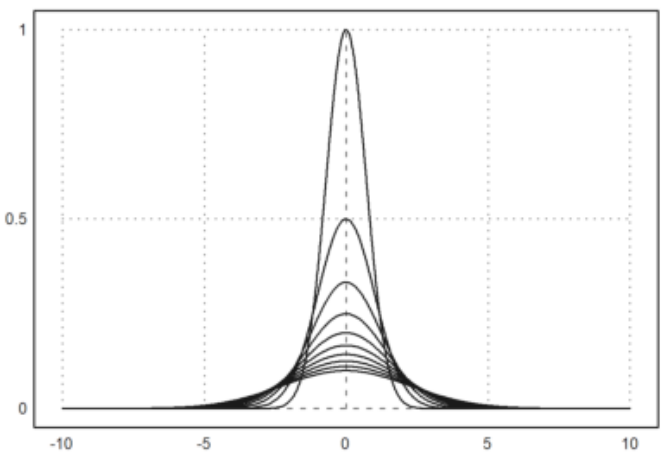
```
>function f(x,a) := 1/a*exp(-x^2/a); ...
>plot2d("f",-10,10;5,thickness=2,title="a=5"):
```



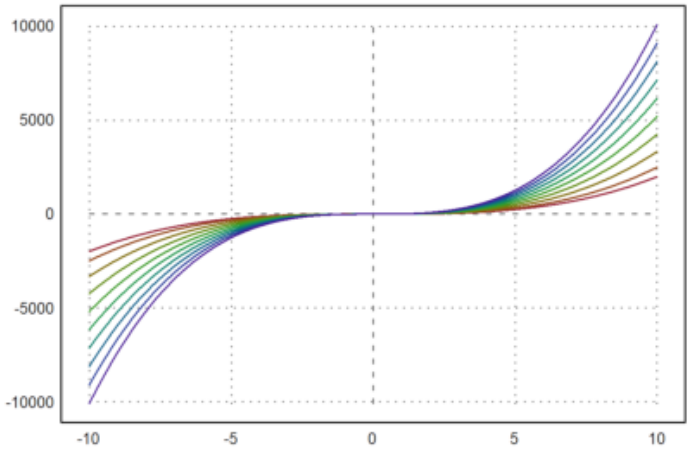
Sebagai alternatif, gunakan sebuah koleksi yang berisi nama fungsi dan semua parameter tambahannya. Daftar khusus ini disebut call collections, dan cara ini lebih disarankan untuk mengirimkan argumen ke sebuah fungsi yang juga dikirimkan sebagai argumen ke fungsi lain.

Pada contoh berikut, kita menggunakan sebuah perulangan untuk memplot beberapa fungsi (lihat tutorial tentang pemrograman dengan perulangan for).

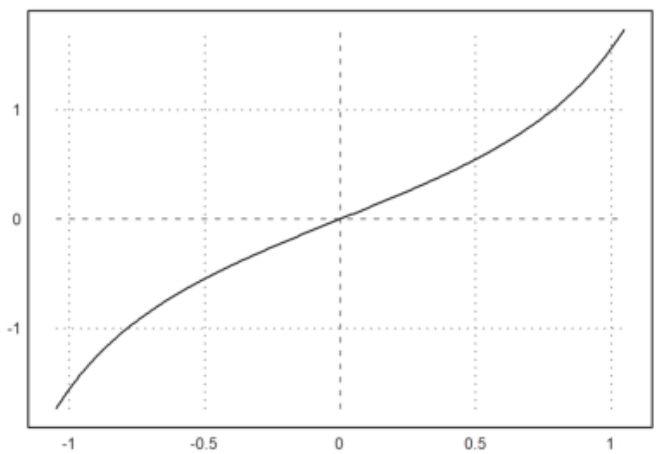
```
>plot2d({{"f",1}},-10,10); ...
>for a=2:10; plot2d({{"f",a}},>add); end:
```




```
>aspect(1.5): // rasio aspek grafik
```

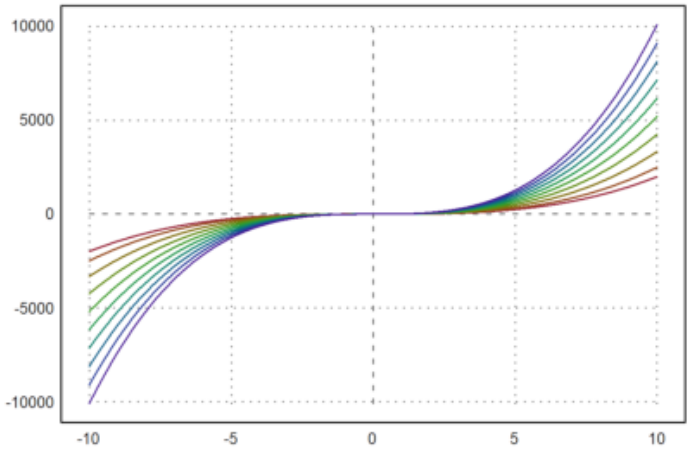



```
>plot2d("tan(x)", -pi/3, pi/3): // plot tan(x) di interval aman
```



Kita bisa memperoleh hasil yang sama dengan cara berikut menggunakan bahasa matriks dari EMT. Setiap baris dari matriks f(x,a) merupakan satu fungsi. Selain itu, kita dapat mengatur warna untuk setiap baris matriks. Klik ganda pada fungsi getspectral() untuk penjelasan lebih lanjut.

```
>
>x=-10:0.01:10; a=(1:10)'; plot2d(x,f(x,a),color=getspectral(a/10)):
```



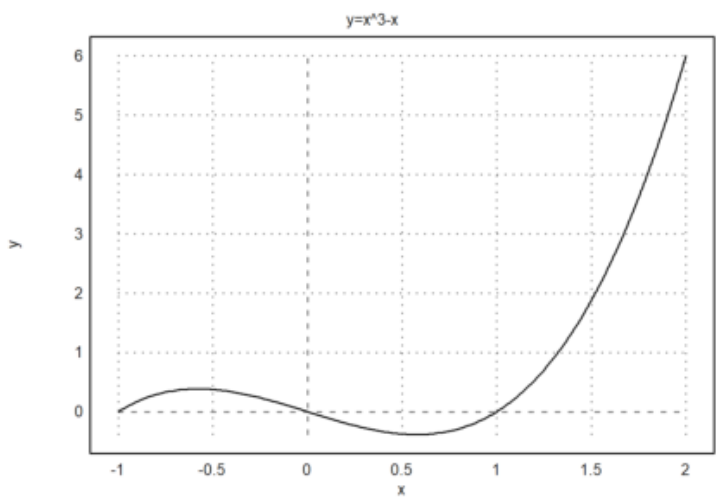
Label Teks

Hiasan sederhana dapat berupa:

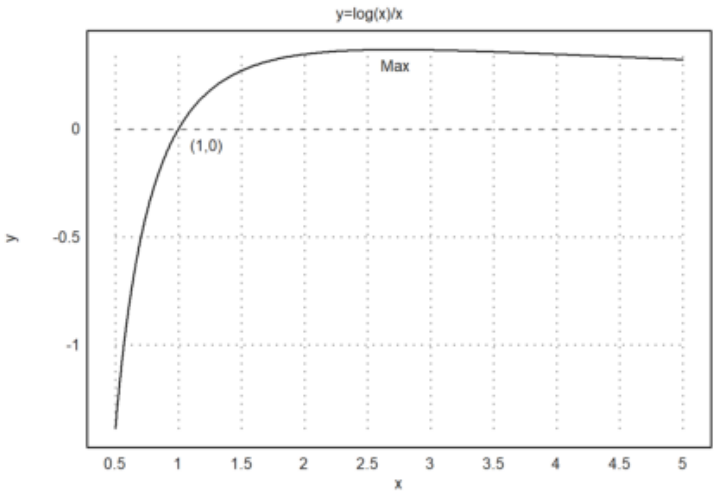
- judul dengan title="..."
- label sumbu-x dan sumbu-y dengan xl="...", yl="..."
- label teks lain dengan label("...",x,y)

Perintah label akan menampilkan teks pada plot saat ini di koordinat plot (x,y). Perintah ini juga dapat menerima argumen posisi.

```
>plot2d("x^3-x",-1,2,title="y=x^3-x",yl="y",xl="x"):
```

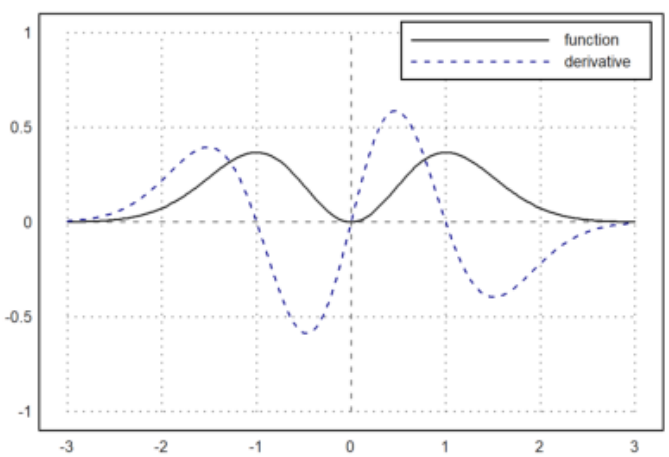



```
>expr := "log(x)/x"; ...
>  plot2d(expr,0.5,5,title="y="+expr,xl="x",yl="y"); ...
>  label("(1,0)",1,0); label("Max",E,expr(E),pos="lc"):
```



Terdapat juga fungsi labelbox(), yang dapat menampilkan fungsi-fungsi beserta sebuah teks. Fungsi ini menerima vektor berupa string dan warna, masing-masing satu untuk setiap fungsi.

```
>function f(x) &= x^2*exp(-x^2);  ...
>plot2d(&f(x),a=-3,b=3,c=-1,d=1);  ...
>plot2d(&diff(f(x),x),>add,color=blue,style="--"); ...
>labelbox(["function","derivative"],styles=["-","--"], ...
>   colors=[black,blue],w=0.4):
```

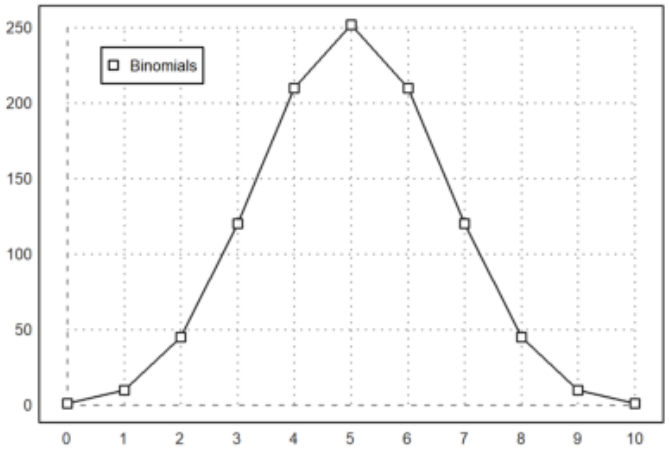


Kotak secara bawaan ditempatkan di kanan atas, tetapi >left akan menempatkannya di kiri atas. Anda dapat memindahkannya ke posisi mana pun yang diinginkan. Posisi jangkar adalah sudut kanan atas kotak, dan angkanya berupa pecahan dari ukuran jendela grafik. Lebarnya diatur secara otomatis.

Untuk plot titik, kotak label juga berfungsi. Tambahkan parameter >points, atau sebuah vektor berisi flag, satu untuk setiap label.

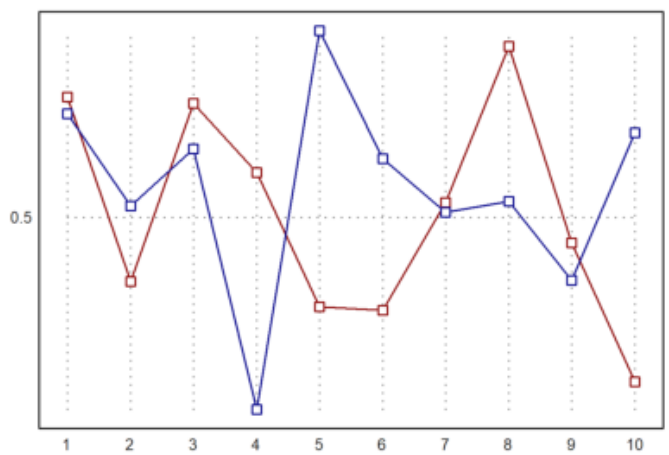
Pada contoh berikut, hanya ada satu fungsi. Jadi kita dapat menggunakan string alih-alih vektor string. Untuk contoh ini, kita atur warna teks menjadi hitam.

```
>n=10; plot2d(0:n,bin(n,0:n),>addpoints); ...
>labelbox("Binomials",styles="[]",>points,x=0.1,y=0.1, ...
>tcolor=black,>left):
```



Gaya plot ini juga tersedia dalam statplot(). Seperti pada plot2d(), warna dapat diatur untuk setiap baris plot. Terdapat lebih banyak plot khusus untuk keperluan statistik (lihat tutorial tentang statistik).

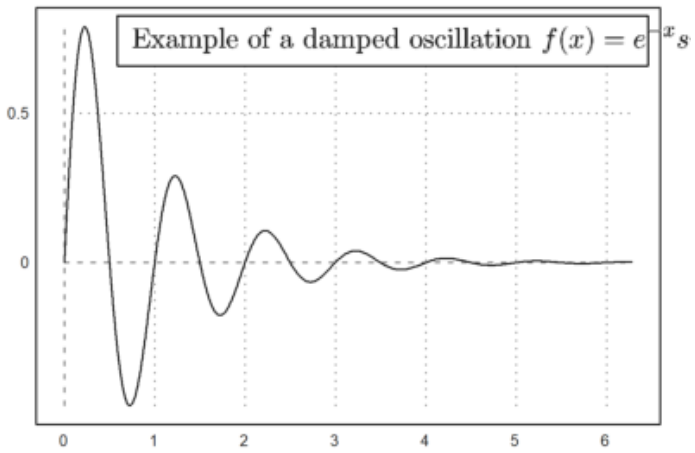
```
>statplot(1:10,random(2,10),color=[red,blue]):
```



Fitur serupa adalah fungsi textbox().

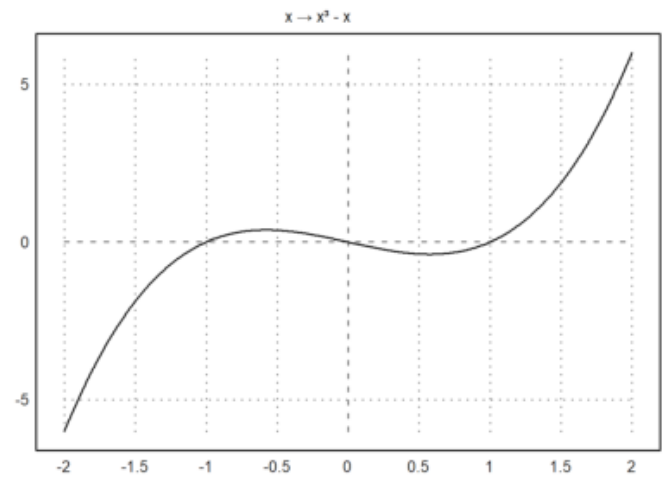
Lebarnya secara bawaan mengikuti lebar maksimal dari baris teks. Namun, lebar tersebut juga dapat diatur oleh pengguna.

```
>function f(x) &= exp(-x)*sin(2*pi*x); ...
>plot2d("f(x)",0,2pi); ...
>textbox(latex("\text{Example of a damped oscillation}\ f(x)=e^{-x}sin(2\pi x)"),w=0.85):
```

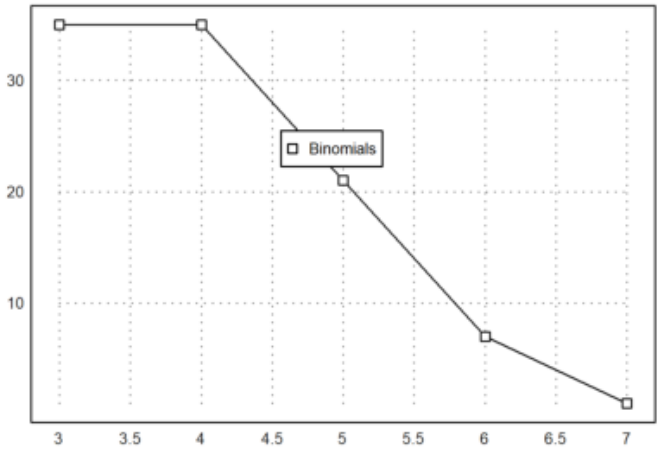


Label teks, judul, kotak label, dan teks lainnya dapat memuat string Unicode (lihat sintaks EMT untuk informasi lebih lanjut tentang string Unicode).

```
>plot2d("x^3-x",title=u"x &rarr; x&sup3; - x"):
```

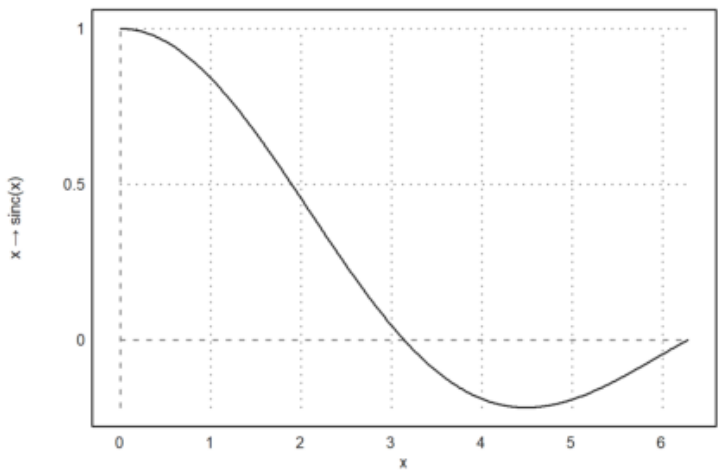



```
>n=7; plot2d(3:n,bin(n,3:n),>addpoints); ...
>labelbox("Binomials",styles="[]",>points,x=0.4,y=0.3, ...
>tcolor=black,>left):
```



Label pada sumbu x- dan y- dapat vertikal, begitu pula sumbunya.

```
>plot2d("sinc(x)",0,2pi,xl="x",yl=u"x &rarr; sinc(x)",>vertical):
```



LaTeX

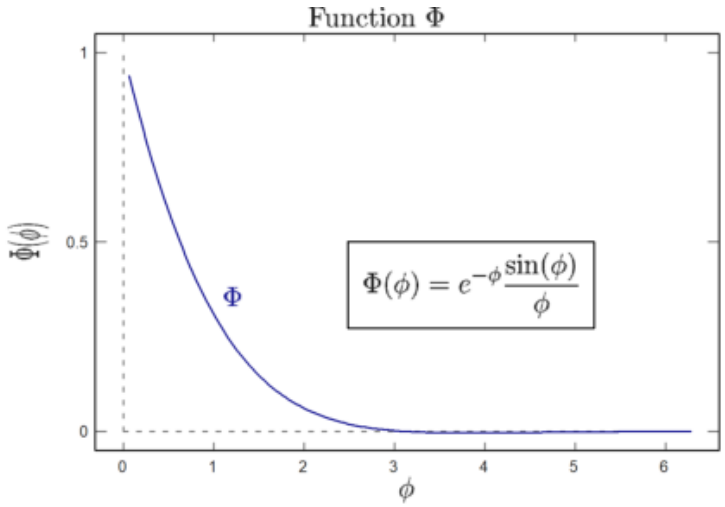
Anda juga dapat memplot rumus LaTeX jika sudah menginstal sistem LaTeX. Saya merekomendasikan MiKTeX. Path menuju binary "latex" dan "dvipng" harus ada di dalam system path, atau Anda perlu mengatur LaTeX di menu opsi.

Perlu dicatat bahwa proses parsing LaTeX cukup lambat. Jika Anda ingin menggunakan LaTeX dalam plot animasi, sebaiknya panggil latex() sekali sebelum perulangan dan gunakan hasilnya (sebuah gambar dalam matriks RGB).

Pada plot berikut, kita menggunakan LaTeX untuk label sumbu-x dan sumbu-y, sebuah label, kotak label, serta judul plot.

```
>plot2d("exp(-x)*sin(x)/x",a=0,b=2pi,c=0,d=1,grid=6,color=blue, ...
>  title=latex("\text{Function $\Phi$}"), ...
>  xl=latex("\phi"),yl=latex("\Phi(\phi)")); ...
```

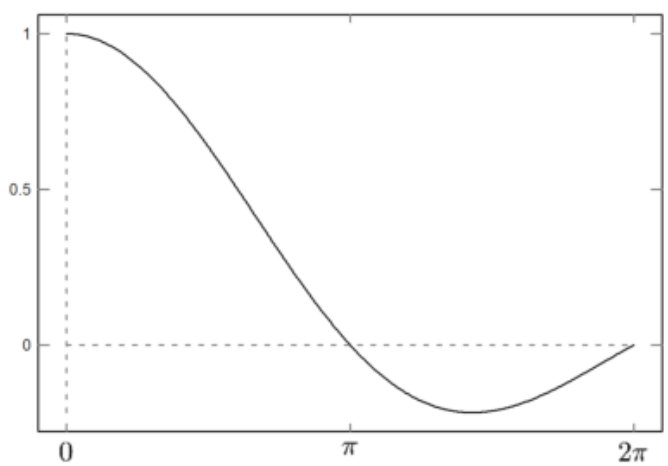
```
>textbox( ...
>  latex("\Phi(\phi) = e^{-\phi} \frac{\sin(\phi)}{\phi}"),x=0.8,y=0.5); ...
>label(latex("\Phi",color=blue),1,0.4):
```



Sering kali, kita menginginkan jarak yang tidak konformal dan label teks pada sumbu-x. Kita dapat menggunakan xaxis() dan yaxis() seperti yang akan ditunjukkan nanti.

Cara termudah adalah membuat plot kosong dengan bingkai menggunakan grid=4, lalu menambahkan grid dengan ygrid() dan xgrid(). Pada contoh berikut, kita menggunakan tiga string LaTeX untuk label pada sumbu-x dengan xtick().

```
>plot2d("sinc(x)",0,2pi,grid=4,<ticks); ...
>ygrid(-2:0.5:2,grid=6); ...
>xgrid([0:2]*pi,<ticks,grid=6);  ...
>xtick([0,pi,2pi],["0","\pi","2\pi"],>latex):
```



Tentu saja, fungsi juga dapat digunakan.

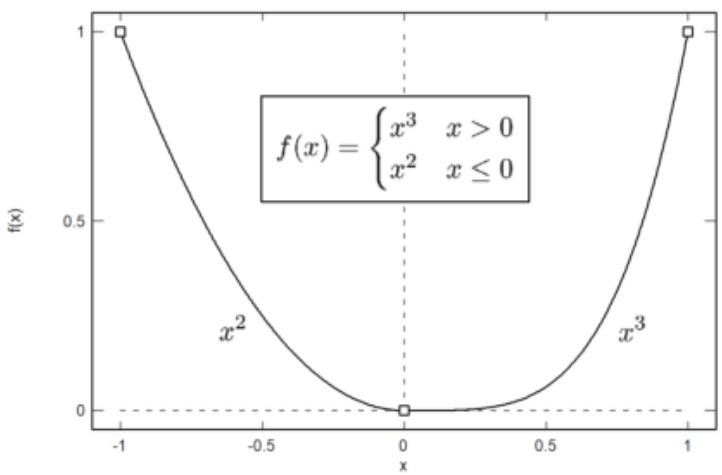
```
>function map f(x) ...
```

```
  if x>0 then return x^4
  else return x^2
  endif
  endfunction
```

Parameter "map" membantu penggunaan fungsi untuk vektor. Untuk plot, sebenarnya hal ini tidak diperlukan. Namun, untuk menunjukkan bahwa vektorisasi itu berguna, kita menambahkan beberapa titik kunci pada plot di x=-1, x=0, dan x=1.

Pada plot berikut, kita juga memasukkan beberapa kode LaTeX. Kita menggunakannya untuk dua label dan sebuah kotak teks. Tentu saja, Anda hanya dapat menggunakan LaTeX jika telah menginstalnya dengan benar.

```
>plot2d("f",-1,1,xl="x",yl="f(x)",grid=6);  ...
>plot2d([-1,0,1],f([-1,0,1]),>points,>add); ...
>label(latex("x^3"),0.72,f(0.72)); ...
>label(latex("x^2"),-0.52,f(-0.52),pos="ll"); ...
>textbox( ...
>  latex("f(x)=\begin{cases} x^3 & x>0 \\ x^2 & x \le 0\end{cases}"), ...
>  x=0.7,y=0.2):
```



Interaksi Pengguna

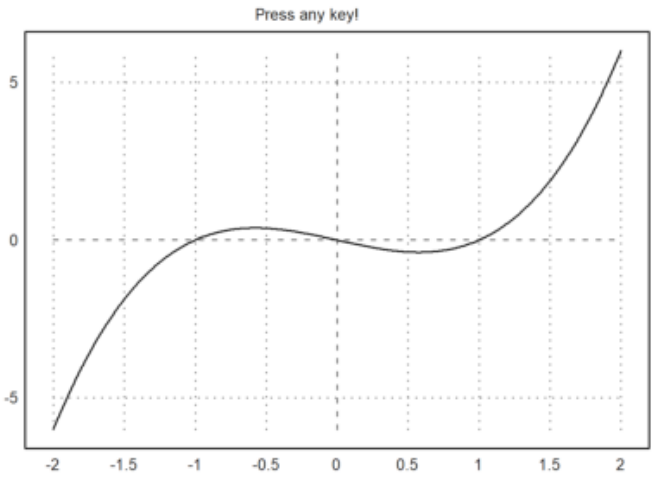
Saat memplot sebuah fungsi atau ekspresi, parameter >user memungkinkan pengguna untuk melakukan zoom dan menggeser plot dengan tombol kursor atau mouse. Pengguna dapat:

- zoom dengan tombol + atau -
- menggeser plot dengan tombol kursor
- memilih jendela plot dengan mouse
- mengatur ulang tampilan dengan tombol spasi
- keluar dengan tombol return

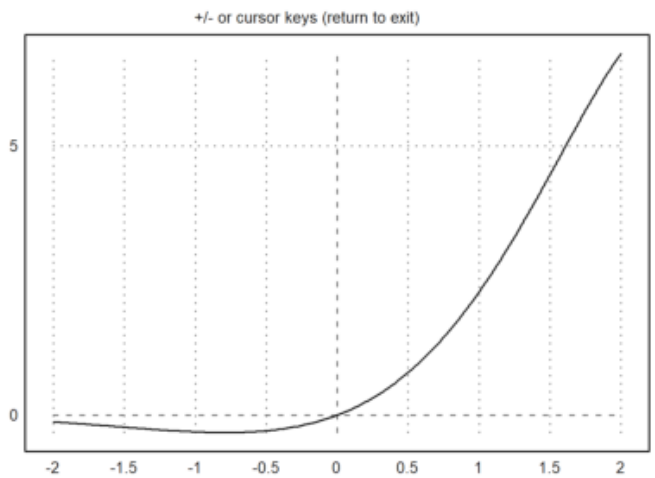
Tombol spasi akan mengatur ulang plot ke jendela plot aslinya.

Saat memplot sebuah data, flag >user hanya akan menunggu penekanan tombol.

```
>plot2d({{"x^3-a*x",a=1}},>user,title="Press any key!"):
```




```
>plot2d("exp(x)*sin(x)",user=true, ...
>  title="+/- or cursor keys (return to exit)"):
```



The following demonstrates an advanced way of user interaction (see the tutorial about programming for details).

The built-in function mousedrag() waits for mouse or keyboard events. It reports mouse down, mouse moved or mouse up, and key presses. The function dragpoints() makes use of this, and lets the user drag any point in a plot.

We need a plot function first. For an example, we interpolate in 5 points with a polynomial. The function should plot into a fixed plot area.

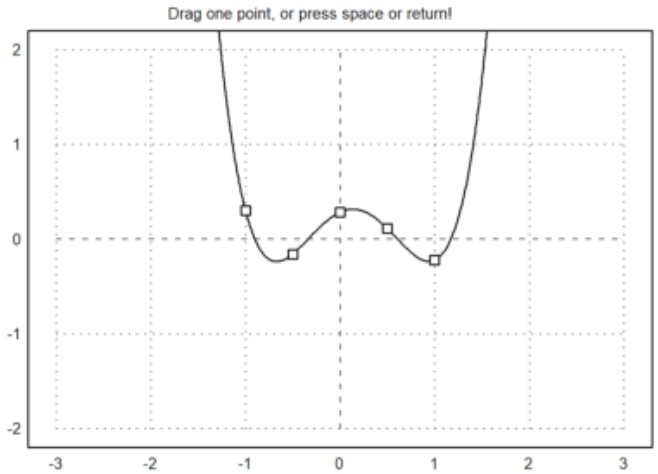
```
>function plotf(xp,yp,select) ...
```

```
    d=interp(xp,yp);
    plot2d("interpval(xp,d,x)";d,xp,r=2);
    plot2d(xp,yp,>points,>add);
    if select>0 then
      plot2d(xp[select],yp[select],color=red,>points,>add);
    endif;
    title("Drag one point, or press space or return!");
  endfunction
```

Perhatikan parameter titik koma dalam plot2d (d dan xp), yang diteruskan ke evaluasi fungsi interp(). Tanpa ini, kita harus menulis fungsi plotinterp() terlebih dahulu, dengan mengakses nilai-nilainya secara global.

Sekarang kita menghasilkan beberapa nilai acak, dan membiarkan pengguna menyeret titik-titik tersebut.

```
>t=-1:0.5:1; dragpoints("plotf",t,random(size(t))-0.5):
```



Terdapat juga sebuah fungsi yang memplot fungsi lain yang bergantung pada sebuah vektor parameter, dan memungkinkan pengguna untuk menyesuaikan parameter-parameter tersebut.

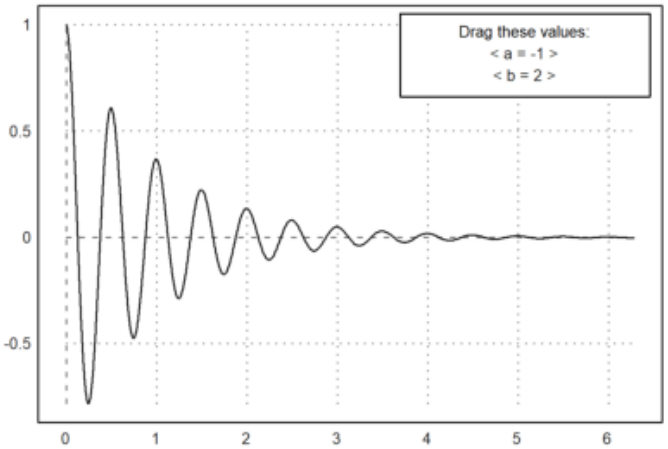
Pertama, kita memerlukan fungsi plotnya.

```
>function plotf([a,b]) := plot2d("exp(a*x)*cos(2pi*b*x)",0,2pi;a,b);
```

Selanjutnya, kita memerlukan nama-nama untuk parameter, nilai awal, dan sebuah matriks nx2 dari rentang nilai, serta opsional sebuah baris judul.

Terdapat slider interaktif yang memungkinkan pengguna mengatur nilai-nilai tersebut. Fungsi dragvalues() menyediakan fitur ini.

```
>dragvalues("plotf",["a","b"],[-1,2],[[-2,2];[1,10]], ...
>  heading="Drag these values:",hcolor=black):
```



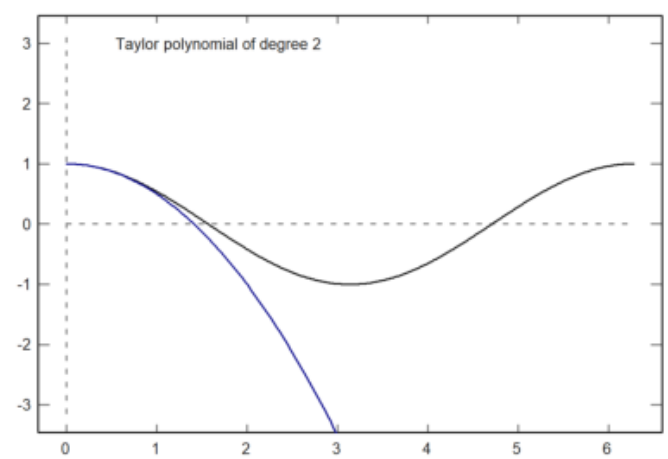
Dimungkinkan untuk membatasi nilai yang diseret agar berupa bilangan bulat. Sebagai contoh, kita menulis sebuah fungsi plot yang memplot polinomial Taylor derajat n untuk fungsi kosinus.

```
>function plotf(n) ...
```

```
  plot2d("cos(x)",0,2pi,>square,grid=6);
  plot2d(&"taylor(cos(x),x,0,@n)",color=blue,>add);
  textbox("Taylor polynomial of degree "+n,0.1,0.02,style="t",>left);
  endfunction
```

Sekarang kita membiarkan derajat n bervariasi dari 0 hingga 20 dalam 20 langkah. Hasil dari dragvalues() digunakan untuk memplot sketsa dengan n tersebut, dan untuk memasukkan plot ke dalam notebook.

```
>nd=dragvalues("plotf","degree",2,[0,20],20,y=0.8, ...
>   heading="Drag the value:"); ...
>plotf(nd):
```



Berikut adalah demonstrasi sederhana dari fungsi ini. Pengguna dapat menggambar di atas jendela plot, meninggalkan jejak titik-titik.

```
>function dragtest ...
```

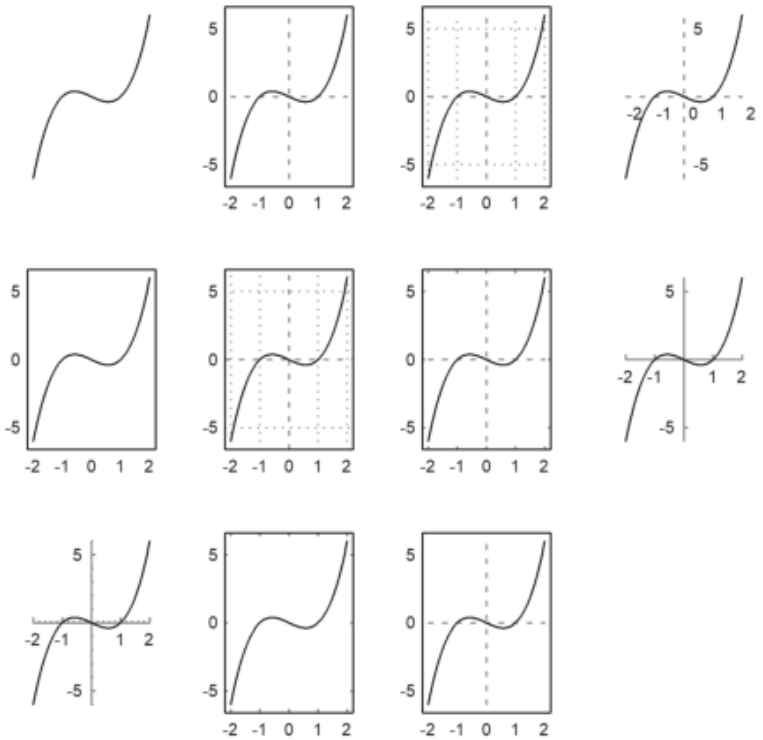
```
    plot2d(none,r=1,title="Drag with the mouse, or press any key!");
    start=0;
    repeat
      {flag,m,time}=mousedrag();
      if flag==0 then return; endif;
      if flag==2 then
        hold on; mark(m[1],m[2]); hold off;
      endif;
    end
  endfunction
```

```
>dragtest // lihat hasilnya dan cobalah lakukan!
```

Gaya Plot 2D

Secara bawaan, EMT menghitung tanda sumbu (axis ticks) secara otomatis dan menambahkan label pada setiap tanda. Hal ini dapat diubah dengan parameter grid. Gaya bawaan dari sumbu dan label dapat dimodifikasi. Selain itu, label dan judul dapat ditambahkan secara manual. Untuk mengembalikan gaya ke pengaturan bawaan, gunakan reset().

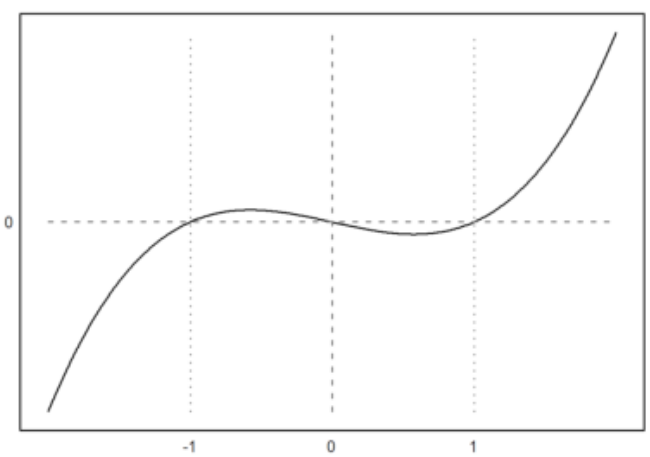
```
>aspect();
>figure(3,4); ...
> figure(1); plot2d("x^3-x",grid=0); ... // no grid, frame or axis
> figure(2); plot2d("x^3-x",grid=1); ... // x-y-axis
> figure(3); plot2d("x^3-x",grid=2); ... // default ticks
> figure(4); plot2d("x^3-x",grid=3); ... // x-y- axis with labels inside
> figure(5); plot2d("x^3-x",grid=4); ... // no ticks, only labels
> figure(6); plot2d("x^3-x",grid=5); ... // default, but no margin
> figure(7); plot2d("x^3-x",grid=6); ... // axes only
> figure(8); plot2d("x^3-x",grid=7); ... // axes only, ticks at axis
> figure(9); plot2d("x^3-x",grid=8); ... // axes only, finer ticks at axis
> figure(10); plot2d("x^3-x",grid=9); ... // default, small ticks inside
> figure(11); plot2d("x^3-x",grid=10); ...// no ticks, axes only
> figure(0):
```



Parameter <frame mematikan bingkai, dan framecolor=blue mengatur bingkai menjadi warna biru.

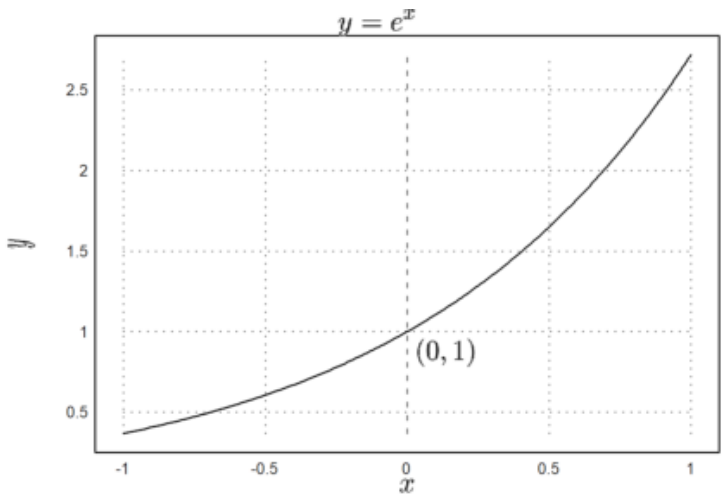
Jika Anda ingin menggunakan tanda sumbu sendiri, Anda dapat menggunakan style=0, lalu menambahkan semuanya nanti.

```
>aspect(1.5);
>plot2d("x^3-x",grid=0); // plot
>frame; xgrid([-1,0,1]); ygrid(0): // add frame and grid
```



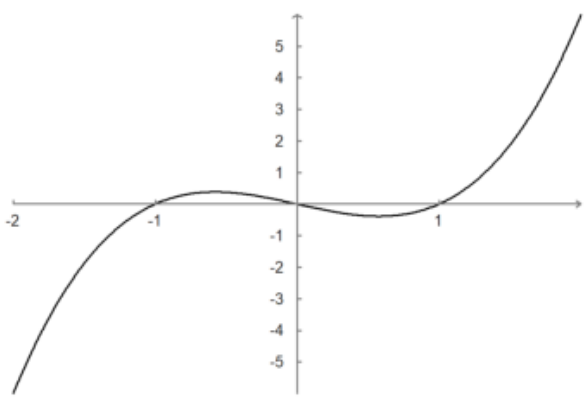
Untuk judul plot dan label sumbu, lihat contoh berikut.

```
>plot2d("exp(x)",-1,1);
>textcolor(black); // set the text color to black
>title(latex("y=e^x")); // title above the plot
>xlabel(latex("x")); // "x" for x-axis
>ylabel(latex("y"),>vertical); // vertical "y" for y-axis
>label(latex("(0,1)"),0,1,color=blue): // label a point
```



Sumbu dapat digambar secara terpisah menggunakan xaxis() dan yaxis().

```
>plot2d("x^3-x",<grid,<frame);
>xaxis(0,xx=-2:1,style="->"); yaxis(0,yy=-5:5,style="->"):
```

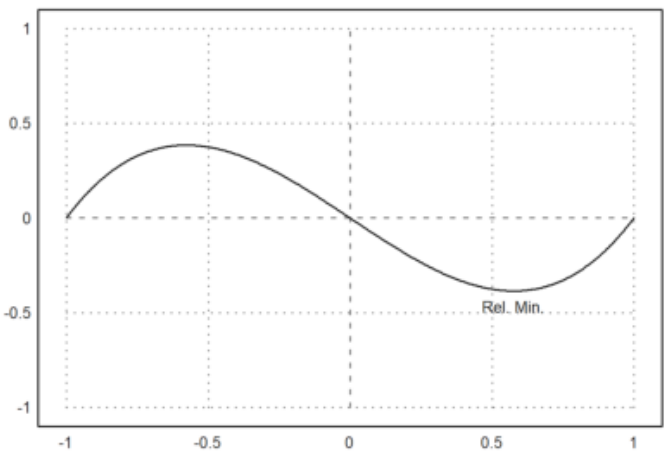


Teks pada plot dapat diatur dengan label(). Pada contoh berikut, "lc" berarti lower center (tengah bawah). Ini mengatur posisi label relatif terhadap koordinat plot.

```
>function f(x) &= x^3-x
```

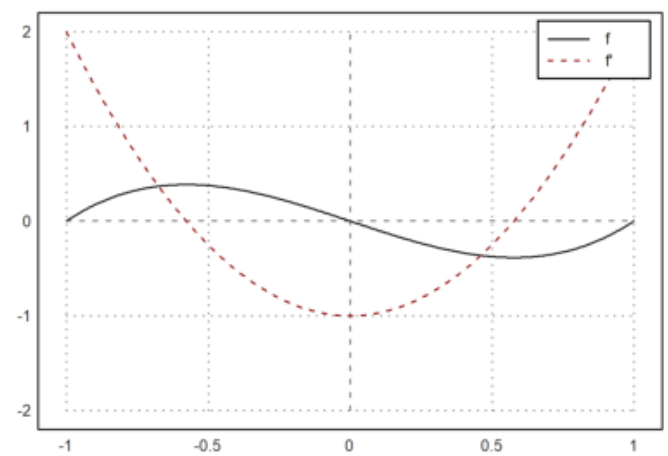
$$x^3 - x$$

```
>plot2d(f,-1,1,>square);
>x0=fmin(f,0,1); // compute point of minimum
>label("Rel. Min.",x0,f(x0),pos="lc"): // add a label there
```

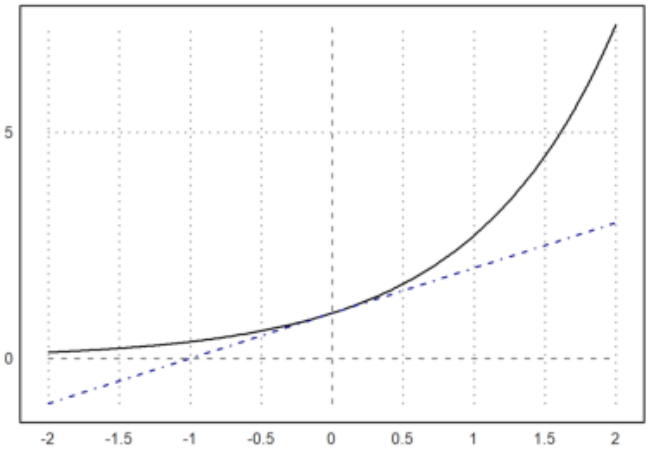


Ada juga kotak teks.

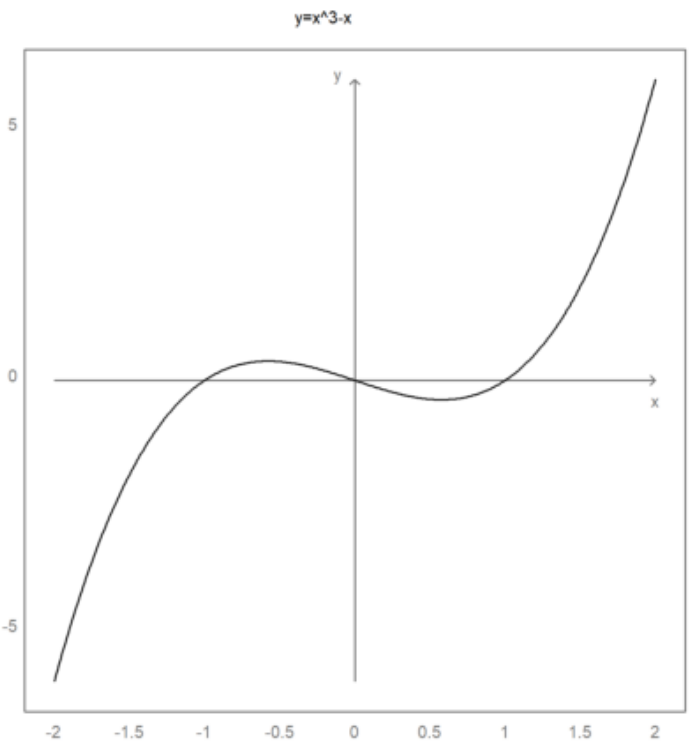
```
>plot2d(&f(x),-1,1,-2,2); // function
>plot2d(&diff(f(x),x),>add,style="--",color=red); // derivative
>labelbox(["f","f'"],["-","--"],[black,red]): // label box
```




```
>plot2d(["exp(x)","1+x"],color=[black,blue],style=["-","-.-"]):
```



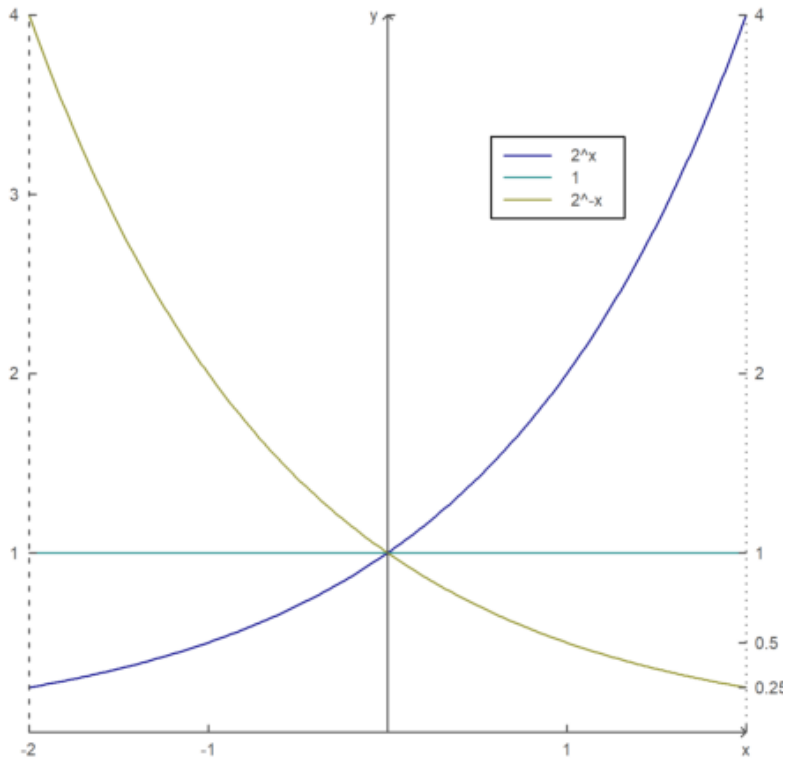

```
>gridstyle("->",color=gray,textcolor=gray,framecolor=gray);  ...
> plot2d("x^3-x",grid=1);   ...
> settitle("y=x^3-x",color=black); ...
> label("x",2,0,pos="bc",color=gray);  ...
> label("y",0,6,pos="cl",color=gray); ...
> reset():
```



Untuk kontrol yang lebih, sumbu-x dan sumbu-y dapat dibuat secara manual.

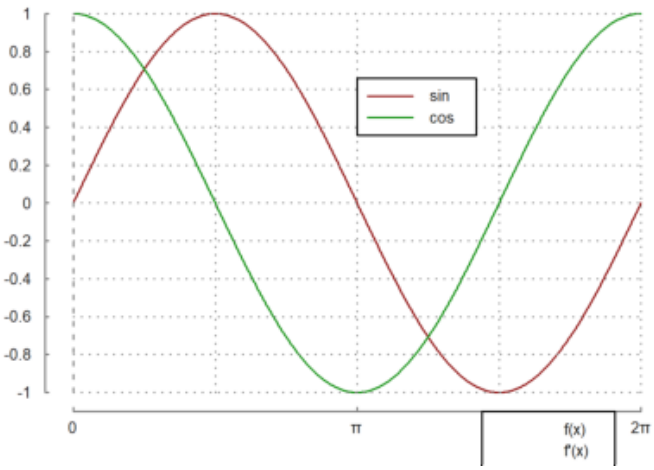
Perintah fullwindow() memperluas jendela plot karena kita tidak lagi memerlukan tempat untuk label di luar jendela plot. Gunakan shrinkwindow() atau reset() untuk mengembalikan ke pengaturan bawaan.

```
>fullwindow; ...
> gridstyle(color=darkgray,textcolor=darkgray); ...
> plot2d(["2^x","1","2^(-x)"],a=-2,b=2,c=0,d=4,<grid,color=4:6,<frame); ...
> xaxis(0,-2:1,style="->"); xaxis(0,2,"x",<axis); ...
> yaxis(0,4,"y",style="->"); ...
> yaxis(-2,1:4,>left); ...
> yaxis(2,2^(-2:2),style=".",<left); ...
> labelbox(["2^x","1","2^-x"],colors=4:6,x=0.8,y=0.2); ...
> reset:
```



Berikut contoh lain, di mana string Unicode digunakan dan sumbu berada di luar area plot.

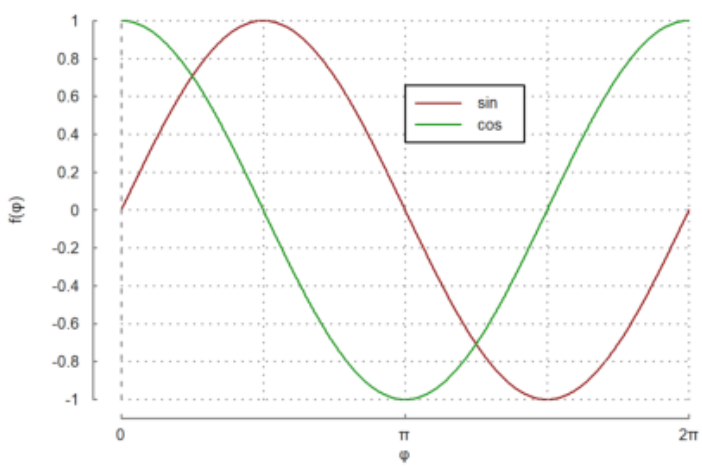
```
>aspect(1.5);
>labelbox(["f(x)","f'(x)"], ["-","--"], [blue,red], x=0.7, y=1, >left):
```




```
>plot2d(["sin(x)","cos(x)"],0,2pi,color=[red,green],<grid,<frame); ...
> xaxis(-1.1,(0:2)*pi,xt=["0",u"&pi;",u"2&pi;"],style="-",>ticks,>zero);  ...
> xgrid((0:0.5:2)*pi,<ticks); ...
> yaxis(-0.1*pi,-1:0.2:1,style="-",>zero,>grid); ...
> labelbox(["sin","cos"],colors=[red,green],x=0.5,y=0.2,>left); ...
```

```
  Syntax error in expression, or unfinished expression!
  Error in:
  ... n","cos"],colors=[red,green],x=0.5,y=0.2,>left); >labelbox(["f ...
                                                       ^
```

```
> xlabel(u"&phi;"); ylabel(u"f(&phi;)"):
```



Plot Data 2D

Jika x dan y adalah vektor data, data ini akan digunakan sebagai koordinat x dan y dari sebuah kurva. Dalam hal ini, a, b, c, dan d, atau radius r dapat ditentukan, atau jendela plot akan menyesuaikan secara otomatis dengan data. Sebagai alternatif, >square dapat digunakan untuk menjaga rasio aspek kotak.

Memplot sebuah ekspresi hanyalah singkatan untuk plot data. Untuk plot data, Anda memerlukan satu atau lebih baris nilai x, dan satu atau lebih baris nilai y. Dari rentang dan nilai x, fungsi plot2d akan menghitung

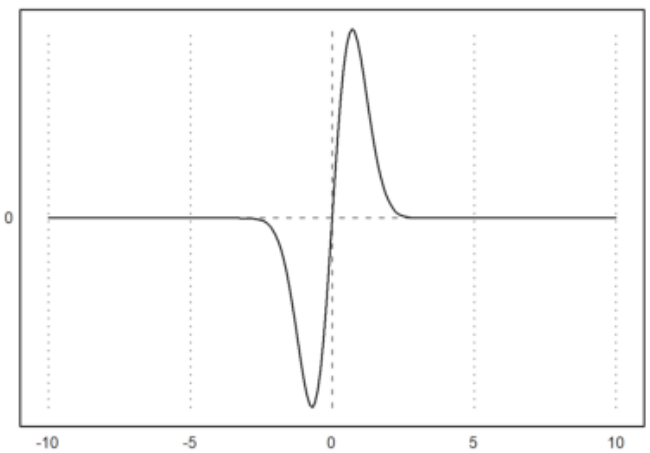
data yang akan diplot, secara bawaan dengan evaluasi adaptif dari fungsi tersebut. Untuk plot titik gunakan ">points", untuk kombinasi garis dan titik gunakan ">addpoints".

Namun, Anda juga dapat memasukkan data secara langsung.

- Gunakan vektor baris untuk x dan y untuk satu fungsi.
- Matriks untuk x dan y akan diplot baris demi baris.

Berikut adalah contoh dengan satu baris untuk x dan y.

```
>x=-10:0.1:10; y=exp(-x^2)*x; plot2d(x,y):
```

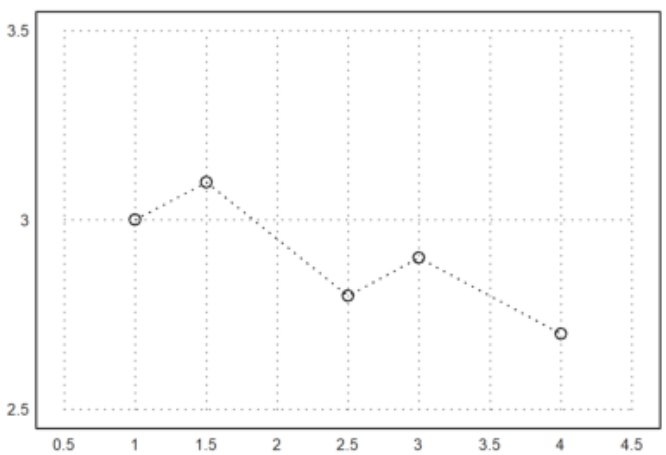


Data juga dapat diplot sebagai titik. Gunakan points=true untuk ini. Plot ini bekerja seperti poligon, tetapi hanya menggambar sudutnya.

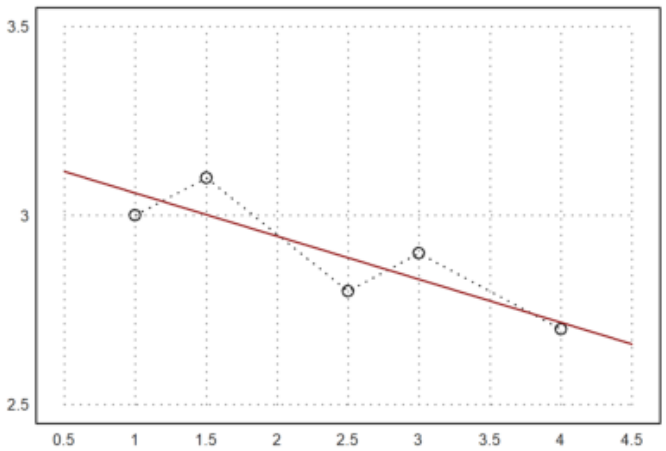
- style="...": Pilih dari "[]", "<>", "o", ".", "..", "+", "*", "[]", "<>", "o", "..", "", "|".

Untuk memplot kumpulan titik, gunakan >points. Jika warnanya adalah vektor warna, setiap titik akan mendapatkan warna yang berbeda. Untuk matriks koordinat dan vektor kolom, warna tersebut berlaku untuk baris-baris matriks.
Parameter >addpoints menambahkan titik ke segmen garis untuk plot data.

```
>xdata=[1,1.5,2.5,3,4]; ydata=[3,3.1,2.8,2.9,2.7]; // data
>plot2d(xdata,ydata,a=0.5,b=4.5,c=2.5,d=3.5,style="."); // lines
>plot2d(xdata,ydata,>points,>add,style="o"): // add points
```



```
>p=polyfit(xdata,ydata,1); // get regression line
>plot2d("polyval(p,x)",>add,color=red): // add plot of line
```



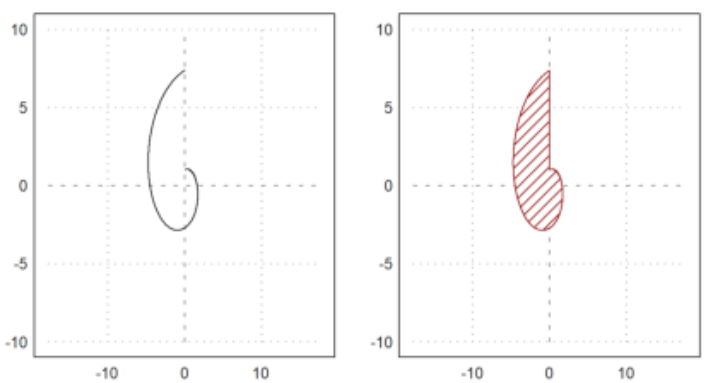
Menggambar Daerah yang Dibatasi Kurva

Plot data sebenarnya adalah poligon. Kita juga dapat memplot kurva atau kurva yang terisi.

- filled=true mengisi plot.
- style="...": Pilih dari "", "/", "\", "\/".
- fillcolor: Lihat di atas untuk warna yang tersedia.

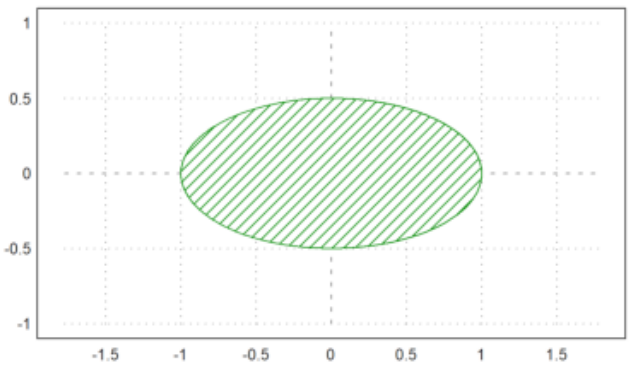
Warna isian ditentukan oleh argumen "fillcolor", dan pada <outline opsional mencegah penggambaran batas untuk semua gaya kecuali yang default.

```
>t=linspace(0,2pi,1000); // parameter for curve
>x=sin(t)*exp(t/pi); y=cos(t)*exp(t/pi); // x(t) and y(t)
>figure(1,2); aspect(16/9)
>figure(1); plot2d(x,y,r=10); // plot curve
>figure(2); plot2d(x,y,r=10,>filled,style="/",fillcolor=red); // fill curve
>figure(0):
```

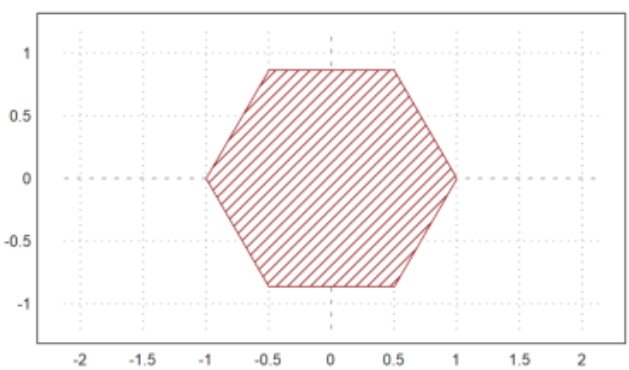


Pada contoh berikut, kita memplot sebuah elips terisi dan dua heksagon terisi menggunakan kurva tertutup dengan 6 titik dengan gaya isian (fill style) yang berbeda.

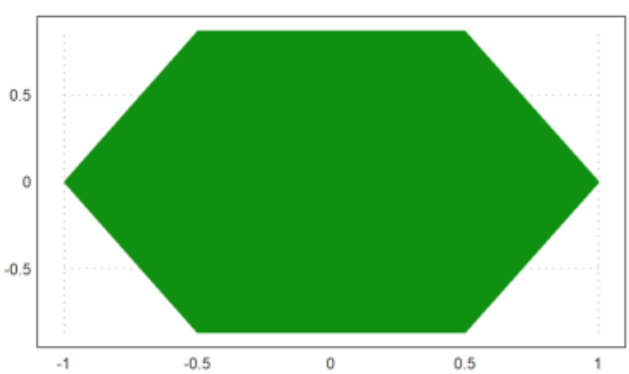
```
>x=linspace(0,2pi,1000); plot2d(sin(x),cos(x)*0.5,r=1,>filled,style="/"):
```




```
>t=linspace(0,2pi,6); ...
>plot2d(cos(t),sin(t),>filled,style="/",fillcolor=red,r=1.2):
```

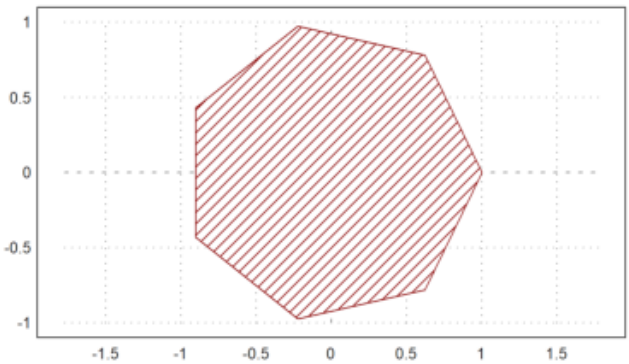



```
>t=linspace(0,2pi,6); plot2d(cos(t),sin(t),>filled,style="#"):
```



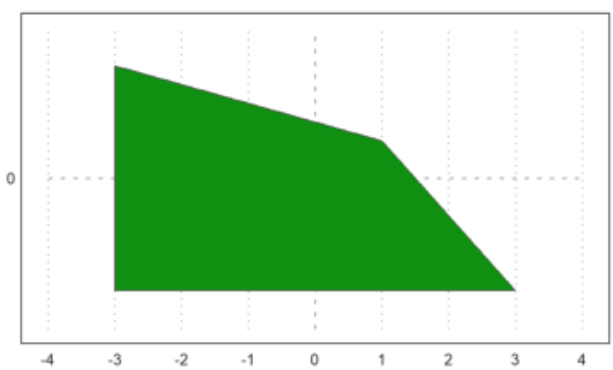
Contoh lain adalah sebuah heptagon, yang kita buat dengan 7 titik pada lingkaran satuan.

```
>t=linspace(0,2pi,7);  ...
> plot2d(cos(t),sin(t),r=1,>filled,style="/",fillcolor=red):
```



Berikut ini adalah himpunan nilai maksimal dari empat kondisi linier yang kurang dari atau sama dengan 3. Ini adalah A[k].v<=3 untuk semua baris A. Untuk mendapatkan sudut yang bagus, kita menggunakan n yang relatif besar.

```
>A=[2,1;1,2;-1,0;0,-1];
>function f(x,y) := max([x,y].A');
>plot2d("f",r=4,level=[0;3],color=green,n=111):
```

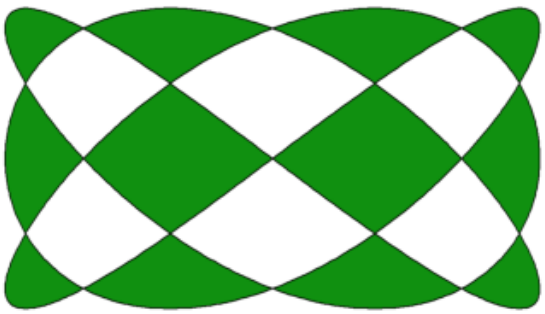


Inti dari bahasa matriks adalah bahwa ia memungkinkan pembuatan tabel fungsi dengan mudah.

```
>t=linspace(0,2pi,1000); x=cos(3*t); y=sin(4*t);
```

Sekarang kita memiliki vektor x dan y dari nilai-nilai. plot2d() dapat memplot nilai-nilai ini sebagai kurva yang menghubungkan titik-titik tersebut. Plot ini dapat diisi (filled). Dalam hal ini, hasilnya akan terlihat baik karena aturan winding digunakan untuk pengisian.

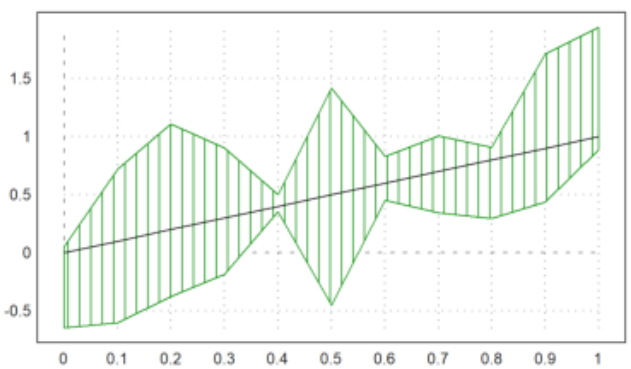
```
>plot2d(x,y,<grid,<frame,>filled):
```



Sebuah vektor interval diplot terhadap nilai x sebagai sebuah wilayah terisi (filled region) antara nilai bawah dan atas dari interval tersebut.

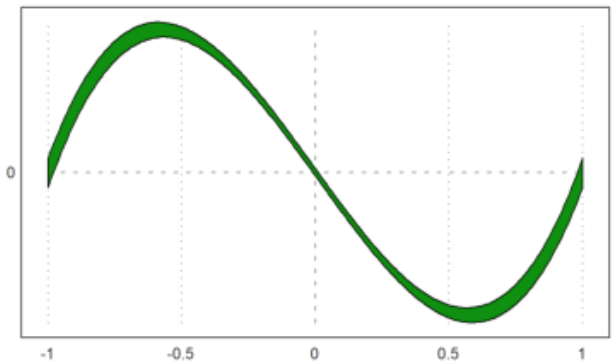
Hal ini bisa berguna untuk memplot kesalahan dari perhitungan. Namun, ini juga dapat digunakan untuk memplot kesalahan statistik.

```
>t=0:0.1:1; ...
> plot2d(t,interval(t-random(size(t)),t+random(size(t))),style="|");  ...
> plot2d(t,t,add=true):
```



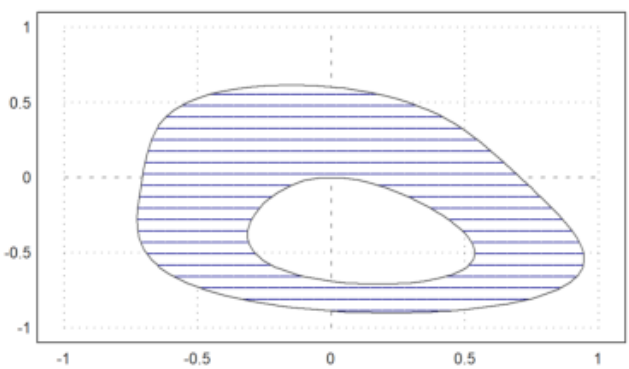
Jika x adalah vektor yang diurutkan, dan y adalah vektor interval, maka plot2d akan memplot rentang interval yang terisi pada bidang. Gaya isiannya sama dengan gaya poligon.

```
>t=-1:0.01:1; x=~t-0.01,t+0.01~; y=x^3-x;
>plot2d(t,y):
```



Dimungkinkan untuk mengisi wilayah nilai untuk fungsi tertentu.
Untuk ini, level harus berupa matriks 2xn. Baris pertama adalah batas bawah, dan baris kedua berisi batas atas.

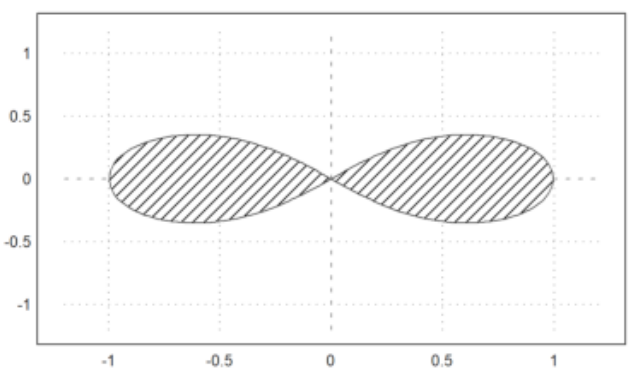
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=[0;1],style="-",color=blue): // 0 <= f(x,y) <= 1
```



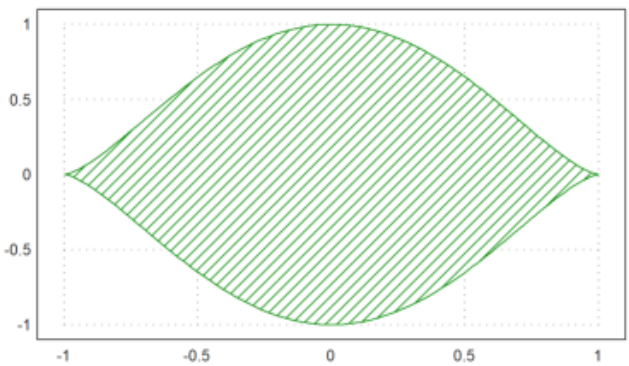
Kita juga dapat mengisi rentang nilai seperti

$$-1 \le (x^2+y^2)^2 - x^2 + y^2 \le 0.$$

```
>plot2d("(x^2+y^2)^2-x^2+y^2",r=1.2,level=[-1;0],style="/"):
```



```
>plot2d("cos(x)","sin(x)^3",xmin=0,xmax=2pi,>filled,style="/"):
```



Grafik Fungsi Parametrik

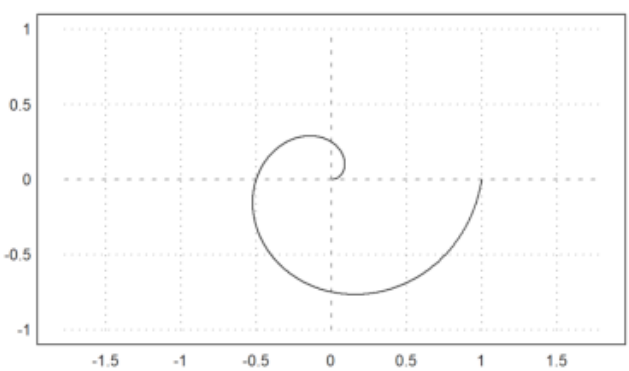
Nilai-nilai x tidak perlu diurutkan. (x,y) hanya menggambarkan sebuah kurva. Jika x diurutkan, kurva tersebut merupakan grafik suatu fungsi.

Dalam contoh berikut, kita memplot spiral

$$\gamma(t) = t \cdot (\cos(2\pi t), \sin(2\pi t))$$

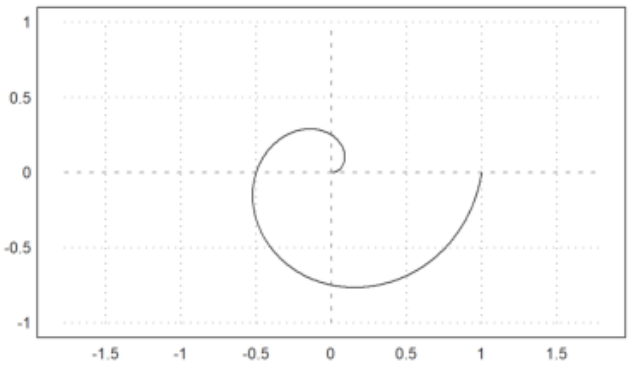
Kita perlu menggunakan banyak titik untuk tampilan yang halus atau fungsi adaptif() untuk mengevaluasi ekspresi (lihat fungsi adaptif() untuk detail lebih lanjut).

```
>t=linspace(0,1,1000); ...
>plot2d(t*cos(2*pi*t),t*sin(2*pi*t),r=1):
```

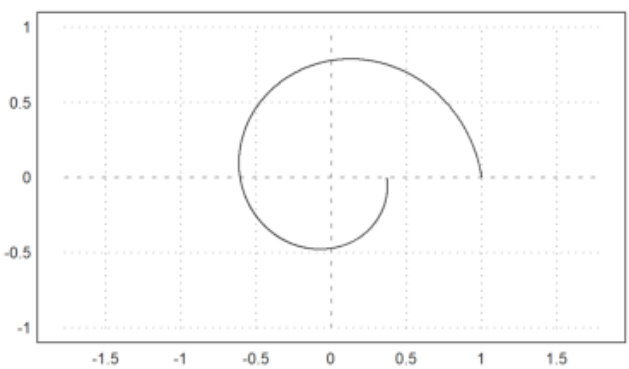


Sebagai alternatif, dimungkinkan untuk menggunakan dua ekspresi untuk kurva. Berikut ini memplot kurva yang sama seperti di atas.

```
>plot2d("x*cos(2*pi*x)","x*sin(2*pi*x)",xmin=0,xmax=1,r=1):
```




```
>t=linspace(0,1,1000); r=exp(-t); x=r*cos(2pi*t); y=r*sin(2pi*t);
>plot2d(x,y,r=1):
```



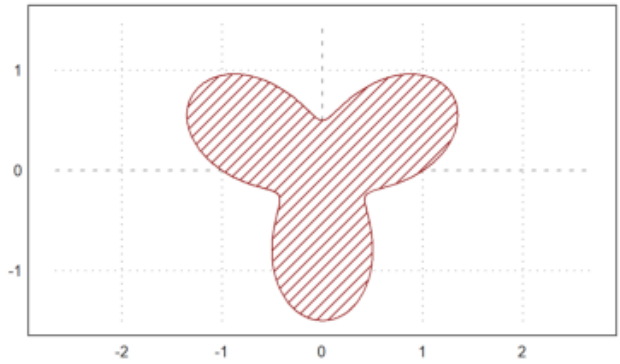
Pada contoh berikutnya, kita plot kurva

$$\gamma(t) = (r(t)\cos(t), r(t)\sin(t))$$

dengan

$$r(t) = 1 + \frac{\sin(3t)}{2}.$$

```
>t=linspace(0,2pi,1000); r=1+sin(3*t)/2; x=r*cos(t); y=r*sin(t); ...
>plot2d(x,y,>filled,fillcolor=red,style="/",r=1.5):
```



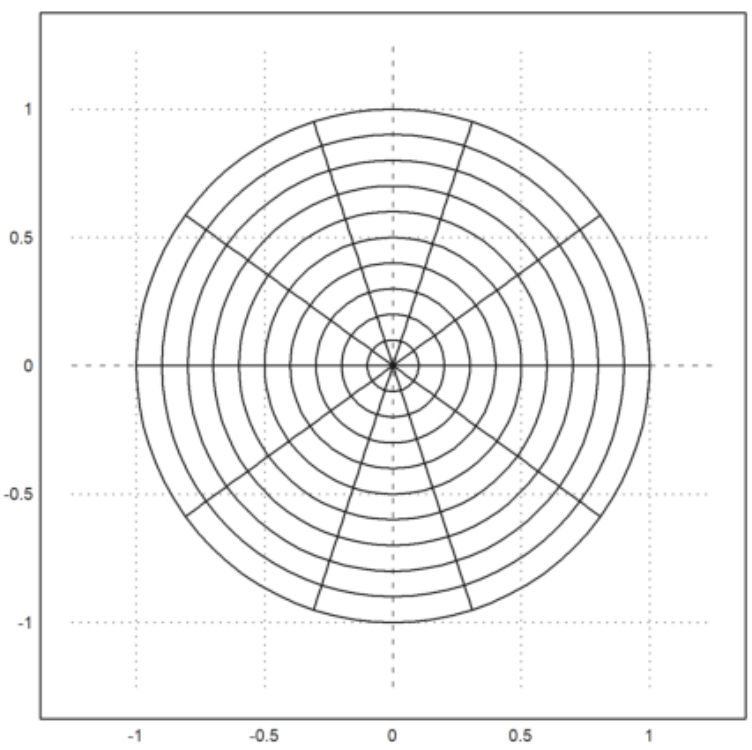
Menggambar Grafik Bilangan Kompleks

Array bilangan kompleks juga dapat diplot. Kemudian, titik-titik grid akan terhubung. Jika sejumlah garis grid ditentukan (atau vektor garis grid 1x2) dalam argumen cgrid, hanya garis grid tersebut yang terlihat.

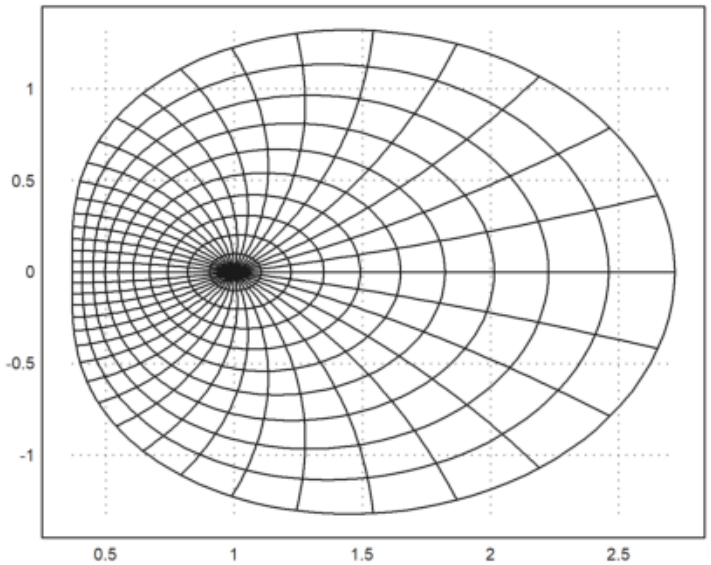
Matriks bilangan kompleks akan secara otomatis diplot sebagai grid pada bidang kompleks.

Dalam contoh berikut, kami memplot gambar lingkaran satuan di bawah fungsi eksponensial. Parameter cgrid menyembunyikan beberapa kurva grid.

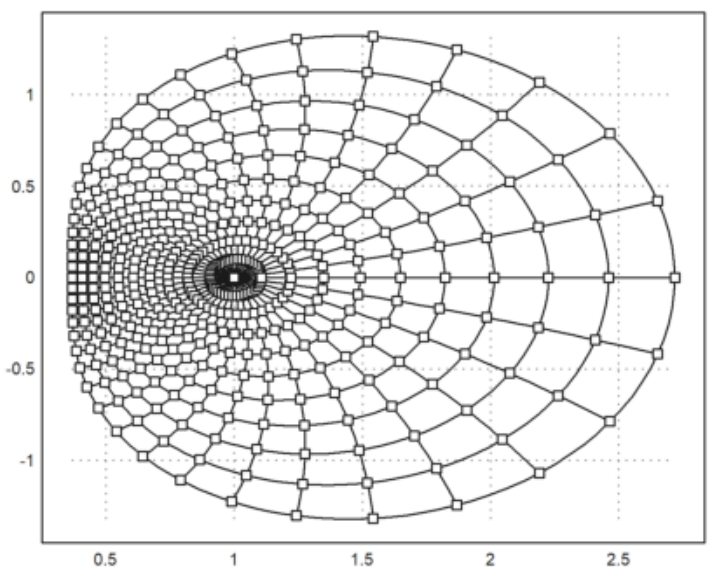
```
>aspect(); r=linspace(0,1,50); a=linspace(0,2pi,80)'; z=r*exp(I*a);...
>plot2d(z,a=-1.25,b=1.25,c=-1.25,d=1.25,cgrid=10):
```




```
>aspect(1.25); r=linspace(0,1,50); a=linspace(0,2pi,200)'; z=r*exp(I*a);
>plot2d(exp(z),cgrid=[40,10]):
```




```
>r=linspace(0,1,10); a=linspace(0,2pi,40)'; z=r*exp(I*a);
>plot2d(exp(z),>points,>add):
```

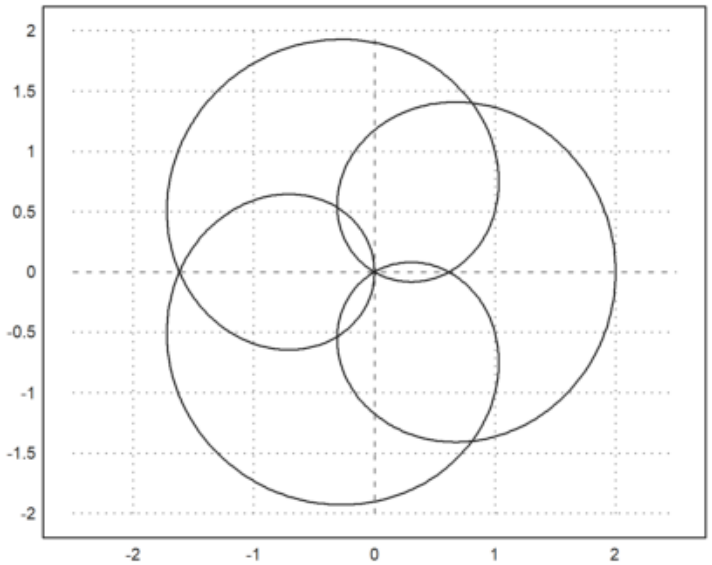


Sebuah vektor bilangan kompleks secara otomatis diplot sebagai kurva pada bidang kompleks dengan bagian real dan bagian imajiner.

Pada contoh ini, kita memplot lingkaran satuan dengan

$$\gamma(t) = e^{it}$$

```
>t=linspace(0,2pi,1000); ...
>plot2d(exp(I*t)+exp(4*I*t),r=2):
```



Plot Statistik

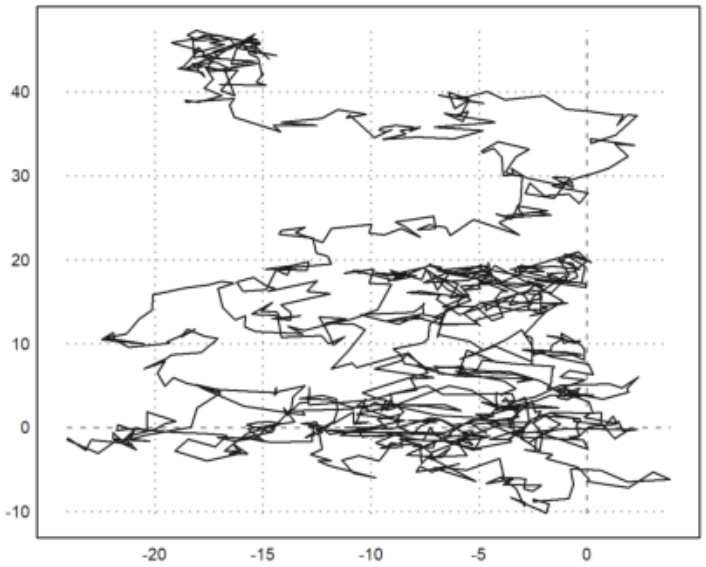
Ada banyak fungsi yang dikhususkan untuk plot statistik. Salah satu plot yang sering digunakan adalah plot kolom.

Penjumlahan kumulatif dari nilai-nilai berdistribusi normal 0-1 menghasilkan pergerakan acak (random walk).

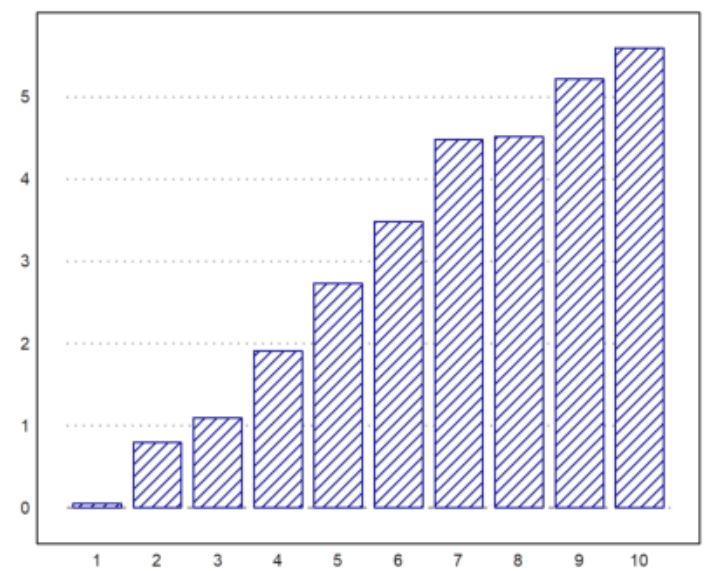
```
>plot2d(cumsum(randnormal(1,1000))):
```

Menggunakan dua baris menunjukkan sebuah lintasan dalam dua dimensi.

```
>X=cumsum(randnormal(2,1000)); plot2d(X[1],X[2]):
```

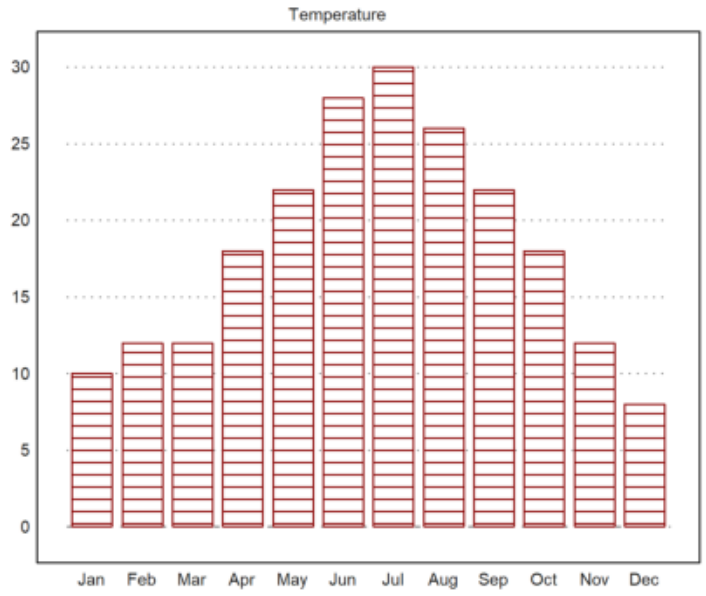



```
>columnsplot(cumsum(random(10)),style="/",color=blue):
```

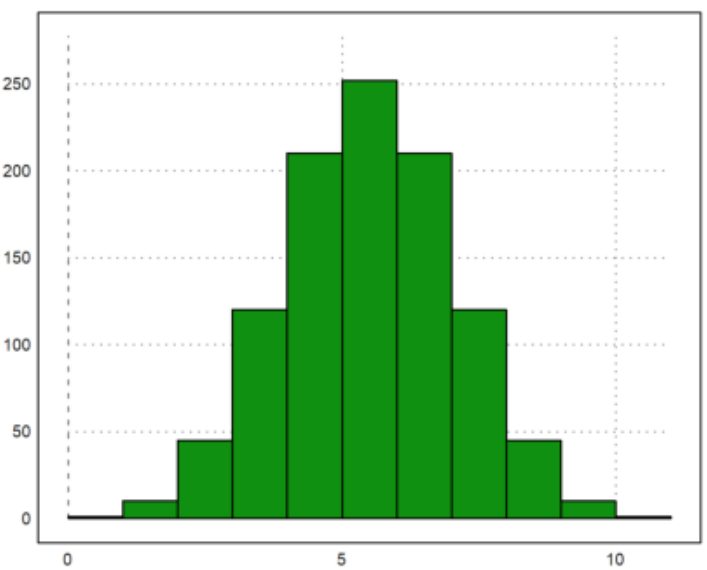


Ini juga dapat menampilkan string sebagai label.

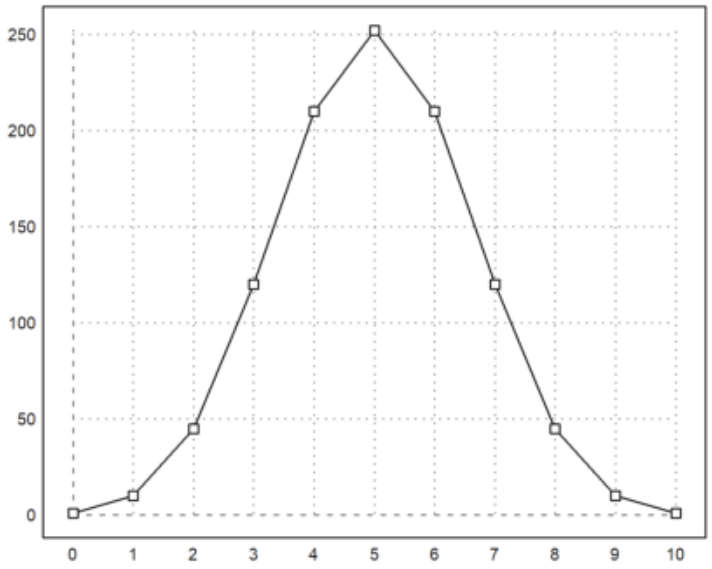
```
>months=["Jan","Feb","Mar","Apr","May","Jun", ...
>  "Jul","Aug","Sep","Oct","Nov","Dec"];
>values=[10,12,12,18,22,28,30,26,22,18,12,8];
>columnsplot(values,lab=months,color=red,style="-");
>title("Temperature"):
```



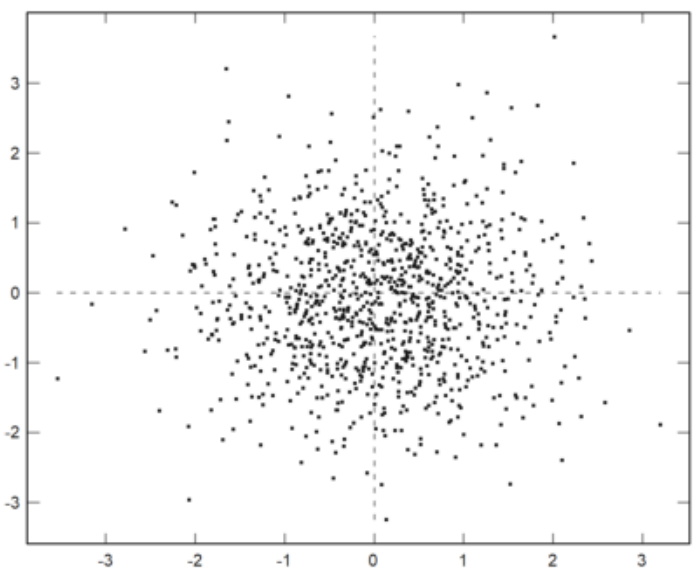

```
>k=0:10;
>plot2d(k,bin(10,k),>bar):
```



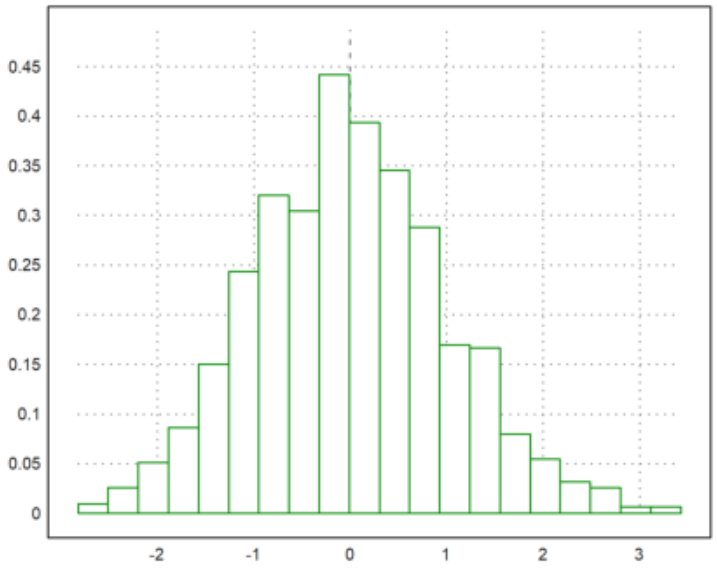

```
>plot2d(k,bin(10,k)); plot2d(k,bin(10,k),>points,>add):
```



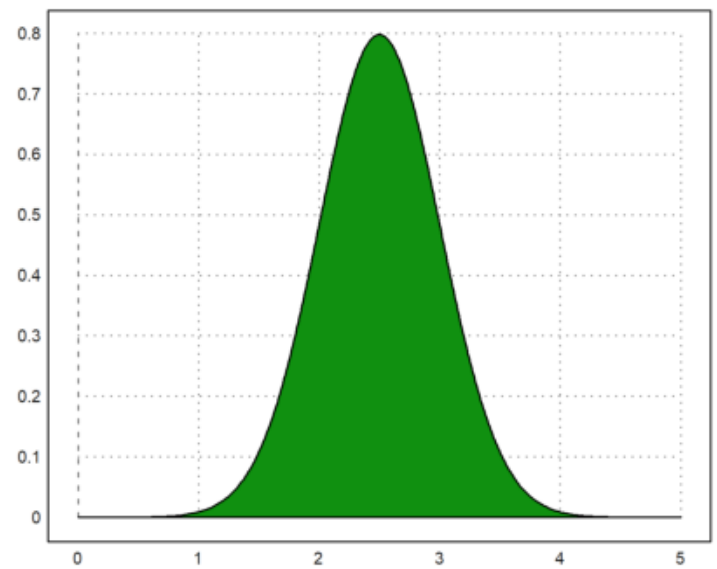

```
>plot2d(normal(1000),normal(1000),>points,grid=6,style="..");
```




```
>plot2d(normal(1,1000),>distribution,style="O");
```

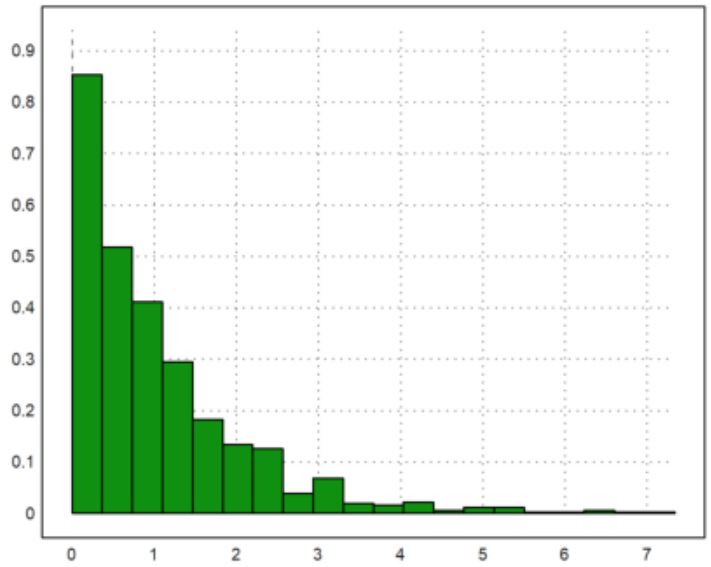



```
>plot2d("qnormal",0,5;2.5,0.5,>filled):
```



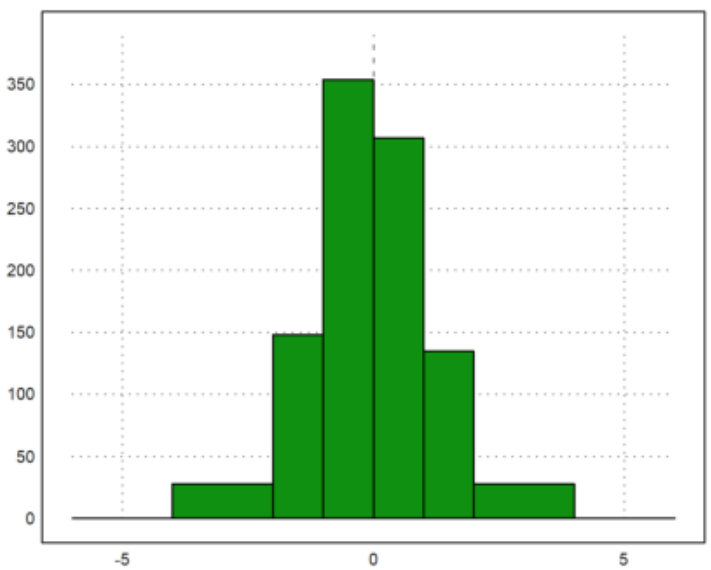
Untuk memplot distribusi statistik eksperimental, Anda dapat menggunakan distribution=n dengan plot2d.

```
>w=randexponential(1,1000); // exponential distribution
>plot2d(w,>distribution): // or distribution=n with n intervals
```



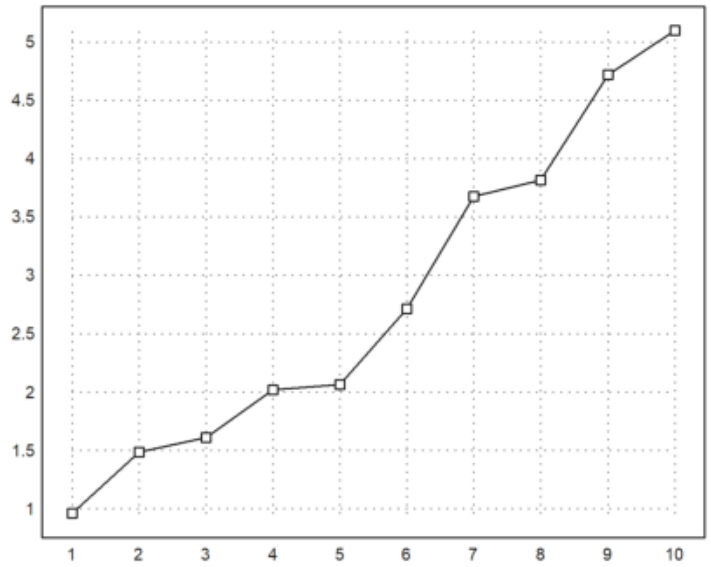
Atau Anda dapat menghitung distribusi dari data dan memplot hasilnya dengan >bar dalam plot3d, atau dengan plot kolom.

```
>w=normal(1000); // 0-1-normal distribution
>{x,y}=histo(w,10,v=[-6,-4,-2,-1,0,1,2,4,6]); // interval bounds v
>plot2d(x,y,>bar):
```

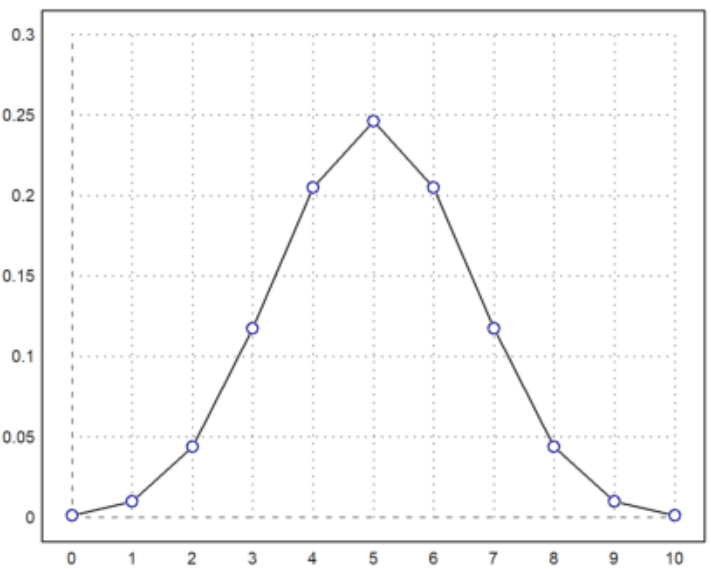


Fungsi statplot() mengatur gaya dengan sebuah string sederhana.

```
>statplot(1:10,cumsum(random(10)),"b"):
```



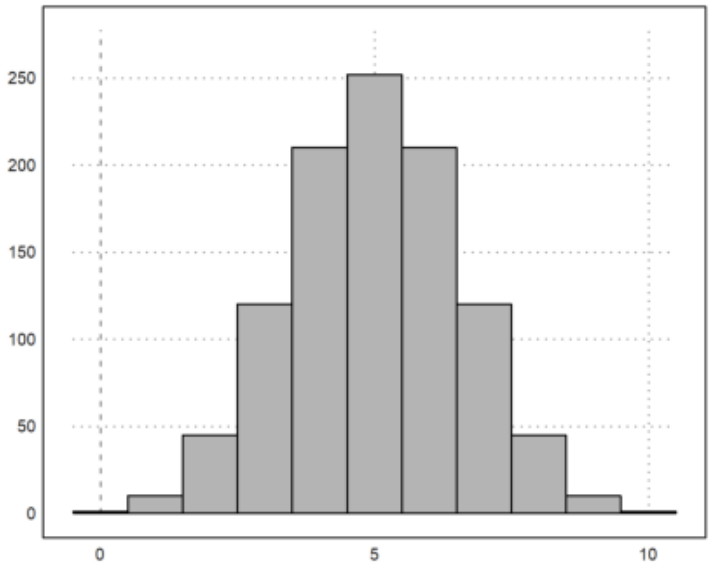

```
>n=10; i=0:n; ...
>plot2d(i,bin(n,i)/2^n,a=0,b=10,c=0,d=0.3); ...
>plot2d(i,bin(n,i)/2^n,points=true,style="ow",add=true,color=blue):
```



Selain itu, data dapat diplot sebagai batang. Dalam hal ini, x harus diurutkan dan memiliki satu elemen lebih banyak daripada y. Batang akan memanjang dari x[i] hingga x[i+1] dengan nilai y[i]. Jika x memiliki ukuran yang sama dengan y, maka x akan diperpanjang dengan satu elemen menggunakan jarak terakhir.

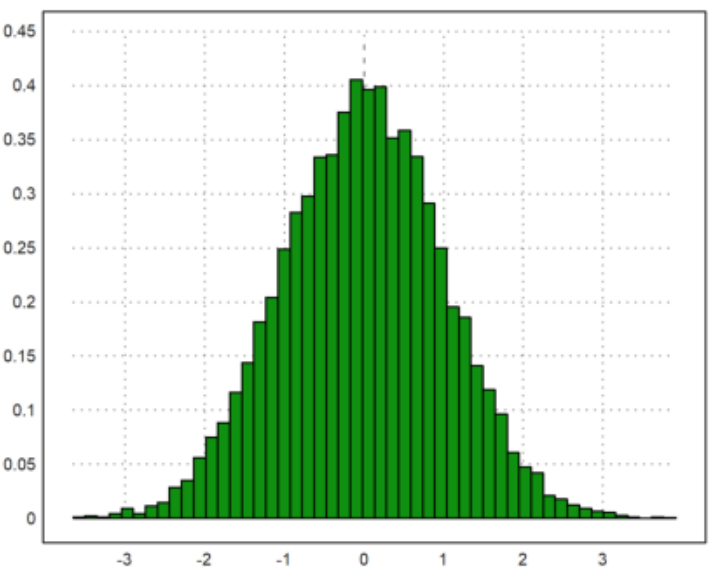
Gaya isian (fill styles) dapat digunakan sama seperti sebelumnya.

```
>n=10; k=bin(n,0:n); ...
>plot2d(-0.5:n+0.5,k,bar=true,fillcolor=lightgray):
```

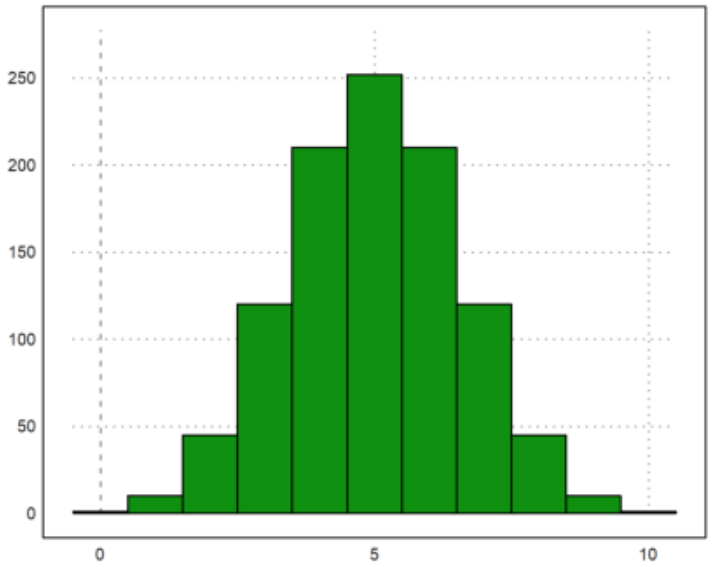


Data untuk plot batang (bar=1) dan histogram (histogram=1) dapat diberikan secara eksplisit dalam xv dan yv, atau dapat dihitung dari distribusi empiris dalam xv dengan >distribution (atau distribution=n). Histogram dari nilai xv akan dihitung secara otomatis dengan >histogram. Jika >even ditentukan, nilai xv akan dihitung dalam interval bilangan bulat.

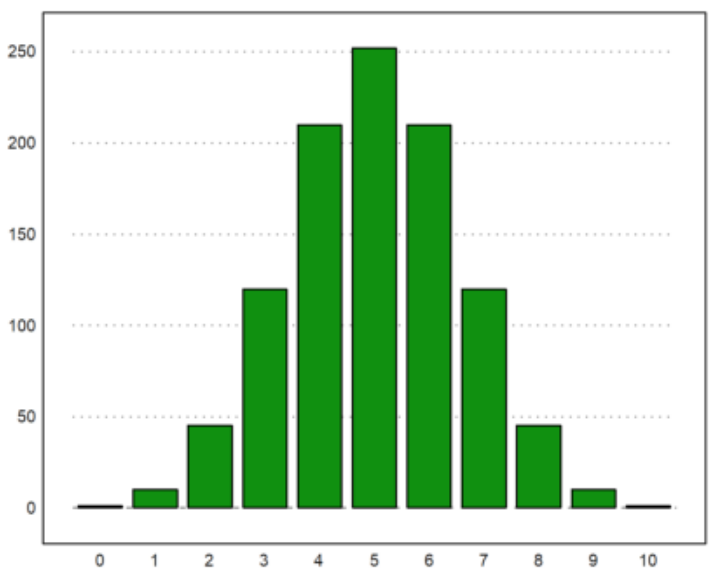
```
>plot2d(normal(10000),distribution=50):
```



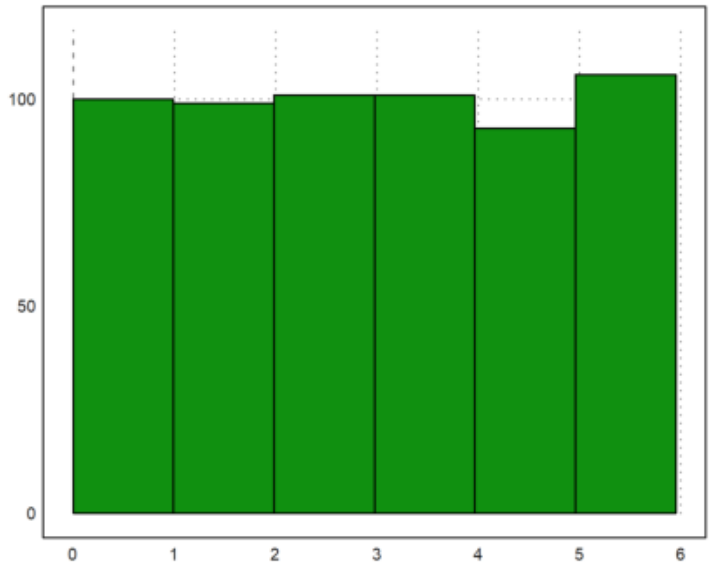

```
>k=0:10; m=bin(10,k); x=(0:11)-0.5; plot2d(x,m,>bar):
```




```
>columnsplot(m,k):
```

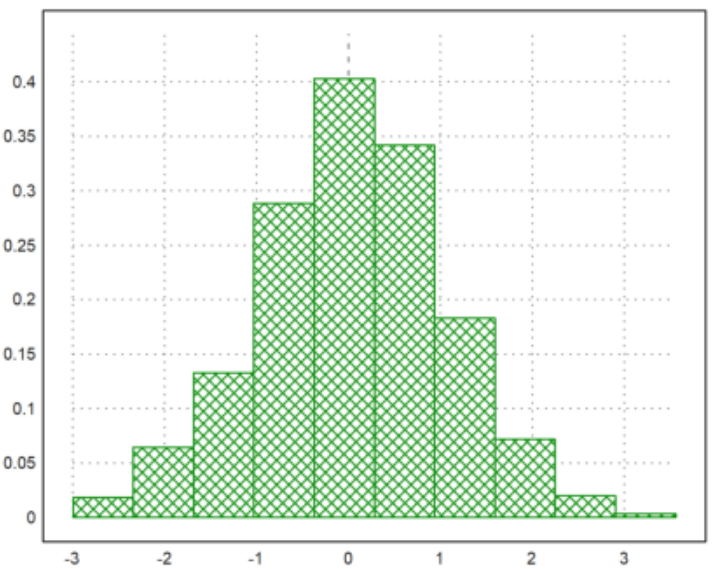



```
>plot2d(random(600)*6,histogram=6):
```



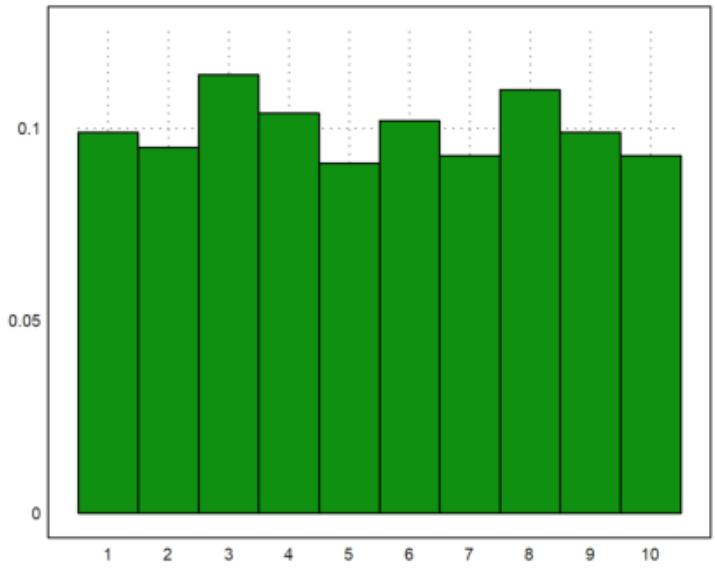
Untuk distribusi, terdapat parameter distribution=n, yang secara otomatis menghitung nilai dan menampilkan distribusi relatif dengan n sub-interval.

```
>plot2d(normal(1,1000),distribution=10,style="\/"):
```



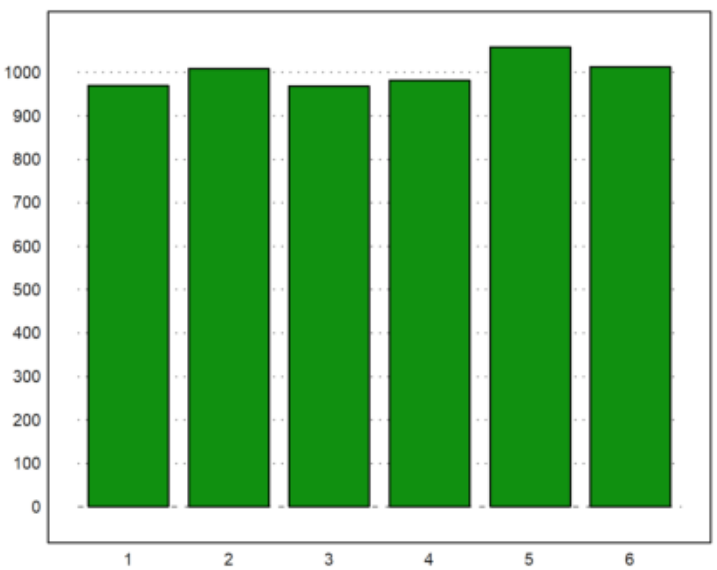
Dengan parameter even=true, ini akan menggunakan interval bilangan bulat.

```
>plot2d(intrandom(1,1000,10),distribution=10,even=true):
```

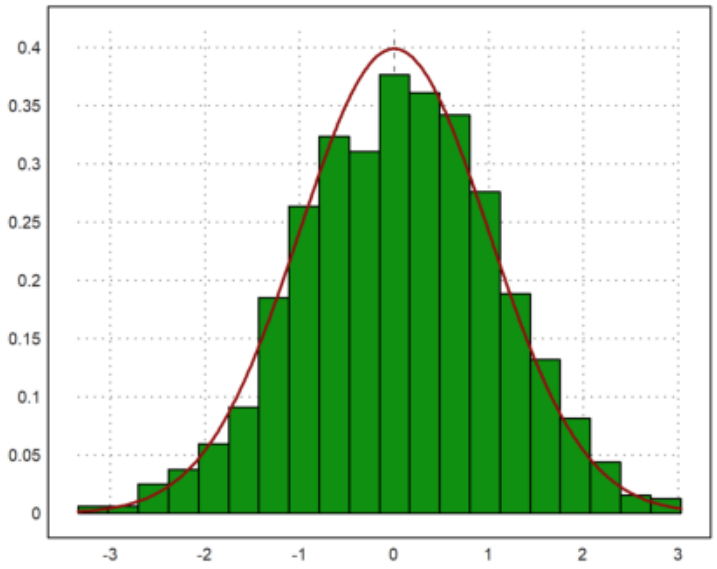


Perlu dicatat bahwa ada banyak plot statistik yang mungkin berguna. Silakan lihat tutorial tentang statistik.

```
>columnsplot(getmultiplicities(1:6,intrandom(1,6000,6))):
```

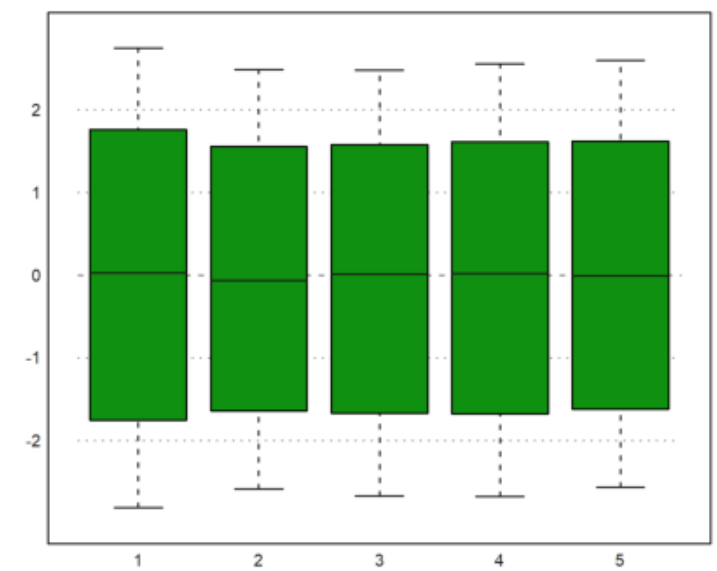



```
>plot2d(normal(1,1000),>distribution); ...
>  plot2d("qnormal(x)",color=red,thickness=2,>add):
```



Terdapat juga banyak plot khusus untuk statistik. Sebuah boxplot menunjukkan kuartil dari distribusi serta banyak pencilan (outliers). Menurut definisi, pencilan dalam sebuah boxplot adalah data yang melebihi 1,5 kali rentang 50% tengah dari plot.

```
>M=normal(5,1000); boxplot(quartiles(M)):
```



Fungsi Implisit

Plot implisit menampilkan garis level yang menyelesaikan f(x,y)=level, di mana level bisa berupa satu nilai atau sebuah vektor nilai. Jika level="auto", maka akan ada nc garis level, yang tersebar merata antara nilai minimum dan maksimum fungsi. Warna yang lebih gelap atau lebih terang dapat ditambahkan dengan >hue untuk menunjukkan nilai fungsi. Untuk fungsi implisit, xv harus berupa fungsi atau ekspresi dari parameter x dan y, atau sebagai alternatif, xv dapat berupa matriks nilai.

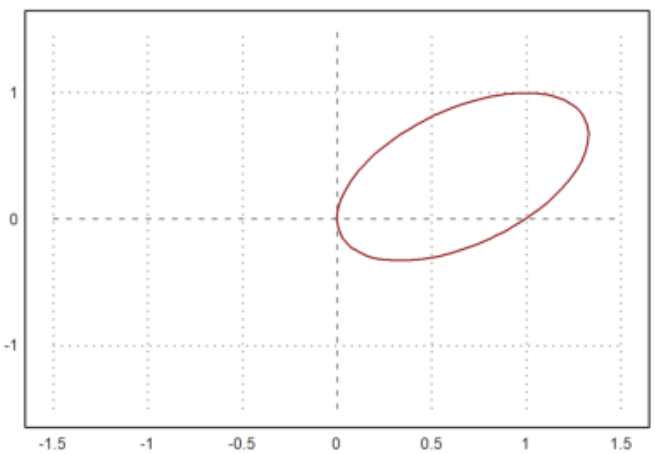
Euler dapat menandai garis level

$$f(x,y) = c$$

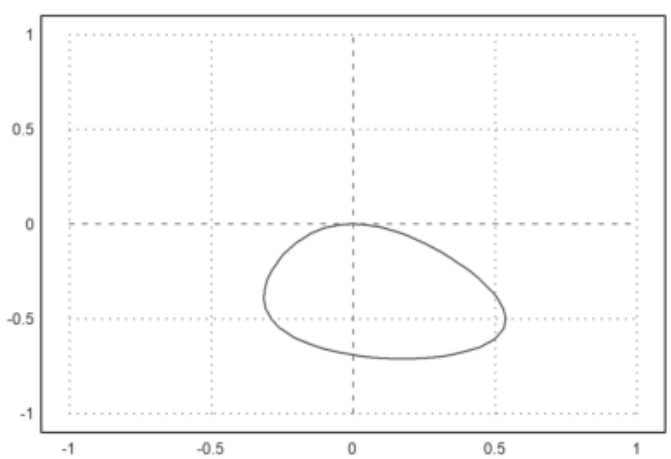
dari fungsi apa pun.

Untuk menggambar himpunan f(x,y)=c untuk satu atau lebih konstanta c, Anda dapat menggunakan plot2d() dengan plot implisitnya pada bidang. Parameter untuk c adalah level=c, di mana c bisa berupa vektor garis level. Selain itu, sebuah skema warna dapat ditampilkan di latar belakang untuk menunjukkan nilai fungsi pada setiap titik dalam plot. Parameter "n" menentukan tingkat ketelitian (fineness) dari plot.

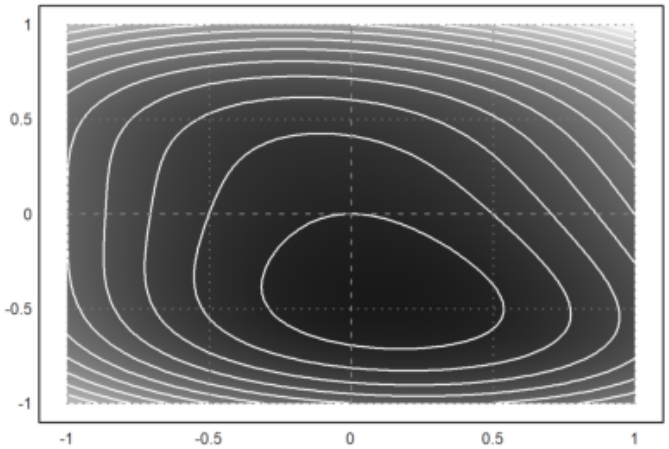
```
>aspect(1.5);
>plot2d("x^2+y^2-x*y-x",r=1.5,level=0,contourcolor=red):
```



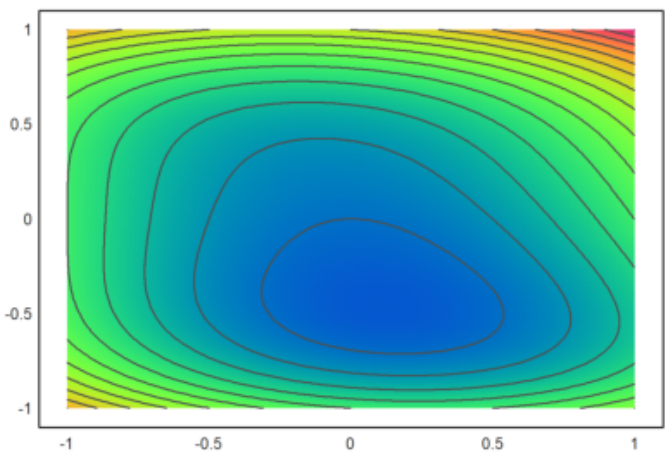

```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=0): // Solutions of f(x,y)=0
```




```
>plot2d(expr,level=0:0.5:20,>hue,contourcolor=white,n=200): // nice
```

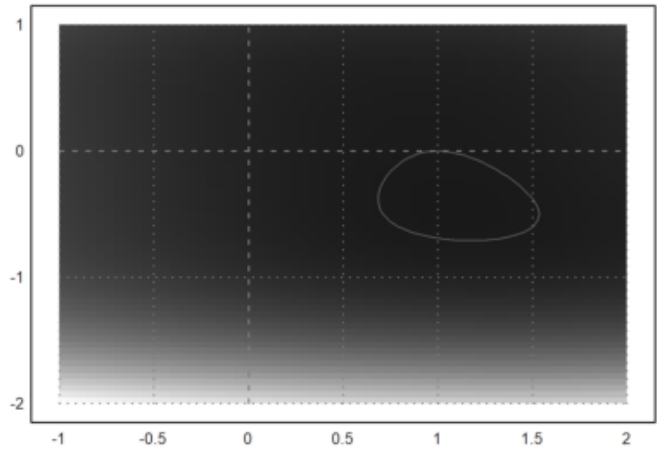



```
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=200,grid=4): // nicer
```

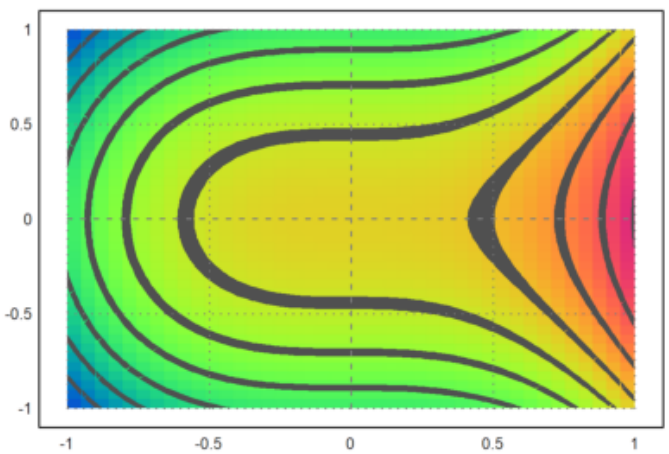


Hal ini juga berfungsi untuk plot data. Namun, Anda harus menentukan rentang untuk label sumbunya.

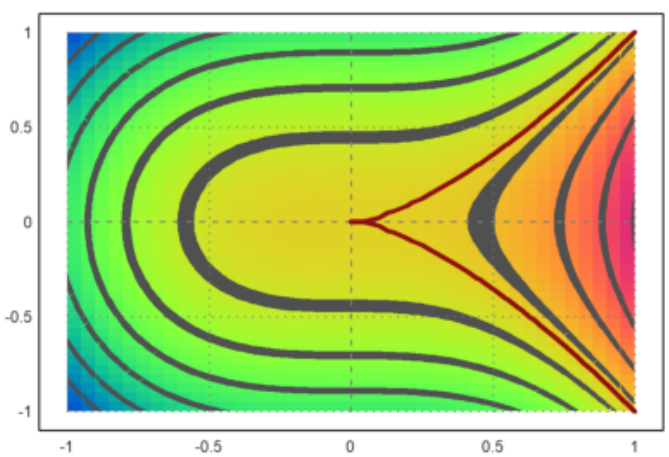
```
>x=-2:0.05:1; y=x'; z=expr(x,y);
>plot2d(z,level=0,a=-1,b=2,c=-2,d=1,>hue):
```



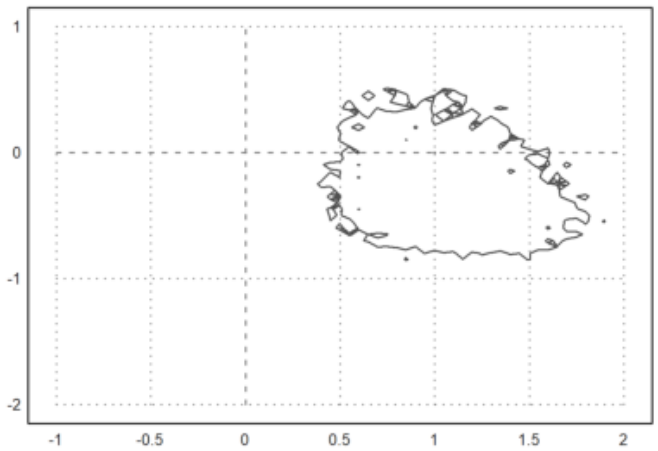

```
>plot2d("x^3-y^2",>contour,>hue,>spectral):
```



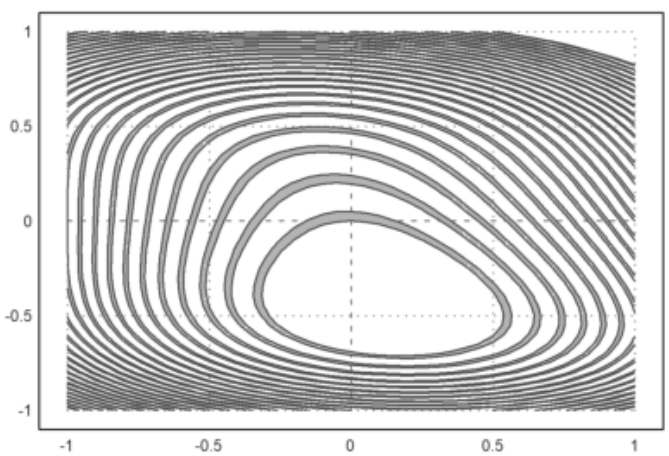

```
>plot2d("x^3-y^2",level=0,contourwidth=3,>add,contourcolor=red):
```



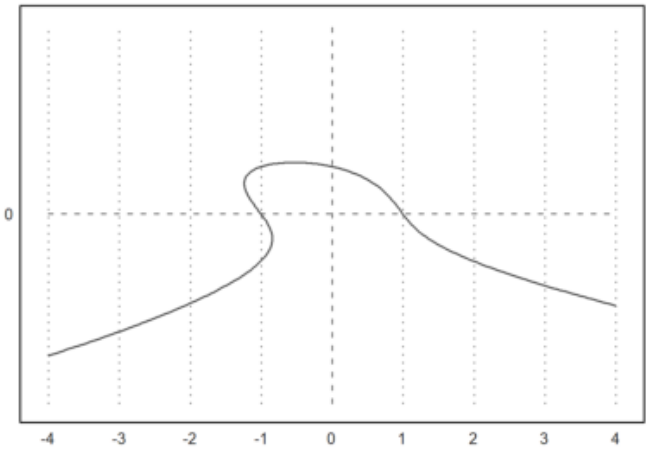
```
>z=z+normal(size(z))*0.2;
>plot2d(z,level=0.5,a=-1,b=2,c=-2,d=1):
```



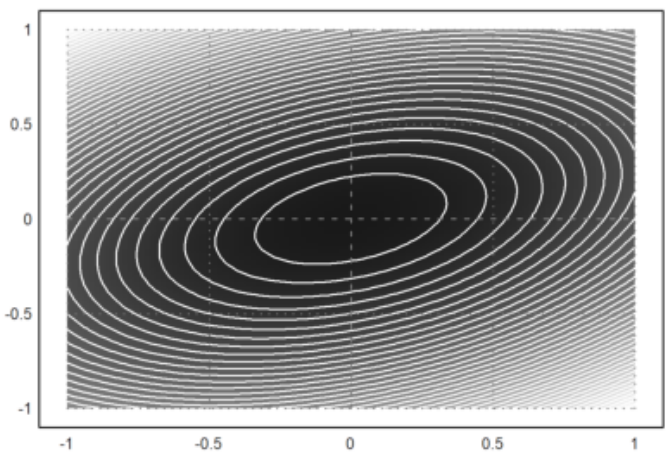

```
>plot2d(expr,level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
```




```
>plot2d("x^2+y^3+x*y",level=1,r=4,n=100):
```




```
>plot2d("x^2+2*y^2-x*y",level=0:0.1:10,n=100,contourcolor=white,>hue):
```



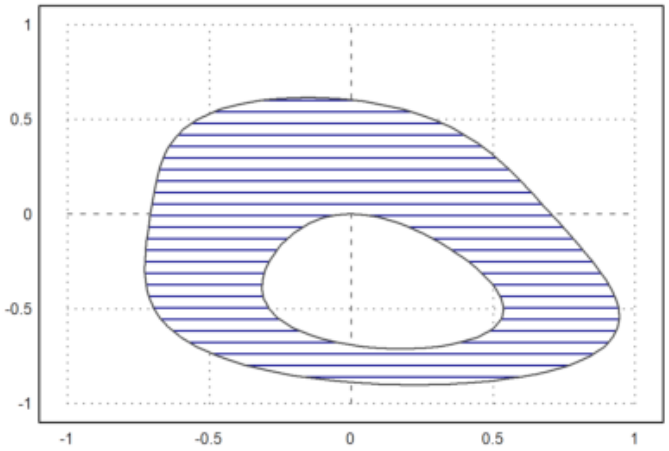
Dimungkinkan juga untuk mengisi himpunan

$$a \le f(x,y) \le b$$

dengan rentang level.

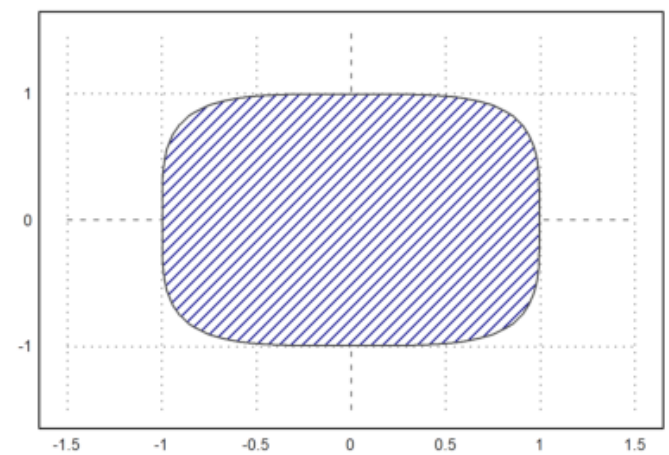
Dimungkinkan untuk mengisi wilayah nilai untuk fungsi tertentu. Untuk ini, level harus berupa matriks 2xn. Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

```
>plot2d(expr,level=[0;1],style="-",color=blue): // 0 <= f(x,y) <= 1
```

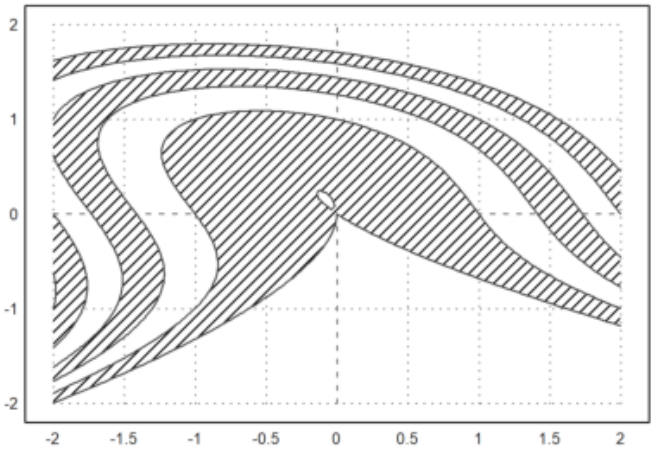


Plot implisit juga dapat menampilkan rentang level. Dalam hal ini, level harus berupa matriks 2×n dari interval level, di mana baris pertama berisi nilai awal dan baris kedua berisi nilai akhir dari setiap interval. Sebagai alternatif, sebuah vektor baris sederhana dapat digunakan untuk level, dan sebuah parameter dl akan memperluas nilai level menjadi interval.

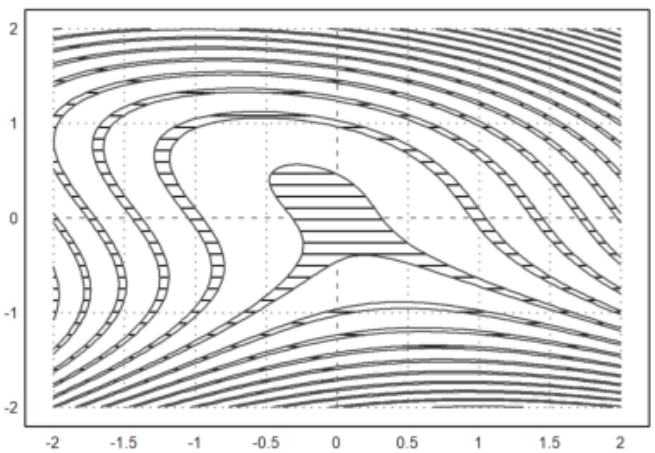
```
>plot2d("x^4+y^4",r=1.5,level=[0;1],color=blue,style="/"):
```



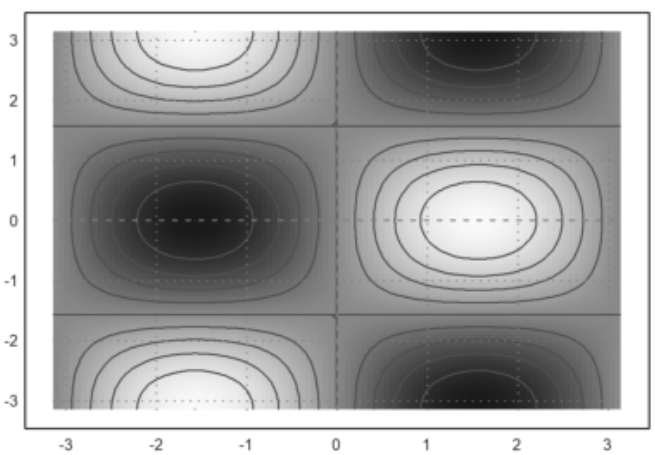

```
>plot2d("x^2+y^3+x*y",level=[0,2,4;1,3,5],style="/",r=2,n=100):
```




```
>plot2d("x^2+y^3+x*y",level=-10:20,r=2,style="-",dl=0.1,n=100):
```




```
>plot2d("sin(x)*cos(y)",r=pi,>hue,>levels,n=100):
```

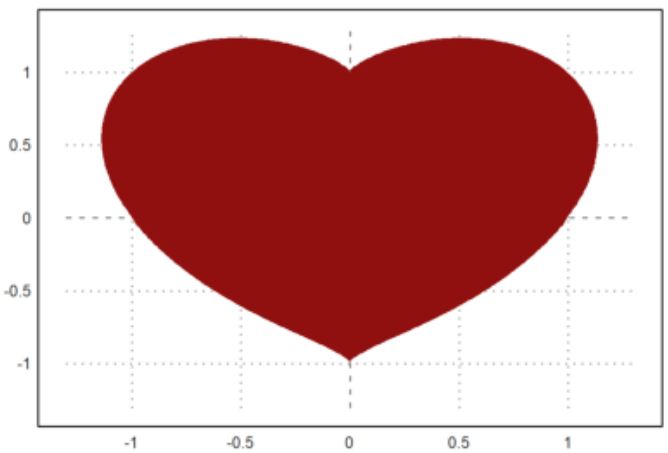


Anda juga dapat menandai suatu wilayah

$$a \leq f(x,y) \leq b.$$

Hal ini dilakukan dengan menambahkan level dengan dua baris.

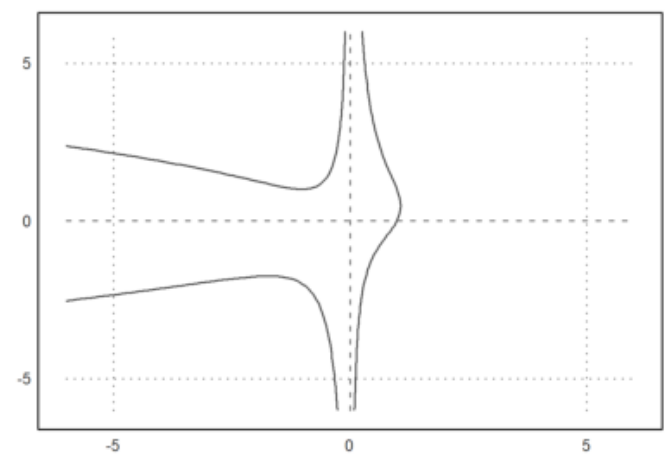
```
>plot2d("(x^2+y^2-1)^3-x^2*y^3",r=1.3, ...
>  style="#",color=red,<outline, ...
>  level=[-2;0],n=100):
```



Dimungkinkan untuk menentukan level tertentu. Misalnya, kita dapat memplot solusi persamaan seperti

$$x^3 - xy + x^2y^2 = 6$$

```
>plot2d("x^3-x*y+x^2*y^2",r=6,level=1,n=100):
```



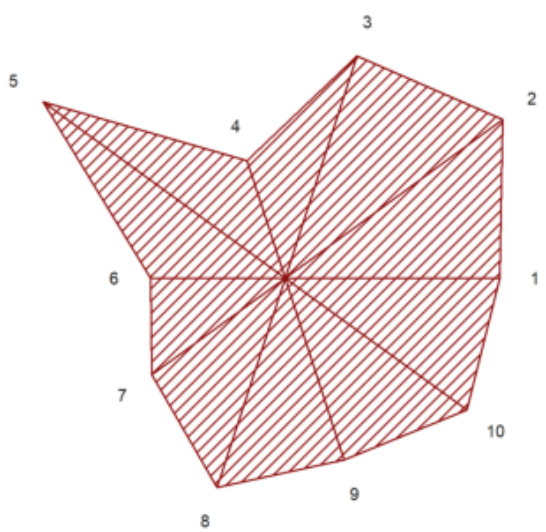
```
>function starplot1 (v, style="/", color=green, lab=none) ...
```

```
    if !holding() then clg; endif;
    w=window(); window(0,0,1024,1024);
    h=holding(1);
    r=max(abs(v))*1.2;
    setplot(-r,r,-r,r);
    n=cols(v); t=linspace(0,2pi,n);
    v=v|v[1]; c=v*cos(t); s=v*sin(t);
    cl=barcolor(color); st=barstyle(style);
    loop 1 to n
      polygon([0,c[#],c[#+1]],[0,s[#],s[#+1]],1);
      if lab!=none then
        rlab=v[#]+r*0.1;
        {col,row}=toscreen(cos(t[#])*rlab,sin(t[#])*rlab);
        ctext(""+lab[#],col,row-textheight()/2);
      endif;
    end;
    barcolor(cl); barstyle(st);
    holding(h);
    window(w);
  endfunction
```

Di sini tidak ada grid atau tanda sumbu. Selain itu, kita menggunakan seluruh jendela untuk plot.

Kita memanggil reset sebelum menguji plot ini untuk mengembalikan pengaturan grafis ke default. Hal ini tidak diperlukan jika Anda yakin bahwa plot Anda sudah berjalan dengan baik.

```
>reset; starplot1(normal(1,10)+5,color=red,lab=1:10):
```



Terkadang, Anda mungkin ingin memplot sesuatu yang tidak dapat dilakukan oleh plot2d, atau hampir bisa tetapi tidak sepenuhnya.

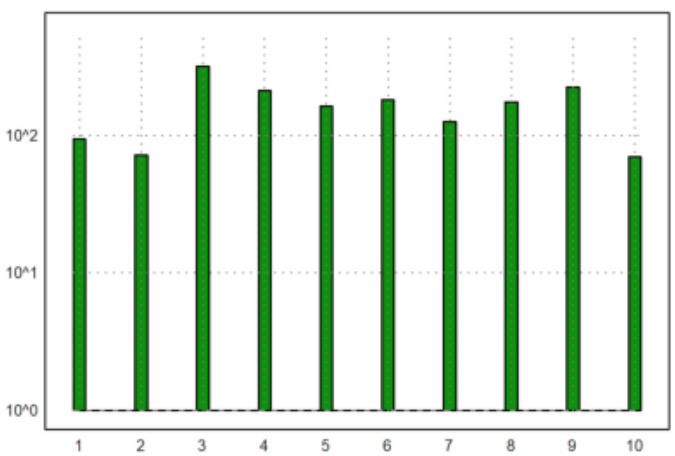
Pada fungsi berikut, kita membuat plot impuls logaritmik. plot2d dapat membuat plot logaritmik, tetapi tidak untuk batang impuls.

```
>function logimpulseplot1 (x,y) ...
```

```
    {x0,y0}=makeimpulse(x,log(y)/log(10));
    plot2d(x0,y0,>bar,grid=0);
    h=holding(1);
    frame();
    xgrid(ticks(x));
    p=plot();
    for i=-10 to 10;
      if i<=p[4] and i>=p[3] then
         ygrid(i,yt="10^"+i);
      endif;
    end;
    holding(h);
  endfunction
```

Mari kita uji dengan nilai yang terdistribusi secara eksponensial.

```
>aspect(1.5); x=1:10; y=-log(random(size(x)))*200; ...
>logimpulseplot1(x,y):
```



Mari kita menganimasikan sebuah kurva 2D menggunakan plot langsung. Perintah plot(x,y) hanya akan memplot sebuah kurva ke dalam jendela plot. setplot(a,b,c,d) mengatur jendela tersebut.

Fungsi wait(0) memaksa plot untuk muncul pada jendela grafik. Jika tidak, proses redraw hanya akan terjadi pada selang waktu tertentu.

```
>function animliss (n,m) ...
```

```
t=linspace(0,2pi,500);
f=0;
c=framecolor(0);
l=linewidth(2);
setplot(-1,1,-1,1);
repeat
  clg;
  plot(sin(n*t),cos(m*t+f));
  wait(0);
  if testkey() then break; endif;
  f=f+0.02;
end;
framecolor(c);
linewidth(l);
endfunction
```

Tekan sembarang tombol untuk menghentikan animasi ini.

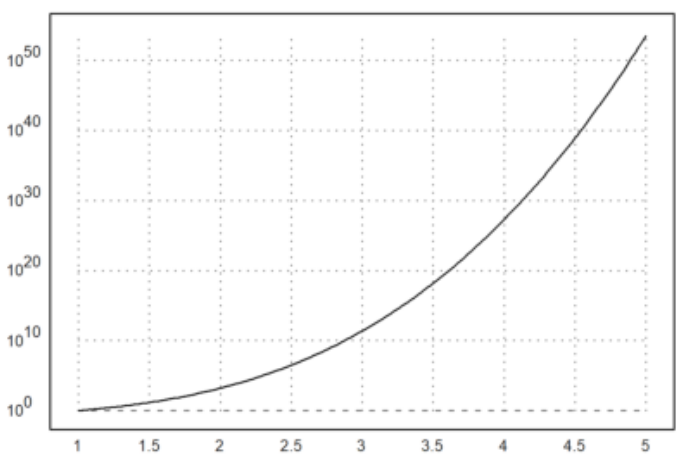
```
>animliss(2,3); // lihat hasilnya, jika sudah puas, tekan ENTER
```

Plot Logaritmik

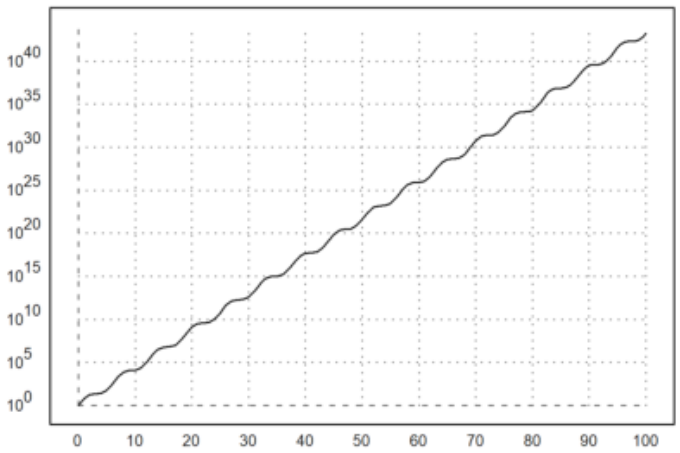
EMT menggunakan parameter "logplot" untuk skala logaritmik.
lot logaritmik dapat dibuat dengan menggunakan skala logaritmik pada y dengan logplot=1, atau menggunakan skala logaritmik pada x dan y dengan logplot=2, atau pada x dengan logplot=3.

- logplot=1: logaritmik pada sumbu-y
- logplot=2: logaritmik pada sumbu-x dan sumbu-y
- logplot=3: logaritmik pada sumbu-x

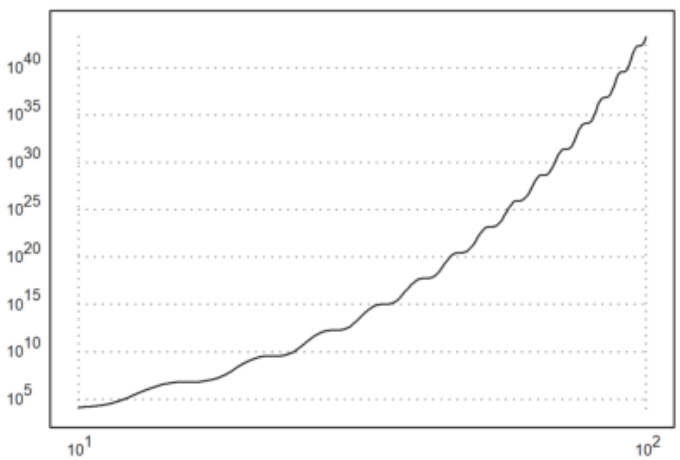
```
>plot2d("exp(x^3-x)*x^2",1,5,logplot=1):
```



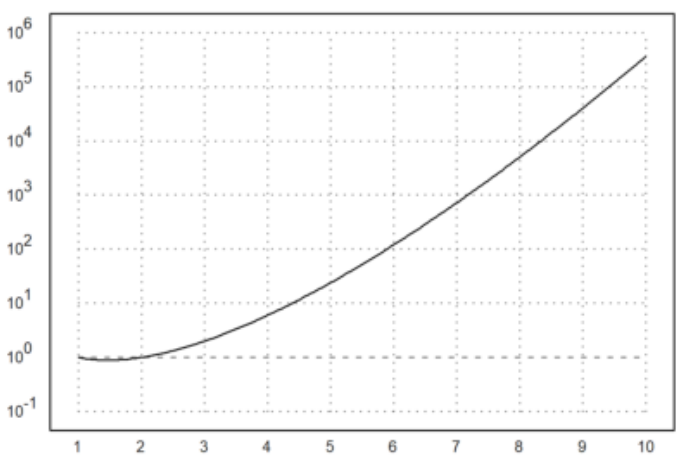

```
>plot2d("exp(x+sin(x))",0,100,logplot=1):
```



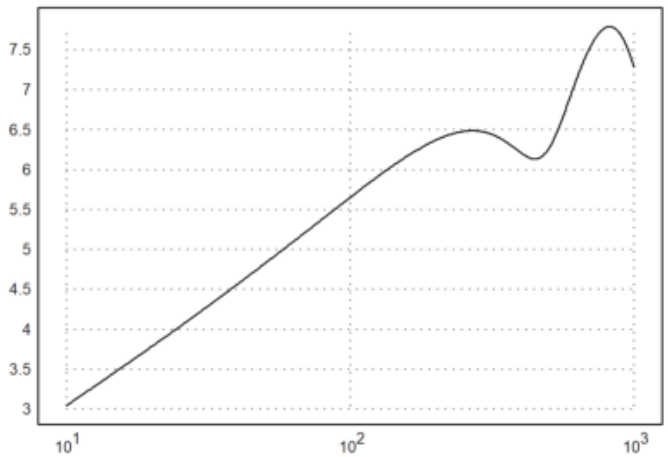

```
>plot2d("exp(x+sin(x))",10,100,logplot=2):
```




```
>plot2d("gamma(x)",1,10,logplot=1):
```

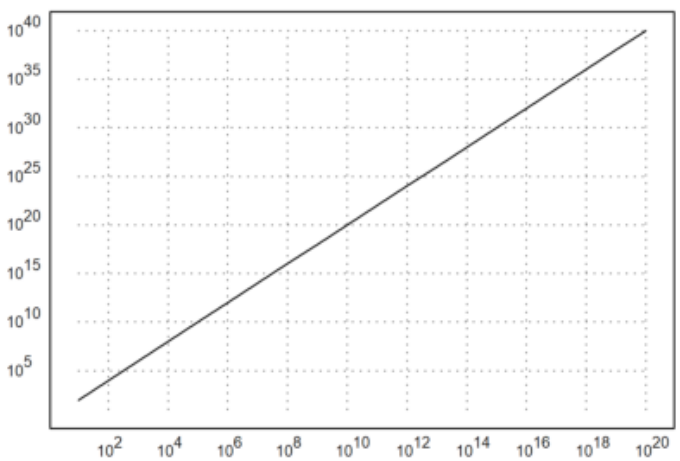


```
>plot2d("log(x*(2+sin(x/100)))",10,1000,logplot=3):
```



Hal ini juga berfungsi dengan plot data.

```
>x=10^(1:20); y=x^2-x;
>plot2d(x,y,logplot=2):
```



Rujukan Lengkap Fungsi plot2d()

```
function plot2d (xv, yv, btest, a, b, c, d, xmin, xmax, r, n,  ..
logplot, grid, frame, framecolor, square, color, thickness, style, ..
auto, add, user, delta, points, addpoints, pointstyle, bar, histogram,  ..
distribution, even, steps, own, adaptive, hue, level, contour,  ..
nc, filled, fillcolor, outline, title, xl, yl, maps, contourcolor, ..
contourwidth, ticks, margin, clipping, cx, cy, insimg, spectral,  ..
cgrid, vertical, smaller, dl, niveau, levels)
```

Multipurpose plot function for plots in the plane (2D plots). This function can do plots of functions of one variables, data plots, curves in the plane, bar plots, grids of complex numbers, and implicit plots of functions of two variables.

Parameters
x,y : equations, functions or data vectors
a,b,c,d : Plot area (default a=-2,b=2)
r : if r is set, then a=cx-r, b=cx+r, c=cy-r, d=cy+r

```
r can be a vector [rx,ry] or a vector [rx1,rx2,ry1,ry2].
```

xmin,xmax : range of the parameter for curves
auto : Determine y-range automatically (default)
square : if true, try to keep square x-y-ranges
n : number of intervals (default is adaptive)
grid : 0 = no grid and labels,

```
1 = axis only,
2 = normal grid (see below for the number of grid lines)
3 = inside axis
4 = no grid
5 = full grid including margin
6 = ticks at the frame
7 = axis only
8 = axis only, sub-ticks
```

frame : 0 = no frame
framecolor: color of the frame and the grid
margin : number between 0 and 0.4 for the margin around the plot
color : Color of curves. If this is a vector of colors,

```
it will be used for each row of a matrix of plots. In the case of
point plots, it should be a column vector. If a row vector or a
full matrix of colors is used for point plots, it will be used for
each data point.
```

thickness : line thickness for curves

```
This value can be smaller than 1 for very thin lines.
```

style : Plot style for lines, markers, and fills.

```
For points use
"[]", "<>", ".", "..", "...",
"*", "+", "|", "-", "o"
"[]#", "<>#", "o#" (filled shapes)
"[]w", "<>w", "ow" (non-transparent)
For lines use
"-", "--", "-.", ".", ".-.", "-.-", "->"
For filled polygons or bar plots use
"#", "#O", "O", "/", "\", "\/",
"+", "|", "-", "t"
```

points : plot single points instead of line segments
addpoints : if true, plots line segments and points
add : add the plot to the existing plot
user : enable user interaction for functions
delta : step size for user interaction
bar : bar plot (x are the interval bounds, y the interval values)
histogram : plots the frequencies of x in n subintervals
distribution=n : plots the distribution of x with n subintervals
even : use inter values for automatic histograms.
steps : plots the function as a step function (steps=1,2)
adaptive : use adaptive plots (n is the minimal number of steps)
level : plot level lines of an implicit function of two variables
outline : draws boundary of level ranges.
If the level value is a 2xn matrix, ranges of levels will be drawn
in the color using the given fill style. If outline is true, it
will be drawn in the contour color. Using this feature, regions of
f(x,y) between limits can be marked.
hue : add hue color to the level plot to indicate the function

```
            value
```

contour : Use level plot with automatic levels
nc : number of automatic level lines
title : plot title (default "")
xl, yl : labels for the x- and y-axis
smaller : if >0, there will be more space to the left for labels.
vertical :

```
  Turns vertical labels on or off. This changes the global variable
  verticallabels locally for one plot. The value 1 sets only vertical
  text, the value 2 uses vertical numerical labels on the y axis.
```

filled : fill the plot of a curve
fillcolor : fill color for bar and filled curves
outline : boundary for filled polygons
logplot : set logarithmic plots

```
            1 = logplot in y,
            2 = logplot in xy,
            3 = logplot in x
```

own :

```
  A string, which points to an own plot routine. With >user, you get
  the same user interaction as in plot2d. The range will be set
  before each call to your function.
```

maps : map expressions (0 is faster), functions are always mapped.
contourcolor : color of contour lines
contourwidth : width of contour lines
clipping : toggles the clipping (default is true)
title :

```
This can be used to describe the plot. The title will appear above
the plot. Moreover, a label for the x and y axis can be added with
xl="string" or yl="string". Other labels can be added with the
functions label() or labelbox(). The title can be a unicode
string or an image of a Latex formula.
```

cgrid :

```
Determines the number of grid lines for plots of complex grids.
Should be a divisor of the the matrix size minus 1 (number of
subintervals). cgrid can be a vector [cx,cy].
```

Overview

The function can plot

- expressions, call collections or functions of one variable,
- parametric curves,
- x data against y data,
- implicit functions,
- bar plots,
- complex grids,
- polygons.

If a function or expression for xv is given, plot2d() will compute values in the given range using the function or expression. The expression must be an expression in the variable x. The range must be defined in the parameters a and b unless the default range should be used. The y-range will be computed automatically, unless c and d are specified, or a radius r, which yields the range r,r for x and y. For plots of functions, plot2d will use an adaptive evaluation of the function by default. To speed up the plot for complicated functions, switch this off with <adaptive, and optionally decrease the number of intervals n. Moreover, plot2d() will by default use mapping. I.e., it will compute the plot element for element. If your expression or your functions can handle a vector x, you can switch that off with <maps for faster evaluation.

Note that adaptive plots are always computed element for element. If functions or expressions for both xv and for yv are specified, plot2d() will compute a curve with the xv values as x-coordinates and the yv values as y-coordinates. In this case, a range should be defined for the parameter using xmin, xmax. Expressions contained in strings must always be expressions in the parameter variable x.