

Menggambar Grafik 2D dengan EMT

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

Basic Plots

Ada fungsi-fungsi plot yang sangat mendasar. Terdapat koordinat layar, yang selalu berkisar antara 0 hingga 1024 di setiap sumbu, terlepas dari apakah layarnya persegi atau tidak. Selain itu, terdapat koordinat plot, yang dapat diatur dengan `setplot()`. Pemetaan antar koordinat bergantung pada jendela plot saat ini. Misalnya, `shrinkwindow()` default menyisakan ruang untuk label sumbu dan judul plot. Dalam contoh ini, kita hanya menggambar beberapa garis acak dengan berbagai warna. Untuk detail tentang fungsi-fungsi ini, pelajari fungsi-fungsi inti 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;
```

Grafik perlu ditahan, karena perintah `plot()` akan menghapus jendela plot.

Untuk menghapus semua yang telah kita lakukan, kita menggunakan `reset()`.

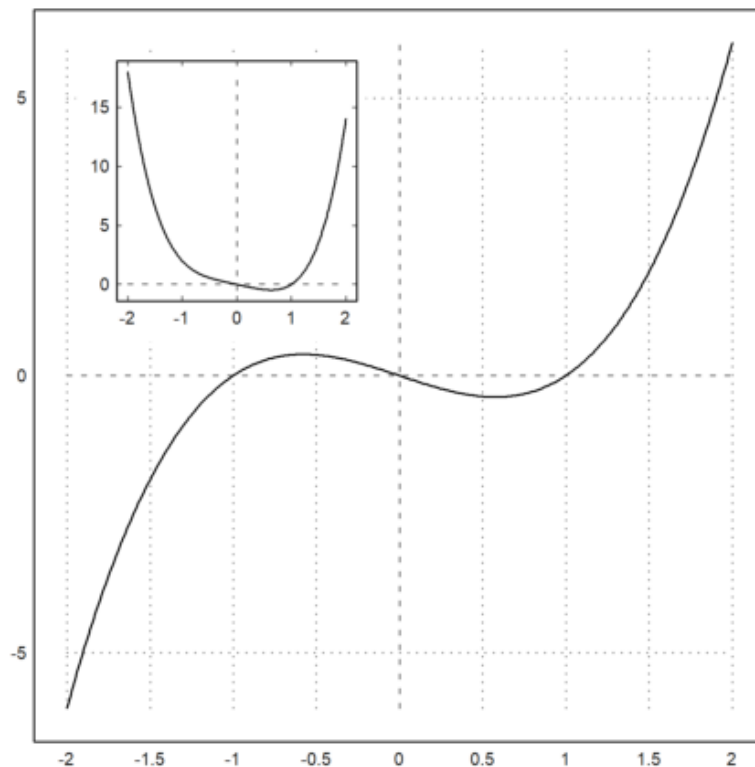
Untuk menampilkan gambar hasil plot di layar notebook, perintah `plot2d()` dapat diakhiri dengan titik dua (:). Cara lainnya adalah perintah `plot2d()` diakhiri dengan titik koma (;), kemudian menggunakan perintah `insimg()` untuk menampilkan gambar hasil plot.

Sebagai contoh lain, kita menggambar plot sebagai inset di plot lain. Ini dilakukan dengan mendefinisikan jendela plot yang lebih kecil. Perhatikan bahwa jendela ini tidak menyediakan ruang untuk label sumbu di luar jendela plot. Kita perlu menambahkan margin untuk ini sesuai kebutuhan. Perhatikan bahwa kita menyimpan dan memulihkan jendela penuh, dan menahan plot saat ini saat kita memplot inset.

```

>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;
>window(ow);

```

A plot with multiple figures is achieved in the same way. There is the utility figure() function for this.

Plot Aspect

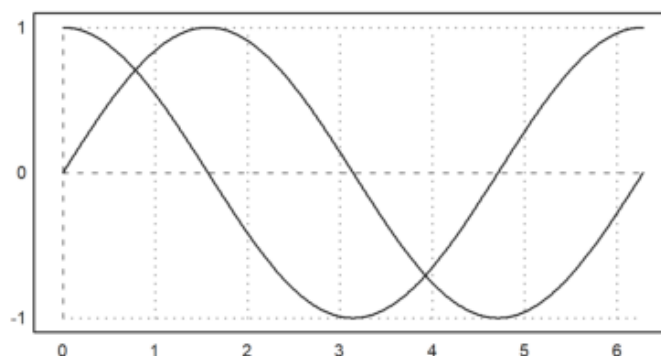
The default plot uses a square plot window. You can change this with the function aspect(). Do not forget to reset the aspect later. You can also change this default in the menu with "Set Aspect" to a specific aspect ratio or to the current size of the graphics window.

But you can change it also for one plot. For this, the current size of the plot area is changed, and the window is set so that the labels have enough space.

```

>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.

2D Plots in Euler

EMT Math Toolbox has plots in 2D, both for data and functions. EMT uses the function `plot2d`. This function can plot functions and data.

It is possible to plot in Maxima using Gnuplot or in Python using Math Plot Lib.

Euler can plot 2D plots of

- ekspresi
- functions, variables, or parameterized curves,
- vectors of x-y-values,
- clouds of points in the plane,
- implicit curves with levels or level regions.
- Complex function

Plot styles include various styles for lines and points, bar plots and shaded plots.

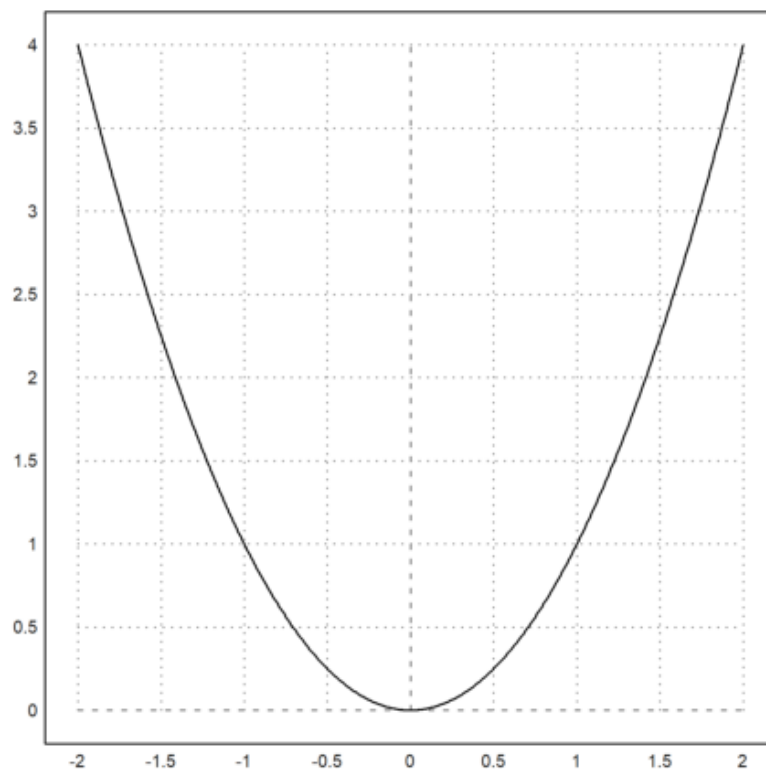
Plot Ekspresi atau Variabel

A single expression in "x" (e.g. " $4x^2$ ") or the name of a function (e.g. "f") produces a graph of the function.

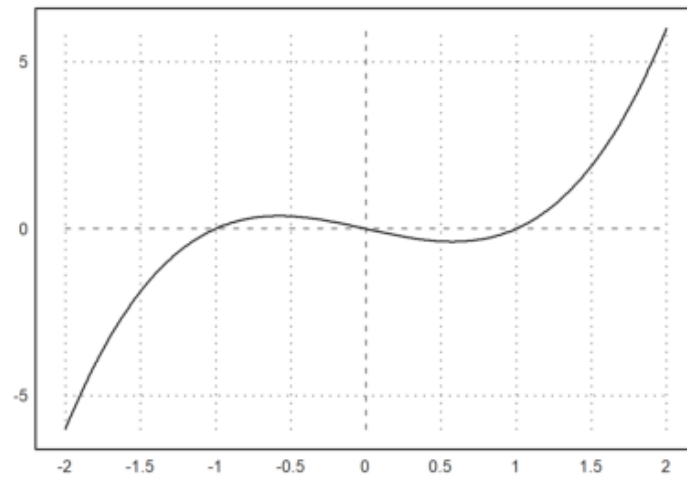
Here is the most basic example, which uses the default range and sets a proper y-range to fit the plot of the function.

Note: If you end a command line with a colon ":", the plot will be inserted into the text window. Otherwise, press TAB to see the plot if the plot window is covered.

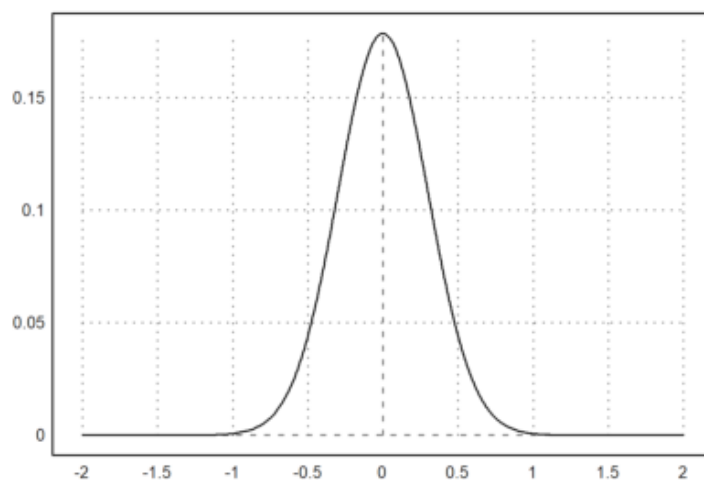
```
>plot2d("x^2"):
```



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



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

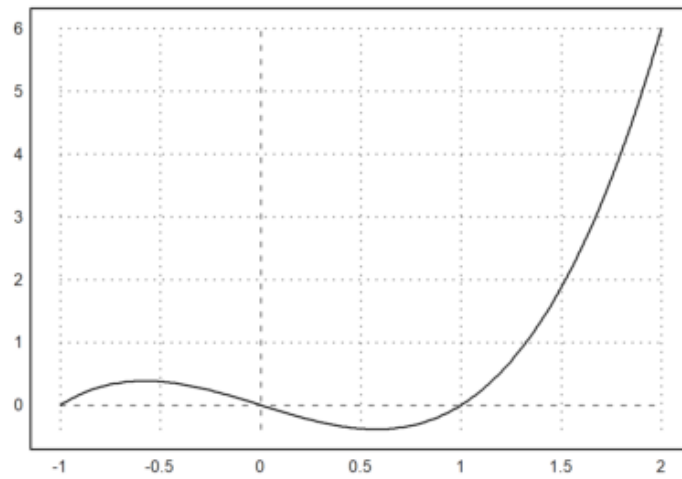


Dari beberapa contoh sebelumnya Anda dapat melihat bahwa aslinya gambar plot menggunakan sumbu X dengan rentang nilai dari -2 sampai dengan 2. Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan nilai-nilai batas X (dan Y) di belakang ekspresi yang digambar.

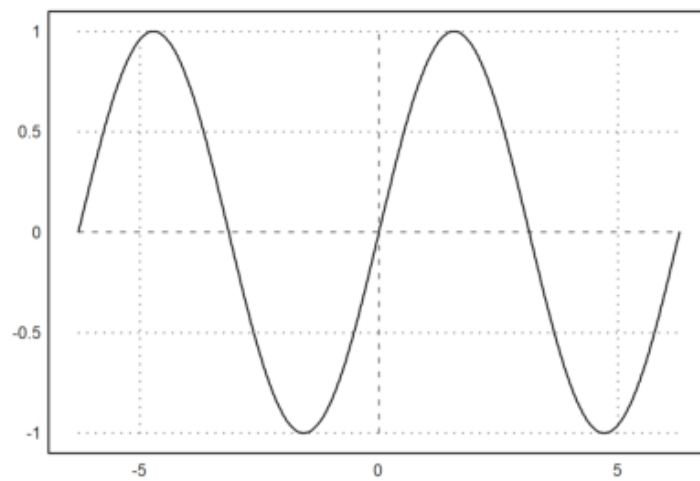
The plot range is set with the following assigned parameters

- a,b: x-range (default -2,2)
- c,d: y-range (default: scale with values)
- r: alternatively a radius around the plot center
- cx,cy: the coordinates of the plot center (default 0,0)

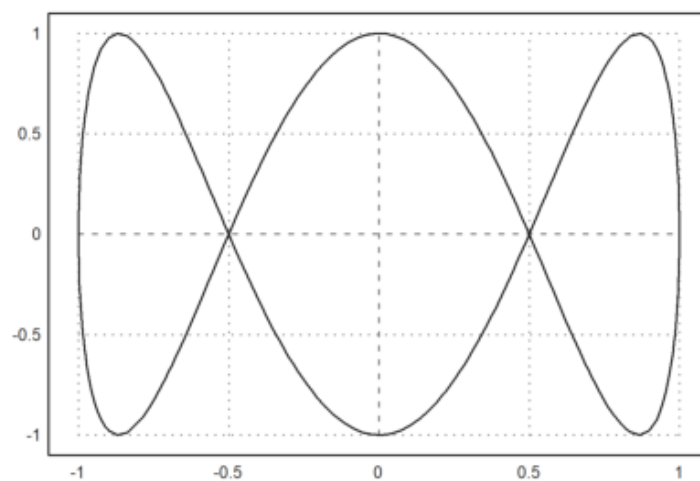
```
>plot2d("x^3-x",-1,2):
```



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



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



Alternatif untuk titik dua adalah perintah `insimg(baris)`, yang menyisipkan plot yang menempati sejumlah baris teks tertentu.

Dalam opsi, plot dapat diatur agar muncul

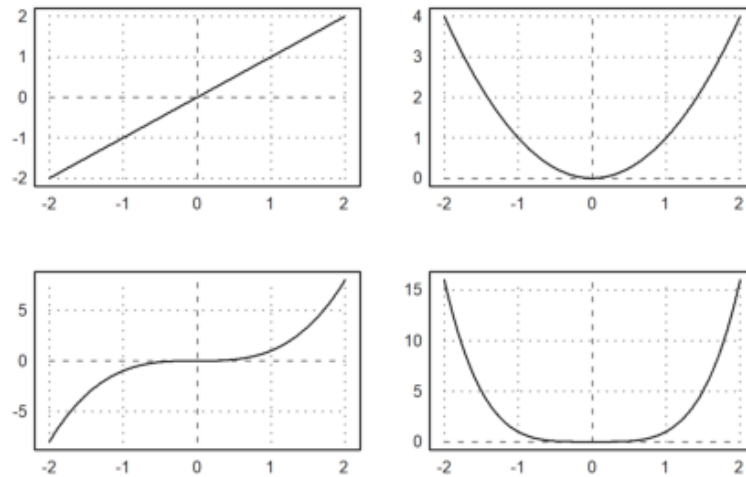
- di jendela terpisah yang dapat diubah ukurannya,
- di jendela buku catatan.

Lebih banyak gaya dapat dicapai dengan perintah plot tertentu.

Apa pun masalahnya, tekan tombol tabulator untuk melihat plot, jika tersembunyi.

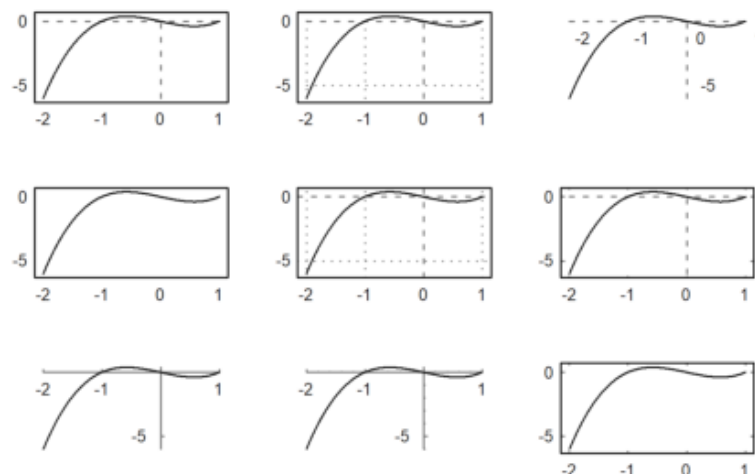
Untuk membagi jendela menjadi beberapa plot, gunakan perintah `figure()`. Dalam contoh ini, kita memplot x^1 hingga x^4 menjadi 4 bagian jendela. `figure(0)` akan mengatur ulang jendela default.

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



Dalam `plot2d()`, terdapat gaya alternatif yang tersedia dengan `grid=x`. Sebagai gambaran umum, kami menampilkan berbagai gaya kisi dalam satu gambar (lihat di bawah untuk perintah `figure()`). Gaya `grid=0` tidak disertakan. Gaya ini tidak menampilkan kisi dan bingkai.

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

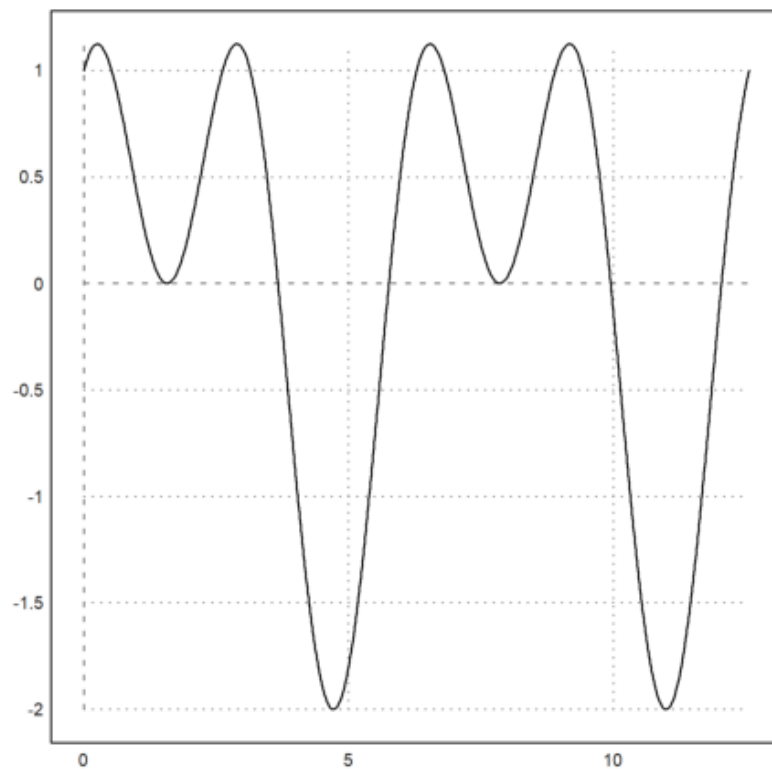


Jika argumen untuk `plot2d()` berupa ekspresi yang diikuti oleh empat angka, angka-angka ini adalah rentang x dan y untuk plot tersebut.

Sebagai alternatif, a , b , c , d dapat ditetapkan sebagai parameter yang ditetapkan sebagai $a=...$ dst.

Dalam contoh berikut, kita mengubah gaya kisi, 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):
```

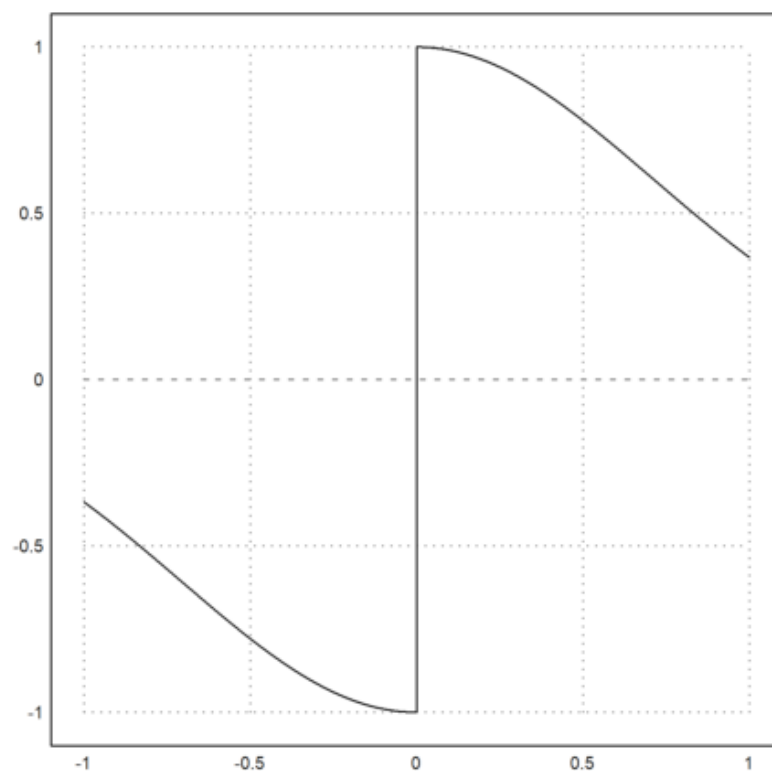


Gambar yang dihasilkan dengan memasukkan plot ke dalam jendela teks disimpan di direktori yang sama dengan buku catatan, secara default dalam subdirektori bernama "images". Gambar tersebut juga digunakan oleh ekspor HTML.

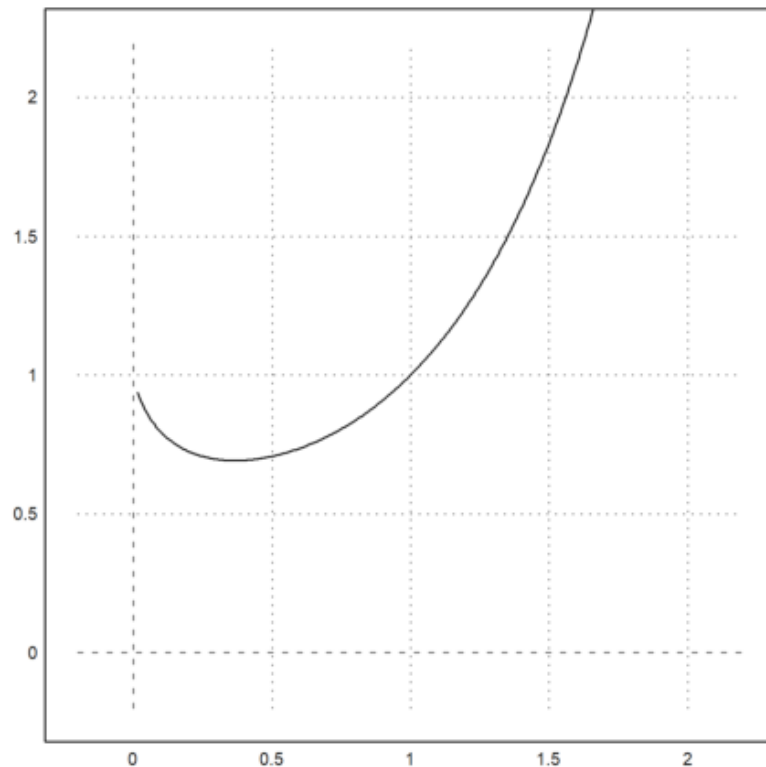
Anda cukup menandai gambar apa pun dan menyalinnya ke clipboard dengan Ctrl-C. Tentu saja, Anda juga dapat mengeksport grafik saat ini dengan fungsi-fungsi di menu File.

Fungsi atau ekspresi dalam plot2d dievaluasi secara adaptif. Untuk kecepatan lebih tinggi, nonaktifkan plot adaptif dengan <adaptive dan tentukan jumlah subinterval dengan n=.... Ini hanya diperlukan dalam kasus yang jarang terjadi.

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

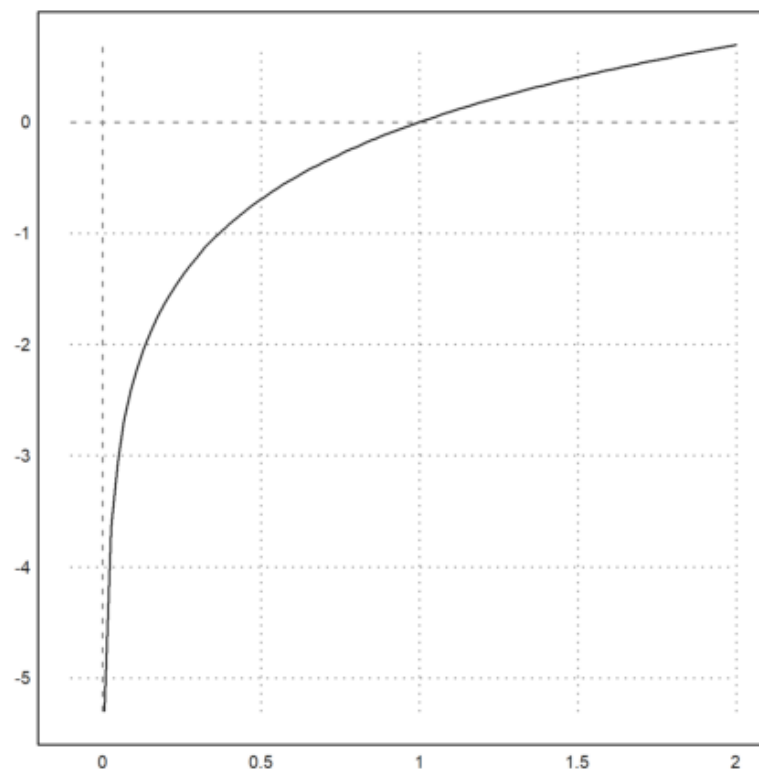


```
>plot2d("x^x",r=1.2,cx=1,cy=1):
```



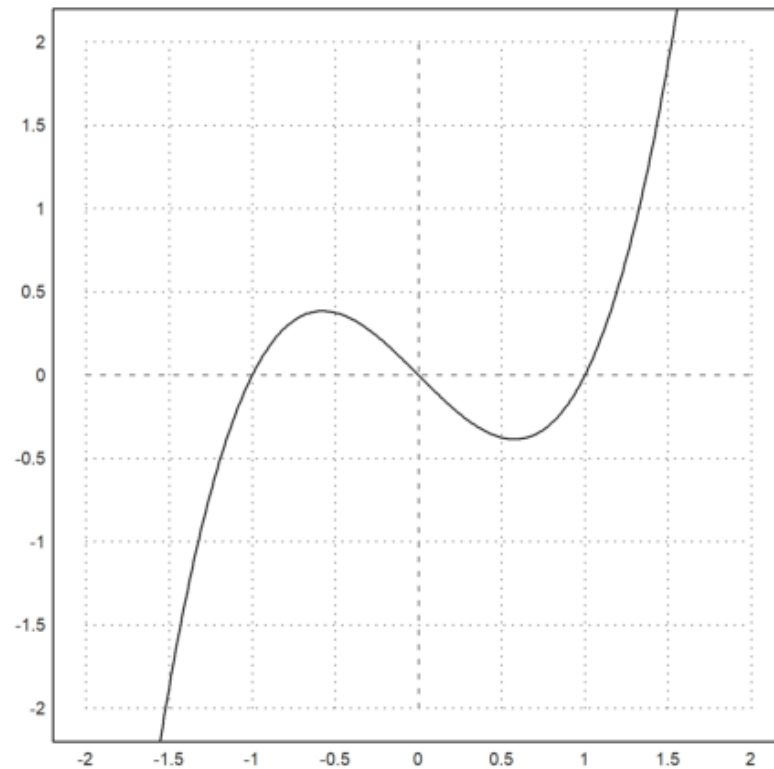
Perhatikan bahwa x^x tidak terdefinisi untuk $x \leq 0$. Fungsi `plot2d` mendeteksi kesalahan ini, dan mulai memplot segera setelah fungsi terdefinisi. Ini berlaku untuk semua fungsi yang mengembalikan NAN di luar rentang definisinya.

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

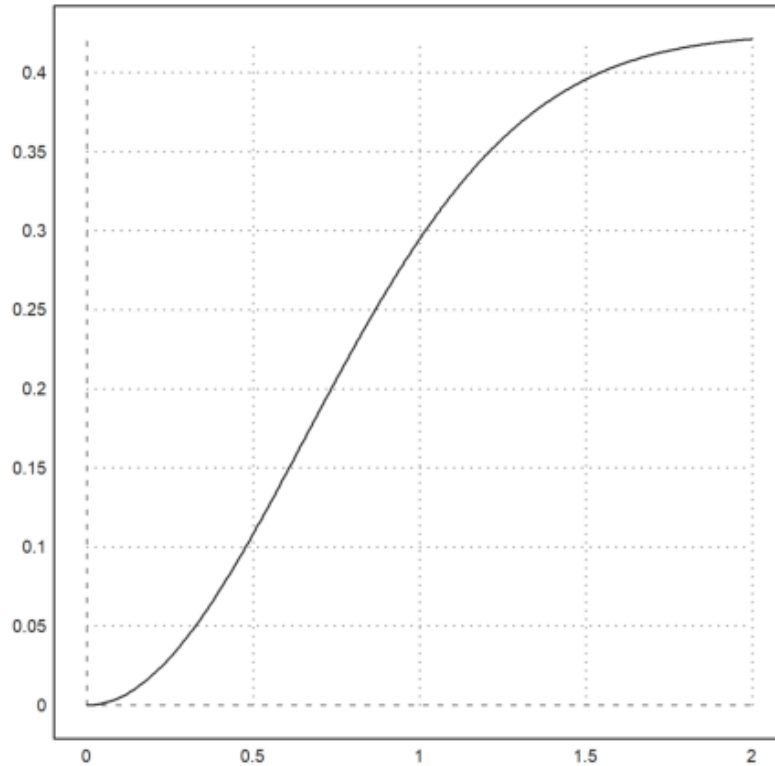


Parameter `square=true` (atau `>square`) memilih rentang-y secara otomatis sehingga hasilnya adalah jendela plot persegi. Perhatikan bahwa secara default, Euler menggunakan ruang persegi di dalam jendela plot.

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

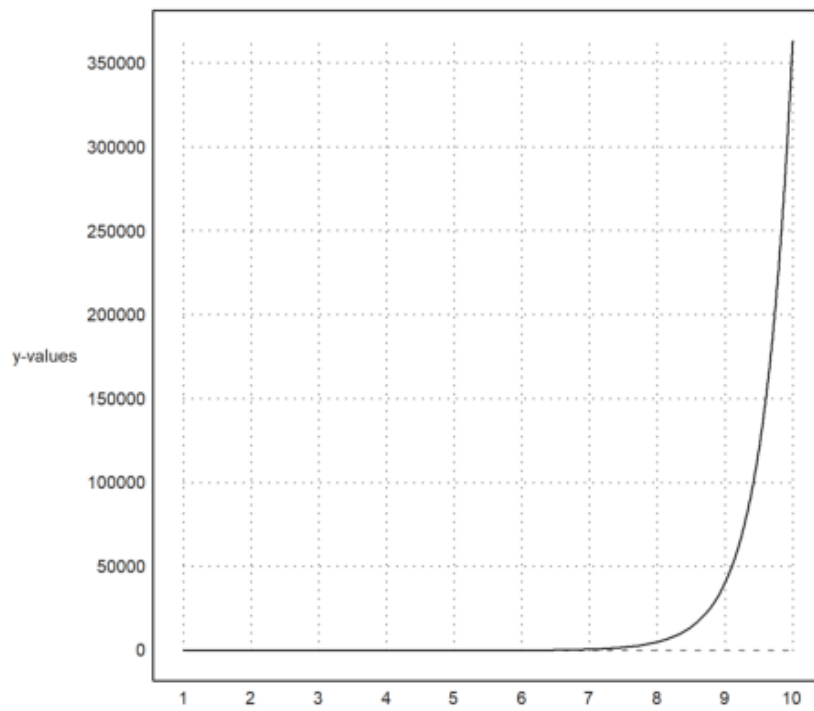


```
>plot2d('integrate("sin(x)*exp(-x^2)","0,x"',0,2): // plot integral
```



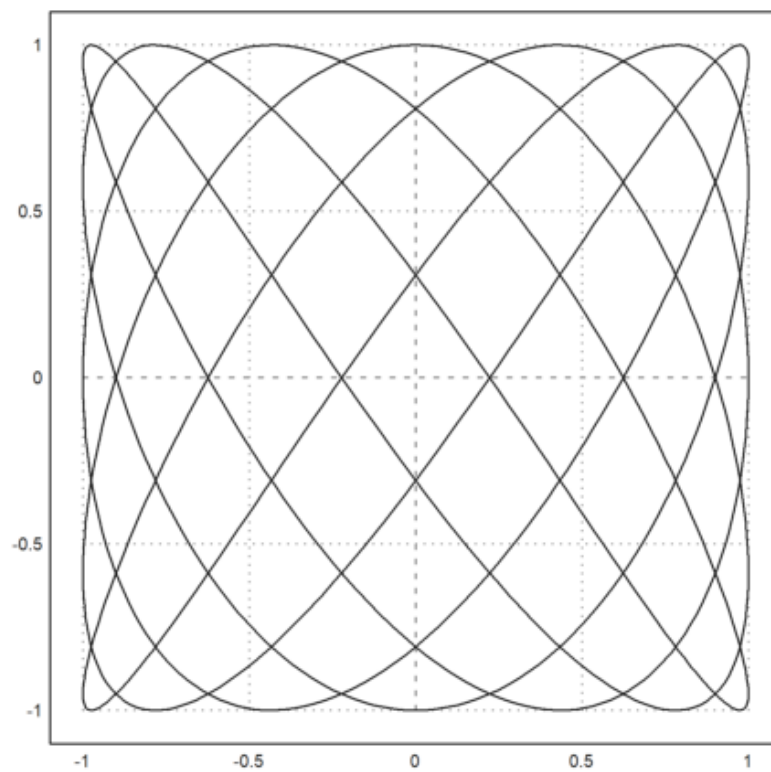
Jika Anda memerlukan lebih banyak ruang untuk label-y, panggil `shrinkwindow()` dengan parameter yang lebih kecil, atau tetapkan nilai positif untuk "lebih kecil" di `plot2d()`.

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

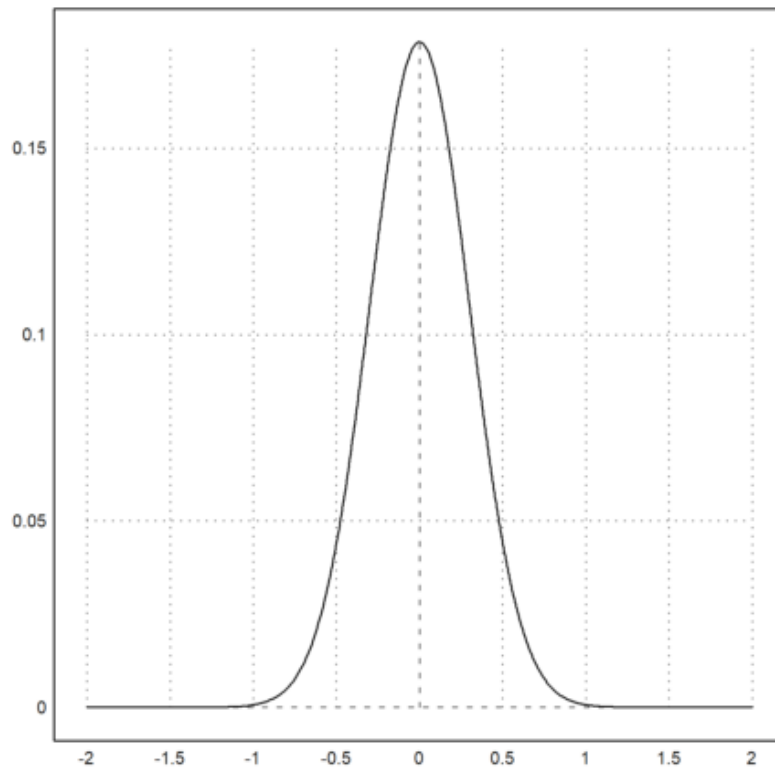


Ekspresi simbolik juga dapat digunakan, karena disimpan sebagai ekspresi string sederhana.

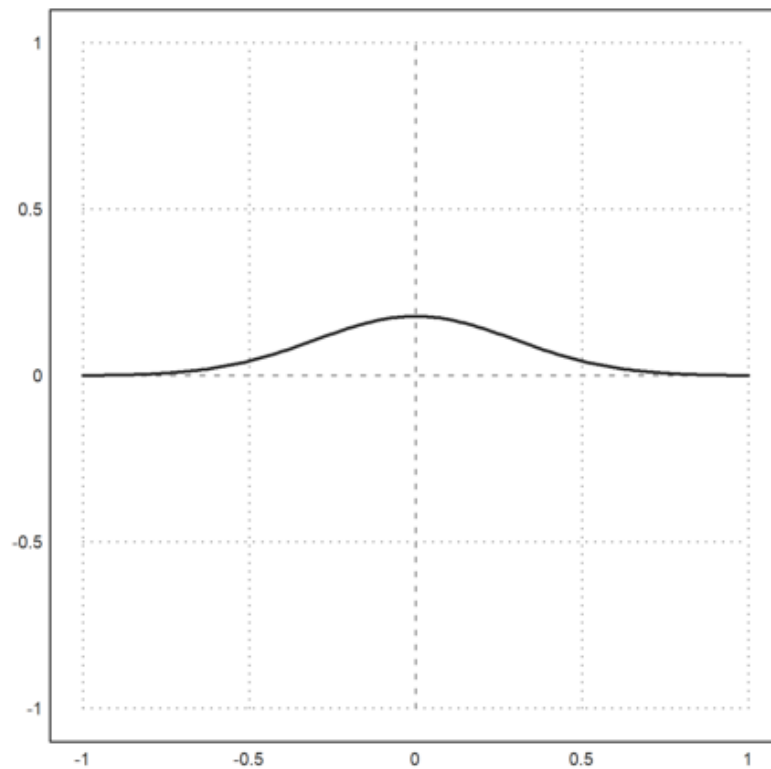
```
>x=linspace(0,2pi,1000); plot2d(sin(5x),cos(7x)):
```



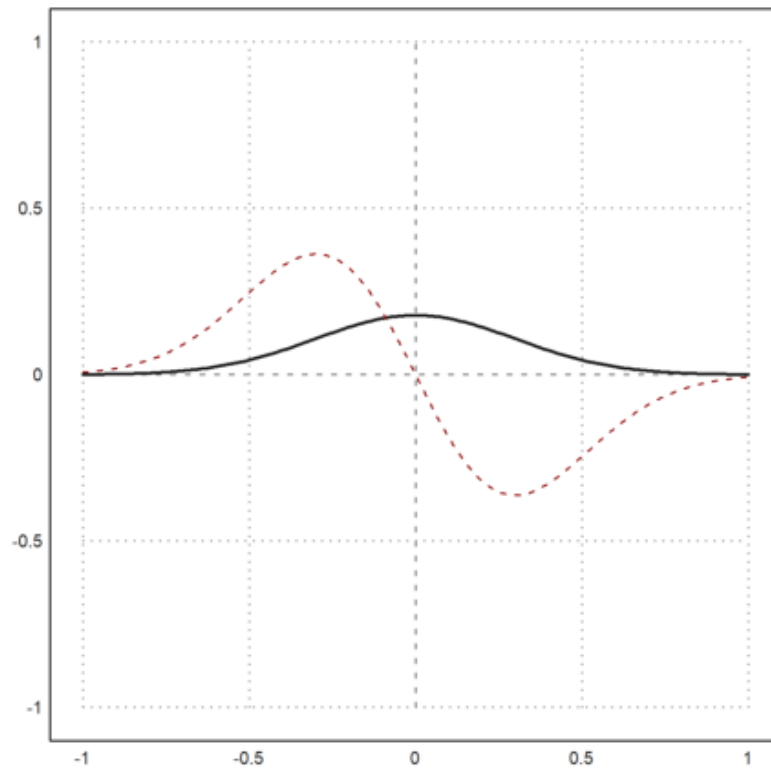
```
>a:=5.6; expr &= exp(-a*x^2)/a; // define expression
>plot2d(expr,-2,2): // plot from -2 to 2
```



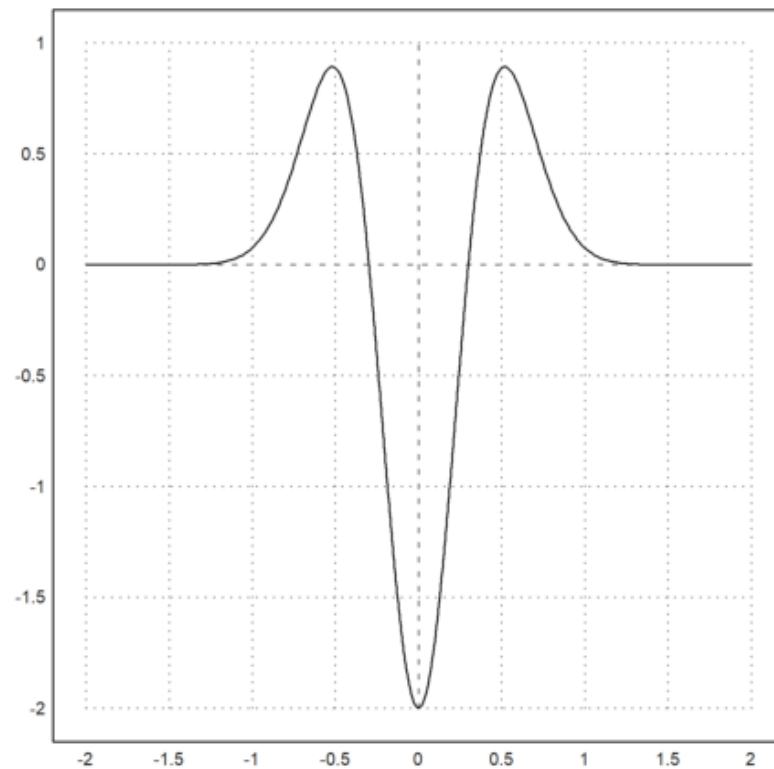
```
>plot2d(expr,r=1,thickness=2): // plot in a square around (0,0)
```



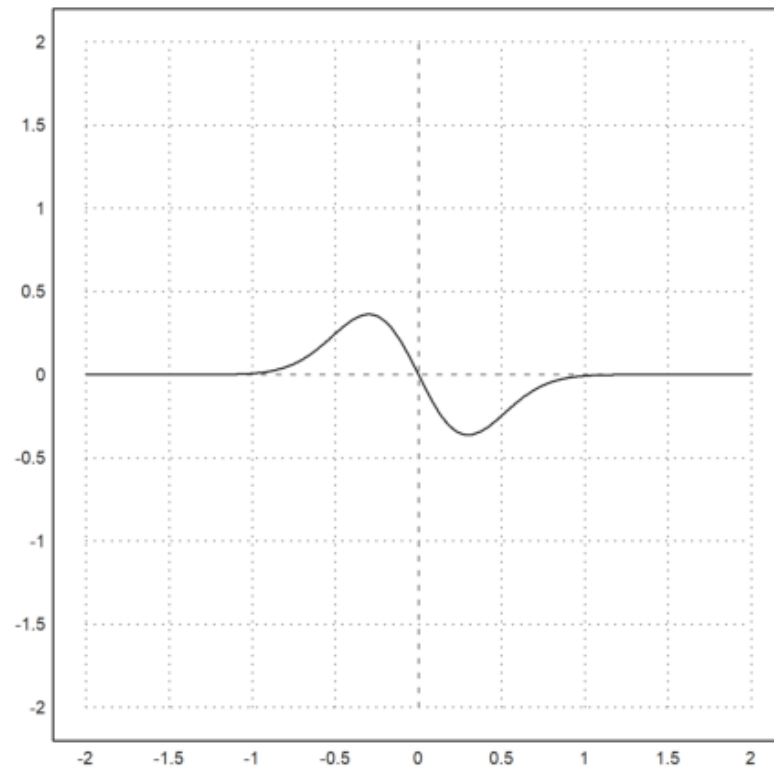
```
>plot2d(&diff(expr,x),>add,style="--",color=red): // add another plot
```



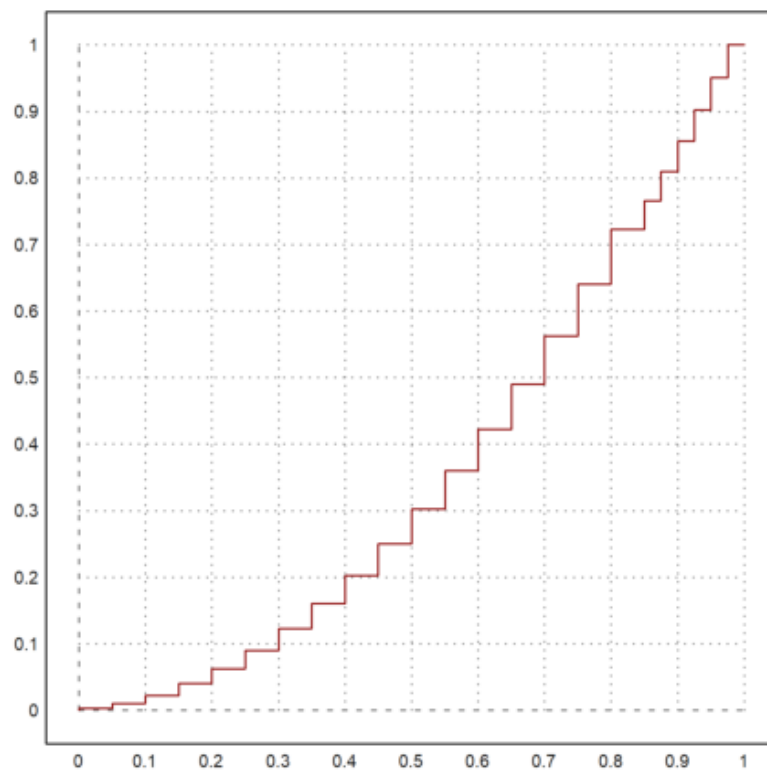
```
>plot2d(&diff(expr,x,2),a=-2,b=2,c=-2,d=1): // plot in rectangle
```



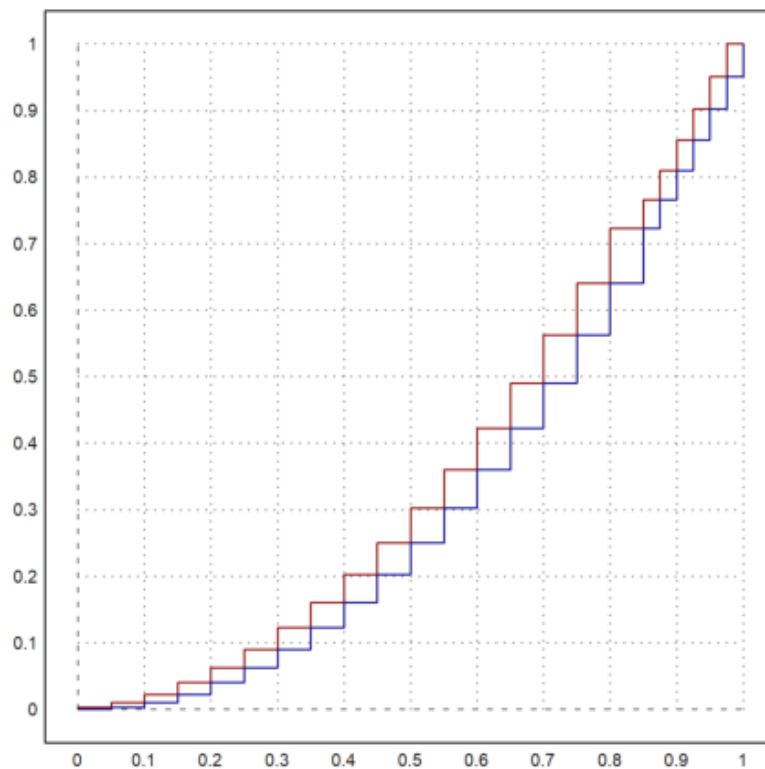
```
>plot2d(&diff(expr,x),a=-2,b=2,>square): // keep plot square
```



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



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

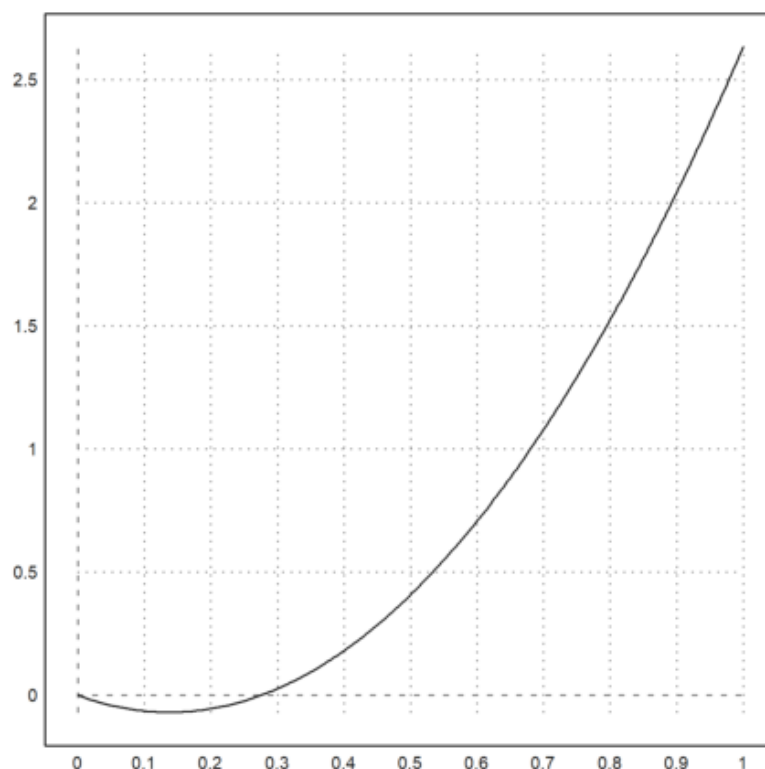


Fungsi dalam satu Parameter

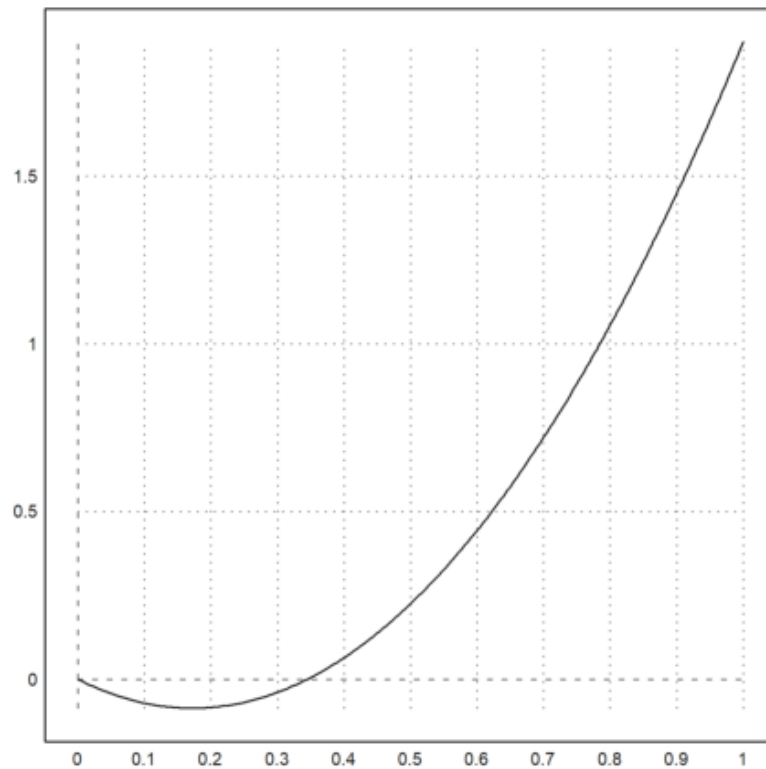
Fungsi plotting terpenting untuk plot planar adalah `plot2d()`. Fungsi ini diimplementasikan dalam bahasa Euler dalam berkas "plot.e", yang dimuat di awal program.

Berikut beberapa contoh penggunaan fungsi. Seperti biasa dalam EMT, untuk fungsi yang berfungsi untuk fungsi atau ekspresi lain, Anda dapat meneruskan parameter tambahan (selain x) yang bukan variabel global ke fungsi tersebut dengan parameter titik koma atau dengan koleksi pemanggilan.

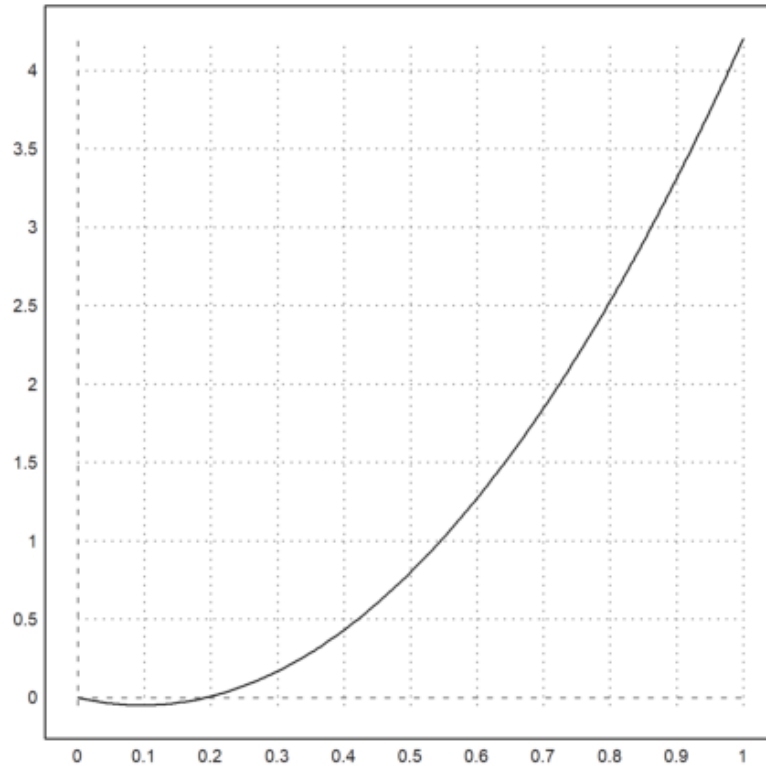
```
>function f(x,a) := x^2/a+a*x^2-x; // define a function
>a=0.3; plot2d("f",0,1;a): // plot with a=0.3
```



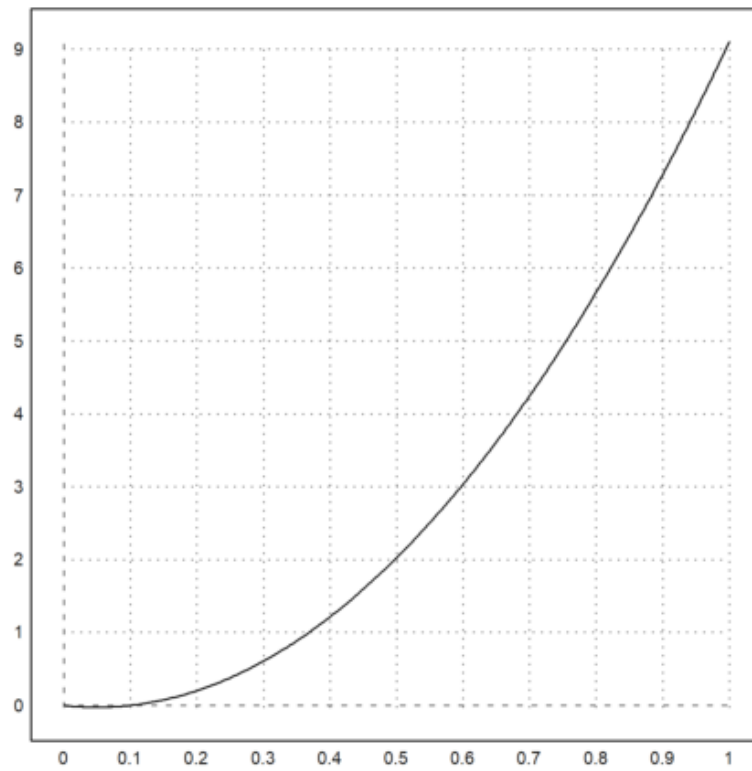
```
>plot2d("f",0,1;0.4): // plot with a=0.4
```



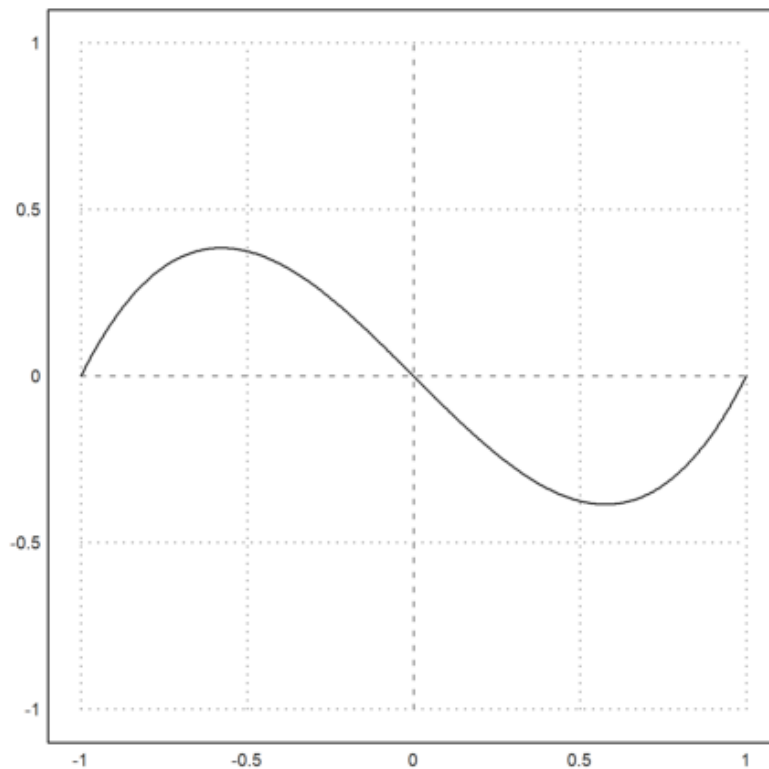
```
>plot2d({"f",0.2}),0,1): // plot with a=0.2
```



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



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



Berikut ringkasan fungsi yang diterima

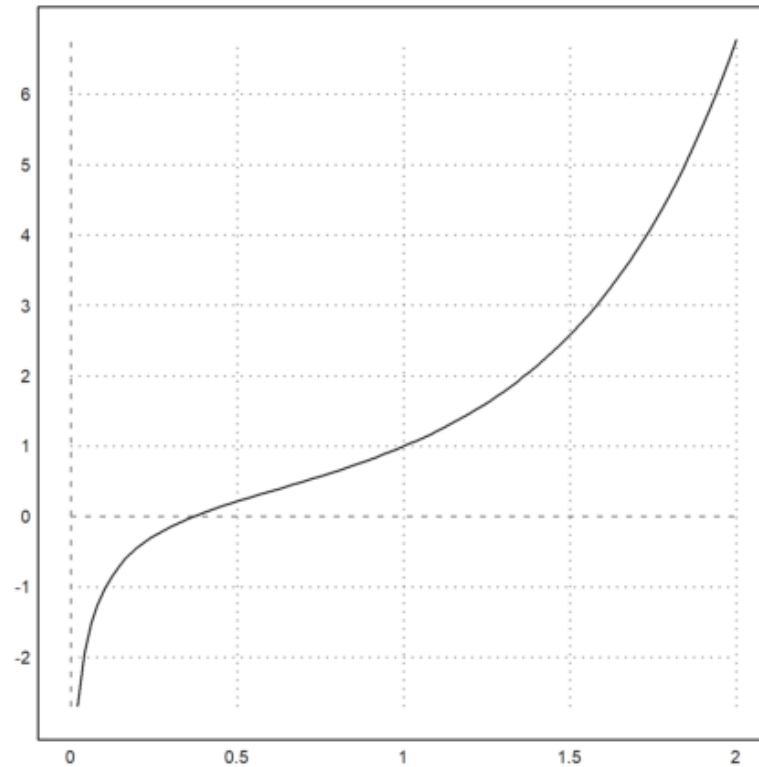
- ekspresi atau ekspresi simbolik dalam x
- fungsi atau fungsi simbolik dengan nama seperti "f"
- fungsi simbolik hanya dengan nama f

Fungsi plot2d() juga menerima fungsi simbolik. Untuk fungsi simbolik, hanya namanya saja yang berfungsi.

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

$$x^x (\log(x) + 1)$$

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

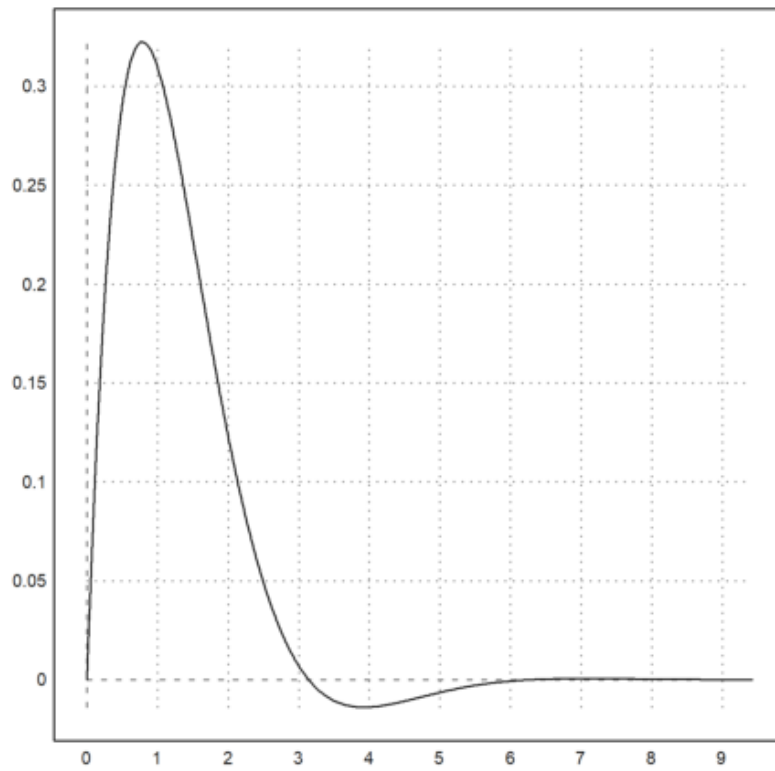


Tentu saja, untuk ekspresi atau ungkapan simbolik, nama variabel sudah cukup untuk memplotnya.

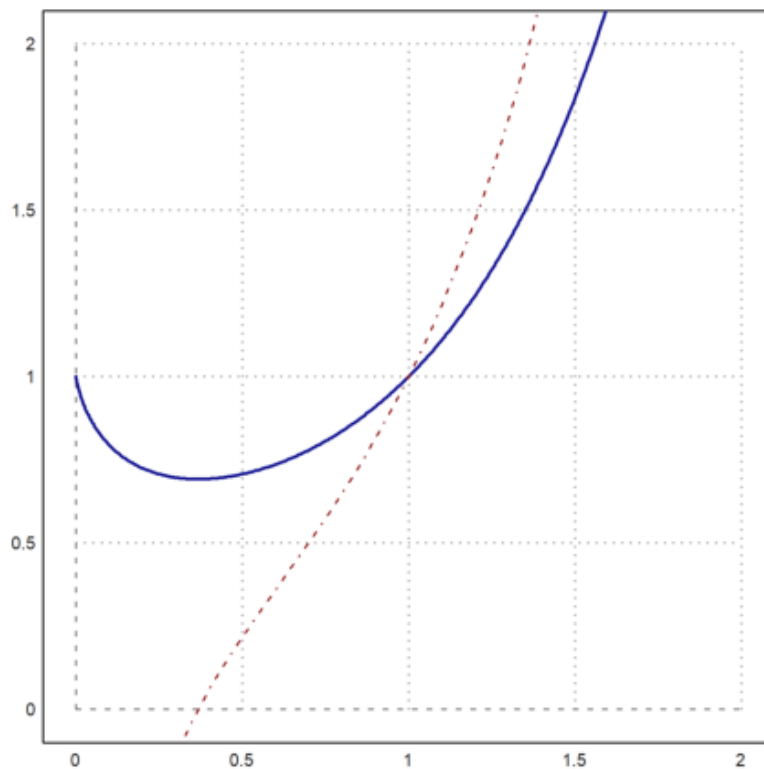
```
>expr &= sin(x)*exp(-x)
```

$$e^{-x} \sin(x)$$

```
>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="-.-") :
```



Terdapat berbagai pilihan untuk gaya garis.

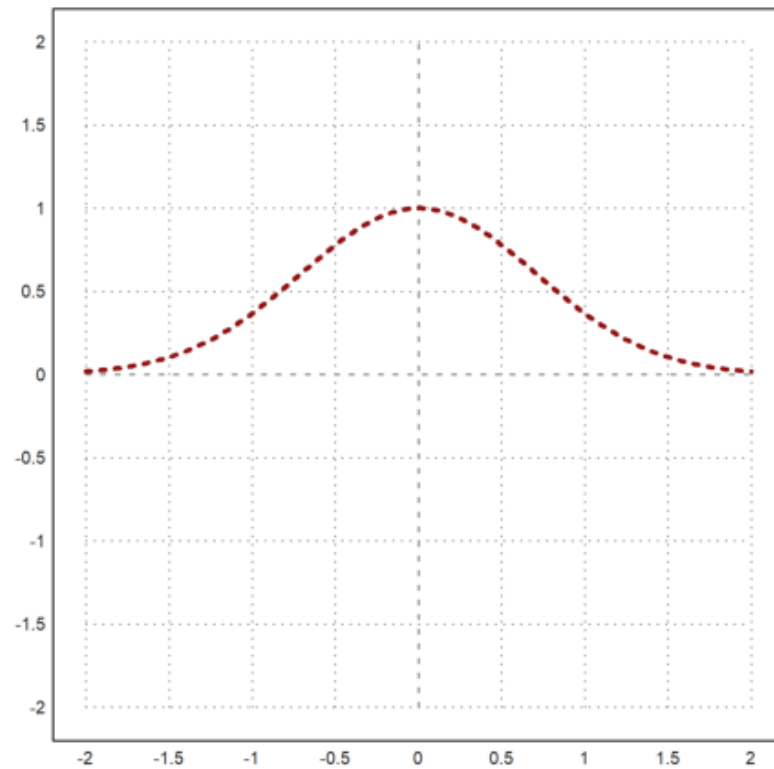
- style="...". Pilih dari "-", "--", ":", ".", "-.-", "-.-".
- color: Lihat di bawah untuk warna.
- thickness: Nilai default adalah 1.

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

- 0..15: indeks warna default.

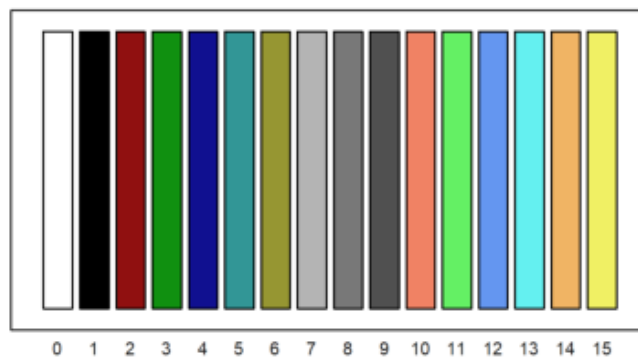
- konstanta warna: putih, hitam, merah, hijau, biru, cyan, zaitun, abu-abu muda, abu-abu, abu-abu tua, oranye, hijau muda, biru kehijauan, biru muda, oranye muda, kuning
- `rgb(red,green,blue)`: parameternya adalah bilangan real dalam $[0,1]$.

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



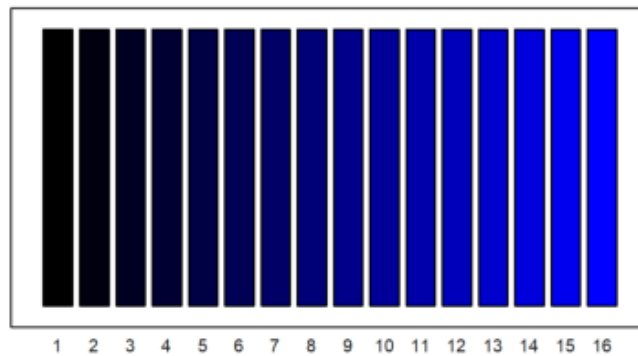
Berikut ini adalah tampilan warna EMT yang telah ditetapkan sebelumnya.

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



Namun Anda dapat menggunakan warna apa pun.

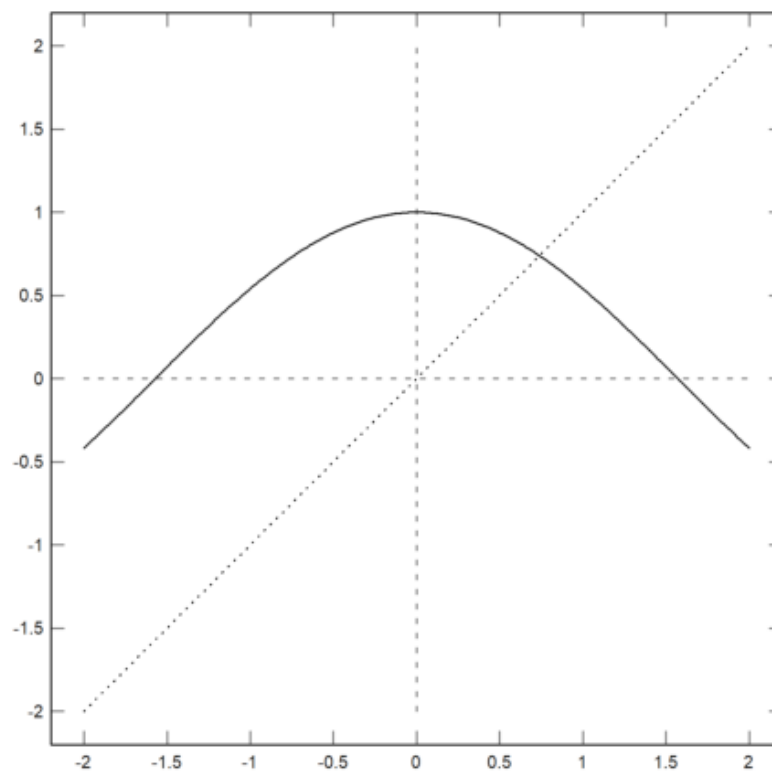
```
>columnsplot(ones(1,16),grid=0,color=rgb(0,0,linspace(0,1,15))):
```



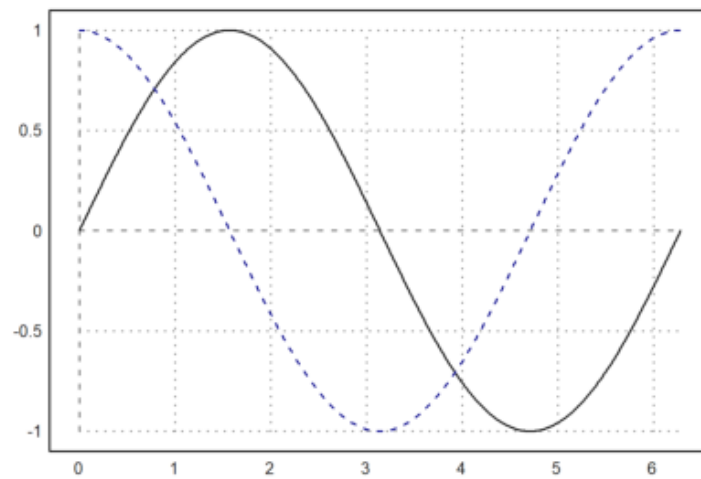
Menggambar beberapa kurva pada bidang koordinat yang sama

Memplot lebih dari satu fungsi (beberapa fungsi) ke dalam satu jendela dapat dilakukan dengan berbagai cara. Salah satu metodenya adalah menggunakan `>add` untuk beberapa panggilan ke `plot2d` secara bersamaan, kecuali panggilan pertama. Kami telah menggunakan fitur ini pada contoh di atas.

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

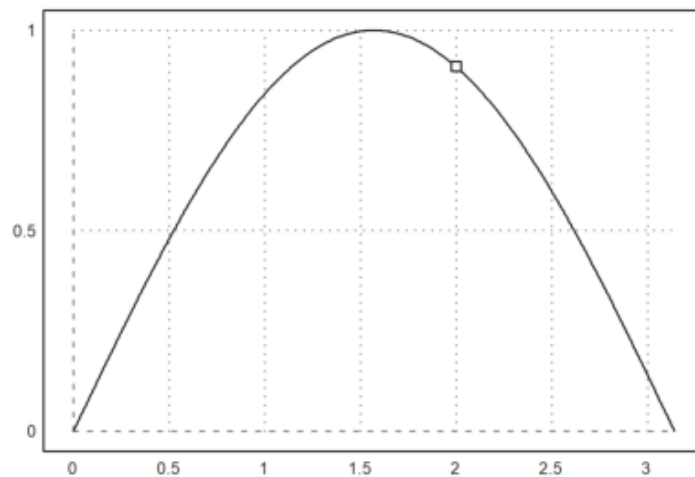


```
>aspect(1.5); plot2d("sin(x)",0,2pi); plot2d("cos(x)",color=blue,style="--",>add):
```



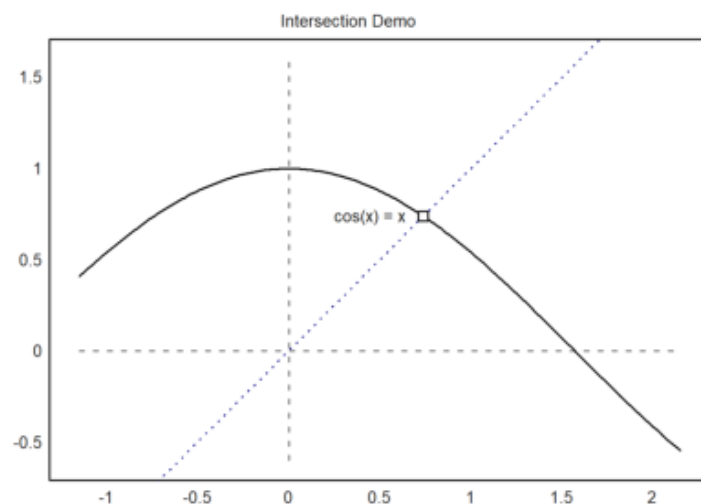
Salah satu kegunaan `>add` adalah untuk menambahkan titik pada kurva.

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



Kami menambahkan titik potong dengan label (pada posisi "cl" untuk kiri tengah), dan memasukkan hasilnya ke dalam buku catatan. Kami juga menambahkan judul pada plot.

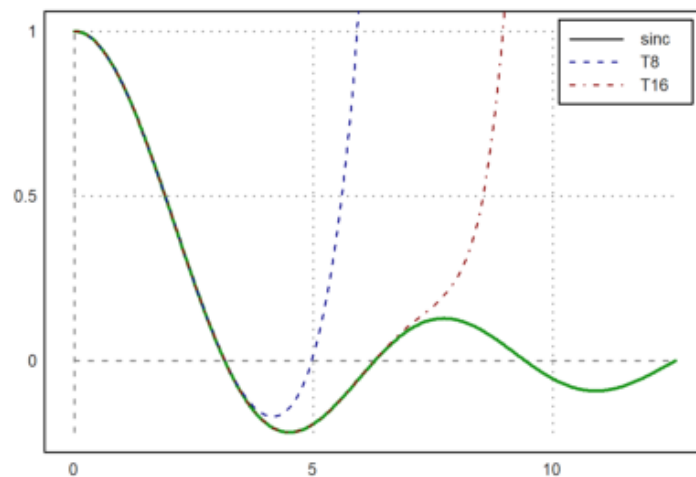
```
>plot2d(["cos(x)", "x"],r=1.1,cx=0.5,cy=0.5, ...
  color=[black,blue],style=["-", "."], ...
  grid=1);
>x0=solve("cos(x)-x",1); ...
  plot2d(x0,x0,>points,>add,title="Intersection Demo"); ...
  label("cos(x) = x",x0,x0,pos="cl",offset=20):
```



Dalam demo berikut, kami memplot fungsi $\text{sinc}(x) = \sin(x)/x$ dan ekspansi Taylor ke-8 dan ke-16. Kami menghitung ekspansi ini menggunakan Maxima melalui ekspresi simbolik. Pemplot ini dilakukan dengan perintah multi-baris berikut dengan tiga panggilan ke `plot2d()`. Perintah kedua dan ketiga memiliki tanda `>add` yang disetel, yang membuat plot menggunakan rentang sebelumnya.

Kami menambahkan kotak label yang menjelaskan fungsi-fungsi tersebut.

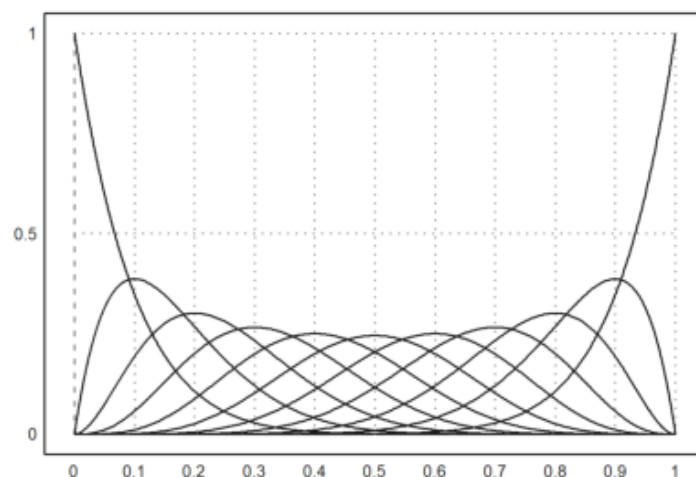
```
>$taylor(sin(x)/x,x,0,4)
>plot2d("sinc(x)",0,4pi,color=green,thickness=2); ...
  plot2d(&taylor(sin(x)/x,x,0,8),>add,color=blue,style="--"); ...
  plot2d(&taylor(sin(x)/x,x,0,16),>add,color=red,style="-.-"); ...
  labelbox(["sinc","T8","T16"],styles=["-","--","-.-"], ...
    colors=[black,blue,red]):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



Dalam contoh berikut, kami menghasilkan Polinomial Bernstein.

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

```
>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;
```

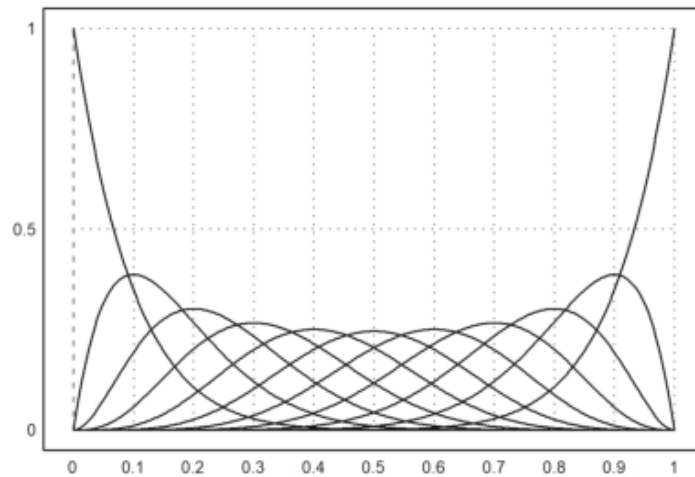


Metode kedua menggunakan pasangan matriks nilai-x dan matriks nilai-y dengan ukuran yang sama.

Kita menghasilkan matriks nilai-nilai dengan satu Polinomial Bernstein di setiap barisnya. Untuk ini, kita cukup menggunakan vektor kolom i . Silakan lihat pengantar tentang bahasa matriks untuk mempelajari lebih lanjut.

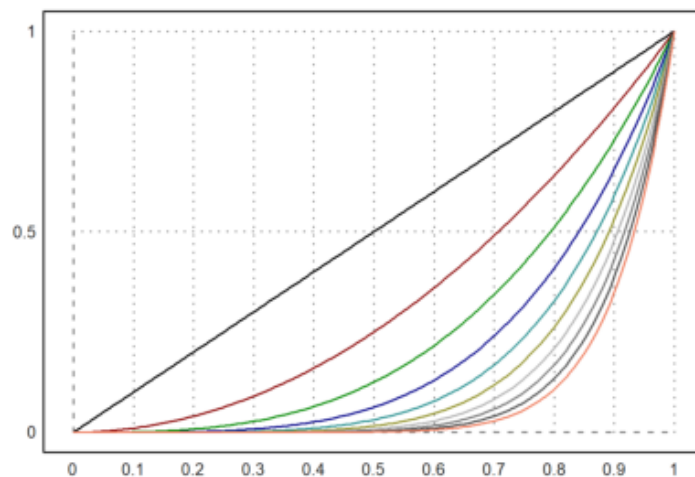
```
>x=linspace(0,1,500);
>n=10; k=(0:n)'; // n is row vector, k is column vector
```

```
>y=bin(n,k)*x^k*(1-x)^(n-k); // y is a matrix then
>plot2d(x,y):
```



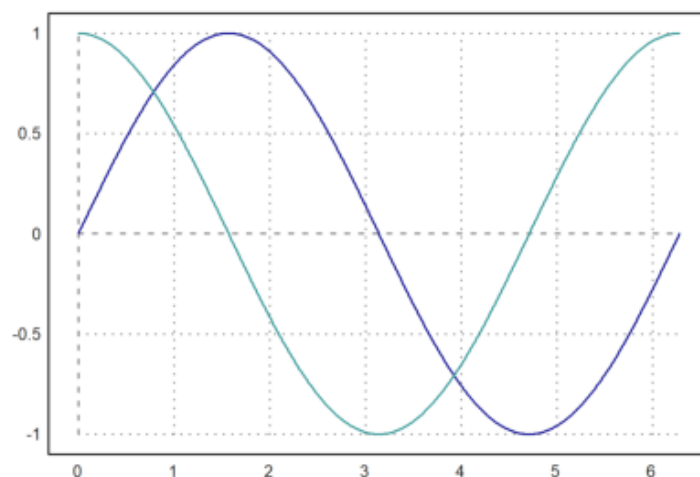
Perhatikan bahwa parameter warna dapat berupa vektor. Kemudian, setiap warna digunakan untuk setiap baris matriks.

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

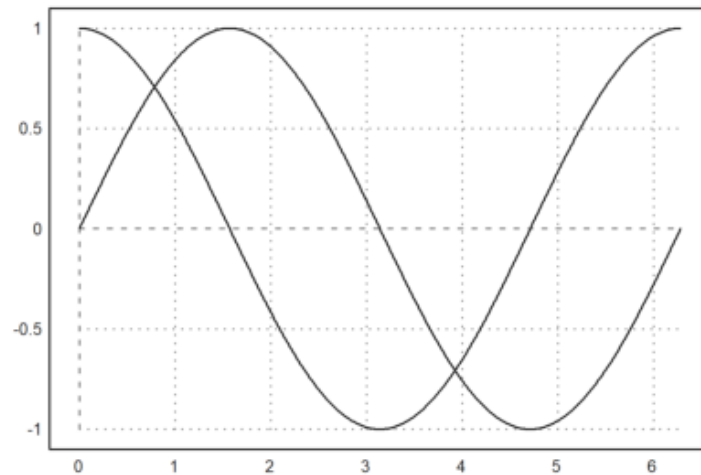


Metode lain adalah menggunakan vektor ekspresi (string). Anda kemudian dapat menggunakan larik warna, larik gaya, dan larik ketebalan dengan panjang yang sama.

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



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



Kita bisa mendapatkan vektor tersebut dari Maxima menggunakan makelist() dan mxm2str().

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

```

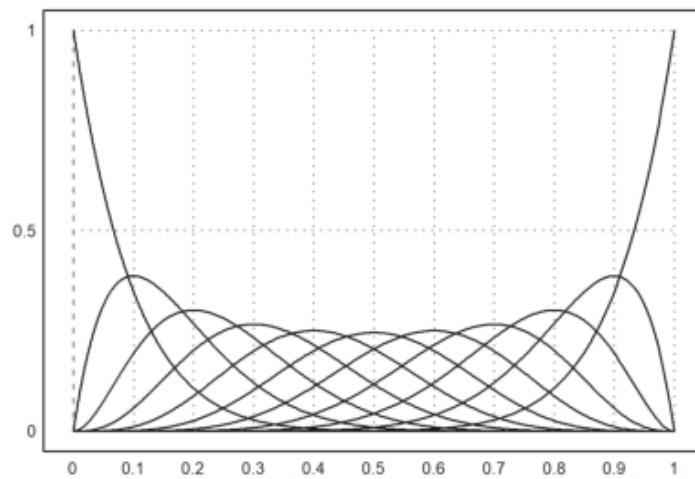
      10      9      8 2      7 3
      [(1 - x) , 10 (1 - x) x, 45 (1 - x) x , 120 (1 - x) x ,
      6 4      5 5      4 6      3 7
210 (1 - x) x , 252 (1 - x) x , 210 (1 - x) x , 120 (1 - x) x ,
      2 8      9 10
45 (1 - x) x , 10 (1 - x) x , x ]
```

```
>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
```

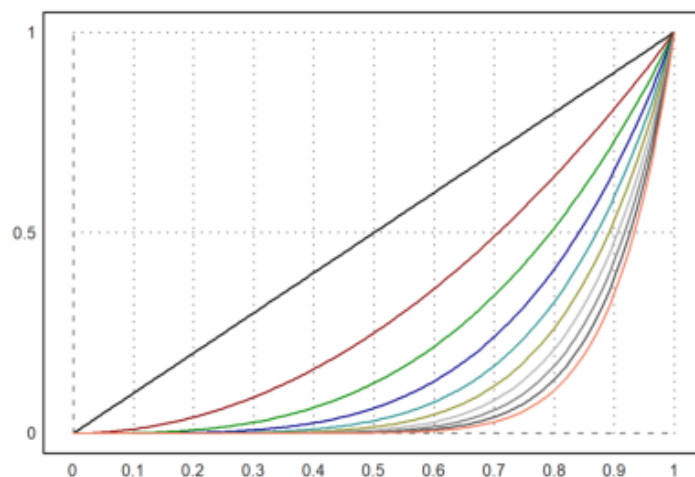


Alternatif lain adalah menggunakan bahasa matriks Euler.

Jika suatu ekspresi menghasilkan matriks fungsi, dengan satu fungsi di setiap baris, semua fungsi ini akan diplot menjadi satu plot.

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

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

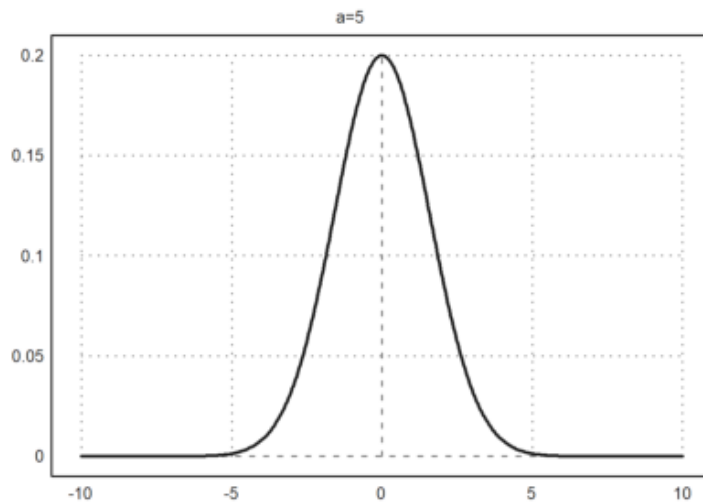


Ekspresi dan fungsi satu baris dapat melihat variabel global.

Jika Anda tidak dapat menggunakan variabel global, Anda perlu menggunakan fungsi dengan parameter tambahan, dan meneruskan parameter ini sebagai parameter titik koma.

Berhati-hatilah untuk meletakkan semua parameter yang ditetapkan di akhir perintah plot2d. Dalam contoh ini, kita meneruskan a=5 ke fungsi f, yang kita plot dari -10 hingga 10.

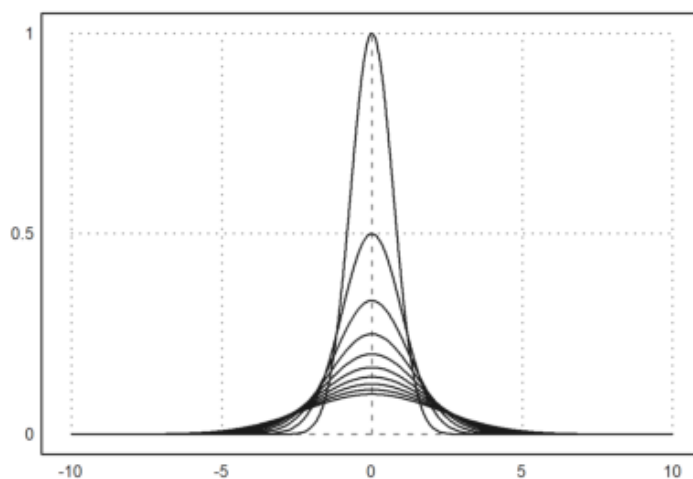
```
>function f(x,a) := 1/a*exp(-x^2/a); ...
plot2d("f",-10,10;5,thickness=2,title="a=5"):
```



Alternatifnya, gunakan koleksi dengan nama fungsi dan semua parameter tambahan. Daftar khusus ini disebut koleksi panggilan, dan merupakan cara yang lebih disukai untuk meneruskan argumen ke suatu fungsi yang diteruskan sebagai argumen ke fungsi lain.

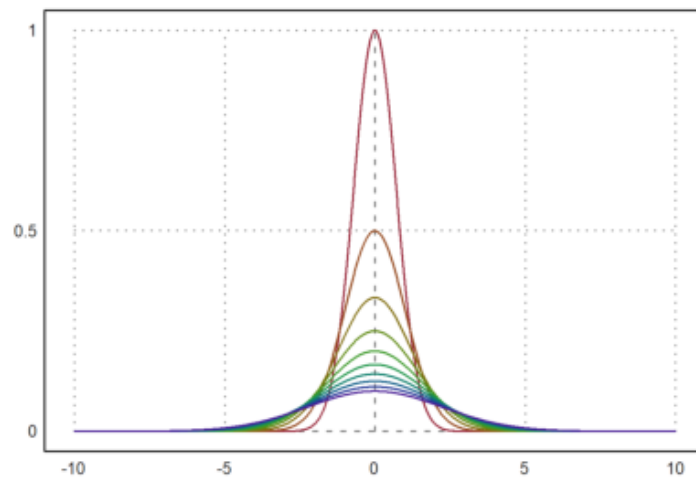
Dalam contoh berikut, kita menggunakan perulangan untuk memplot beberapa fungsi (lihat tutorial tentang pemrograman untuk perulangan).

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



Kita bisa mendapatkan hasil yang sama dengan cara berikut menggunakan bahasa matriks EMT. Setiap baris matriks $f(x,a)$ adalah satu fungsi. Selain itu, kita bisa mengatur warna untuk setiap baris matriks. Klik dua kali pada fungsi `getspectral()` untuk penjelasannya.

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



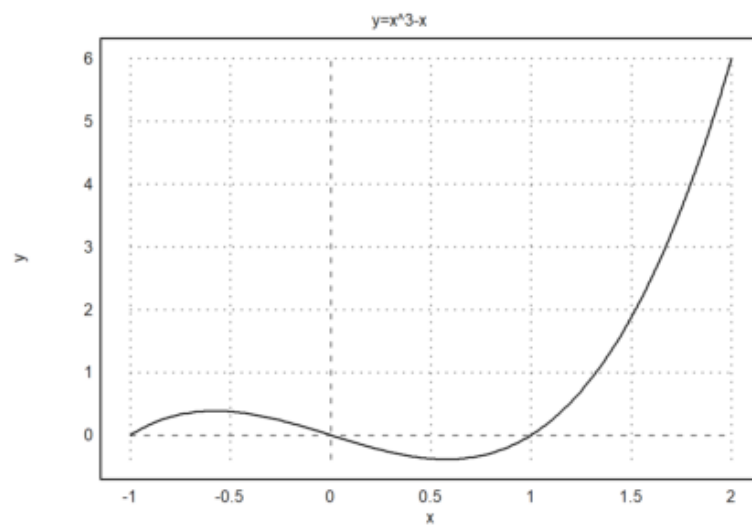
Label Teks

Dekorasi sederhana dapat berupa

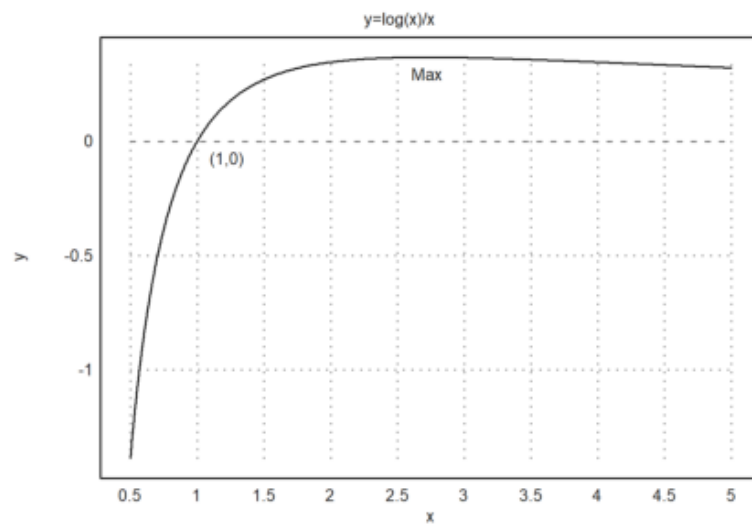
- judul dengan `title="..."`
- label x dan y dengan `xl="...", yl="..."`
- label teks lain dengan `label("...",x,y)`

Perintah label akan memplot ke dalam plot saat ini pada koordinat plot (x,y). Perintah ini dapat menerima argumen posisi.

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

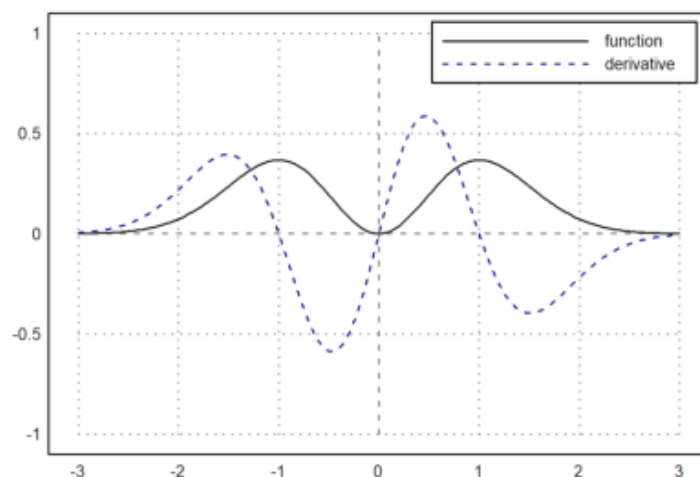


```
>expr := "log(x)/x"; ...
plot2d(expr,0.5,5,title="y="+expr,xl="x",yl="y"); ...
label("(1,0)",1,0); label("Max",E,expr(E),pos="lc") :
```



Ada juga fungsi `labelbox()`, yang dapat menampilkan fungsi dan teks. Fungsi ini menerima vektor string dan warna, satu item 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):
```

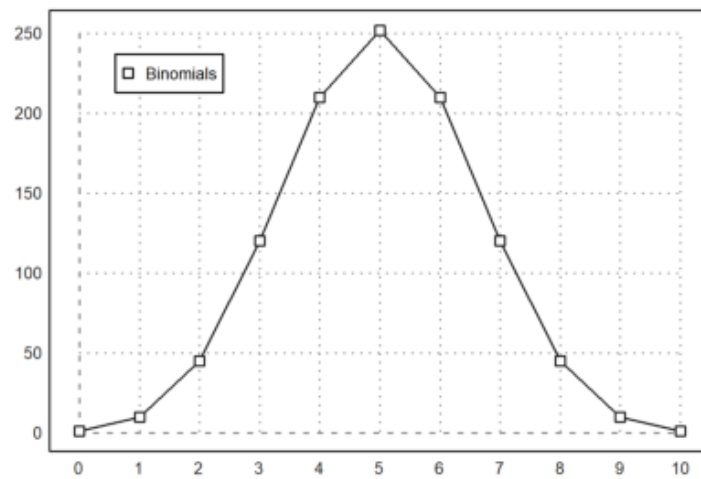


Kotak tersebut dijangkarkan di kanan atas secara default, tetapi `>left` menjangkarkannya di kiri atas. Anda dapat memindahkannya ke tempat mana pun yang Anda inginkan. Posisi jangkar berada di pojok kanan atas kotak, dan angka-angkanya merupakan pecahan dari ukuran jendela grafik. Lebaranya otomatis.

Untuk plot titik, kotak label juga berfungsi. Tambahkan parameter `>points`, atau vektor bendera, satu untuk setiap label.

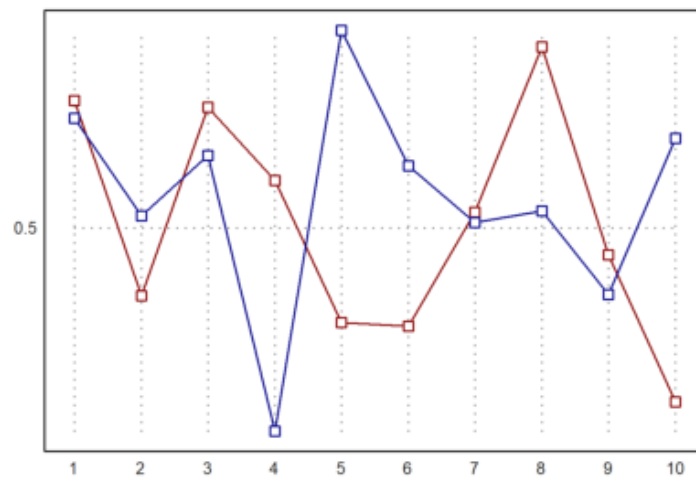
Dalam contoh berikut, hanya ada satu fungsi. Jadi, kita dapat menggunakan string, bukan vektor string. Kita atur warna teks menjadi hitam untuk contoh ini.

```
>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 di `statplot()`. Seperti di `plot2d()`, warna dapat diatur untuk setiap baris plot. Terdapat plot yang lebih khusus untuk keperluan statistik (lihat tutorial tentang statistik).

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



Fitur serupa adalah fungsi `textbox()`.

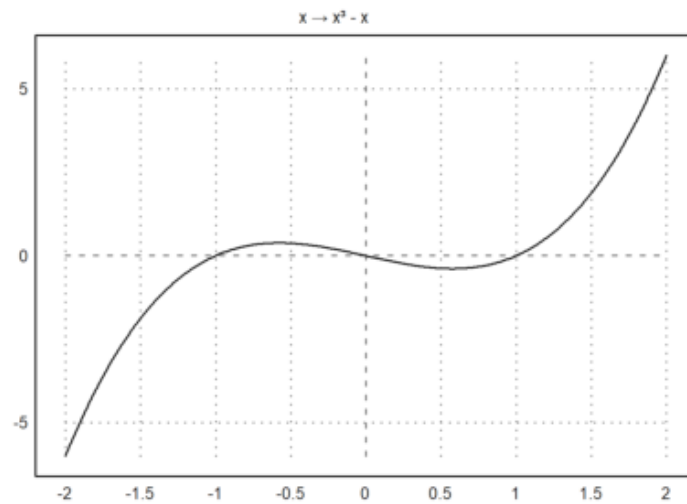
Lebarinya secara default adalah lebar maksimal baris teks. Namun, lebarinya juga dapat diatur oleh pengguna.

```
>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):
```

```
Error with Latex formula
\text{Example of a damped oscillation}\ f(x)=e^{-x}\sin(2\pi x)
texpng:
  return _texpng(latex,usealias,factor,backcolor,color);
Try "trace errors" to inspect local variables after errors.
latex:
  return loadrgb(texpng(s,=color,=factor));
Error in:
... f a damped oscillation}\ f(x)=e^{-x}\sin(2\pi x)"),w=0.85): ...
      ^
```

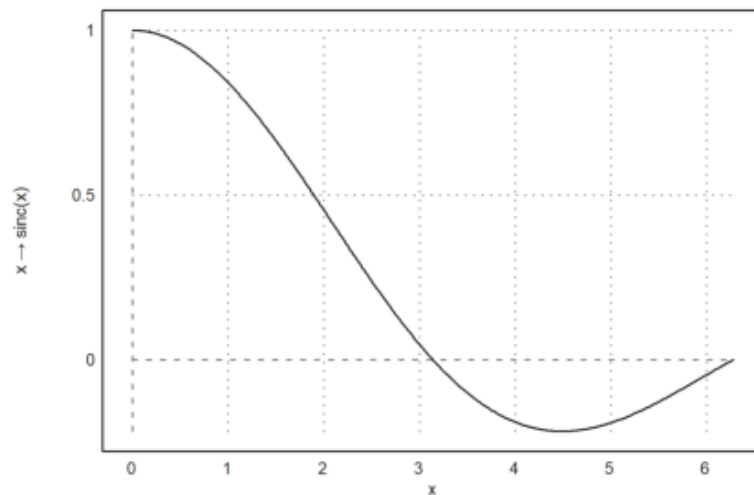
Label teks, judul, kotak label, dan teks lainnya dapat berisi string Unicode (lihat sintaksis EMT untuk informasi lebih lanjut tentang string Unicode).

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



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

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



LaTeX

Anda juga dapat memplot rumus LaTeX jika Anda telah menginstal sistem LaTeX. Saya merekomendasikan MiKTeX. Jalur ke biner "latex" dan "dvi2png" harus berada di jalur sistem, atau Anda harus mengatur LaTeX di menu opsi.

Perhatikan bahwa penguraian LaTeX lambat. Jika Anda ingin menggunakan LaTeX dalam plot animasi, Anda harus memanggil latex() sebelum loop sekali dan menggunakan hasilnya (gambar dalam matriks RGB).

Dalam plot berikut, kami menggunakan LaTeX untuk label x dan y, label, kotak label, dan 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):
```

```
Error with Latex formula
\text{Function }\Phi$
texpng:
  return _texpng(latex,usealias,factor,backcolor,color);
Try "trace_errors" to inspect local variables after errors.
latex:
  return loadrgb(texpng(s,=color,=factor));
Error in:
```

```
... lor=blue, title=latex("\text{Function $\Phi$}"), xl=latex( ...
```

Seringkali, kita menginginkan spasi non-konformal dan label teks pada sumbu x. Kita dapat menggunakan `xaxis()` dan `yaxis()` seperti yang akan kita bahas nanti.

Cara termudah adalah membuat plot kosong dengan bingkai menggunakan `grid=4`, lalu menambahkan grid dengan `ygrid()` dan `xgrid()`. Dalam 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):
```

```
Error with Latex formula
0
texpng:
return _texpng(latex,usealias,factor,backcolor,color);
latex:
return loadrgb(texpng(s,=color,=factor));
Try "trace errors" to inspect local variables after errors.
xtick:
elseif latex then str=latex(s{#}); d=0.6;
```

Tentu saja, fungsi juga dapat digunakan.

```
>function map f(x) ...
if x>0 then return x^4
else return x^2
endif
endfunction
```

Parameter "peta" membantu penggunaan fungsi untuk vektor. Untuk plot, parameter ini tidak diperlukan. Namun, untuk menunjukkan bahwa vektorisasi bermanfaat, kami menambahkan beberapa poin kunci pada plot pada $x=-1$, $x=0$, dan $x=1$.

Pada plot berikut, kami juga memasukkan beberapa kode LaTeX. Kami menggunakannya untuk dua label dan satu kotak teks. Tentu saja, Anda hanya dapat menggunakan LaTeX jika Anda telah menginstal LaTeX 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):
```

```
Error with Latex formula
x^3
texpng:
return _texpng(latex,usealias,factor,backcolor,color);
Try "trace errors" to inspect local variables after errors.
latex:
return loadrgb(texpng(s,=color,=factor));
Error in:
... ,1],f([-1,0,1]),>points,>add); label(latex("x^3"),0.72,f(0.72) ...
```

Interaksi Pengguna

Saat memplot fungsi atau ekspresi, parameter `>user` memungkinkan pengguna untuk memperbesar dan menggeser plot dengan tombol kursor atau tetikus. Pengguna dapat

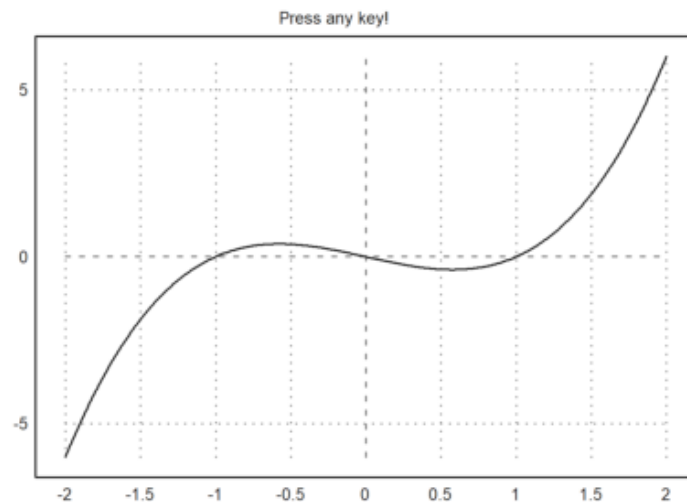
- memperbesar dengan `+` atau `-`
- memindahkan plot dengan tombol kursor
- memilih jendela plot dengan tetikus
- mengatur ulang tampilan dengan spasi

- keluar dengan kembali

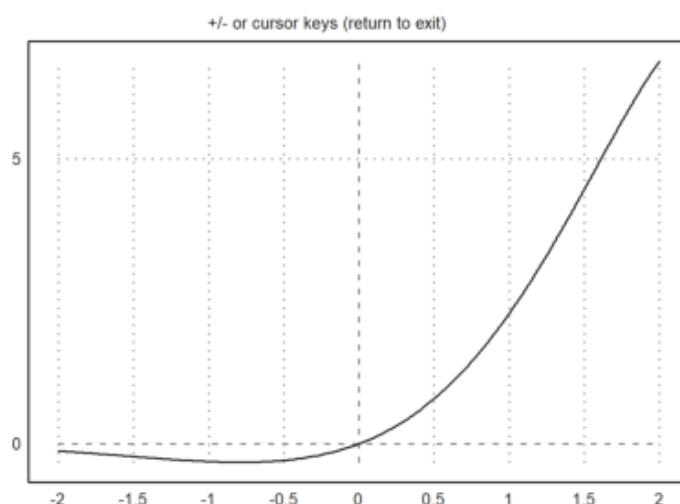
Tombol spasi akan mengatur ulang plot ke jendela plot semula.

Saat memplot data, tanda >user 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)"): 
```



Berikut ini mendemonstrasikan cara interaksi pengguna tingkat lanjut (lihat tutorial pemrograman untuk detailnya).

Fungsi bawaan `mousedrag()` menunggu peristiwa tetikus atau papan ketik. Fungsi ini melaporkan gerakan tetikus ke bawah, gerakan tetikus ke atas, dan penekanan tombol. Fungsi `dragpoints()` memanfaatkan hal ini, dan memungkinkan pengguna menyeret titik mana pun dalam plot.

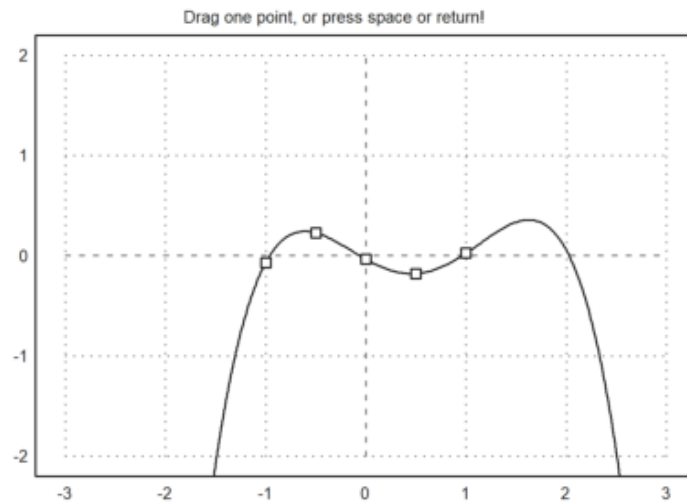
Kita membutuhkan fungsi plot terlebih dahulu. Sebagai contoh, kita menginterpolasi 5 titik dengan polinomial. Fungsi ini harus memplot ke dalam area plot yang tetap.

```
>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 di plot2d (d dan xp), yang diteruskan ke evaluasi fungsi interp(). Tanpa ini, kita harus menulis fungsi plotinterp() terlebih dahulu, yang mengakses nilai-nilai secara global.

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

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



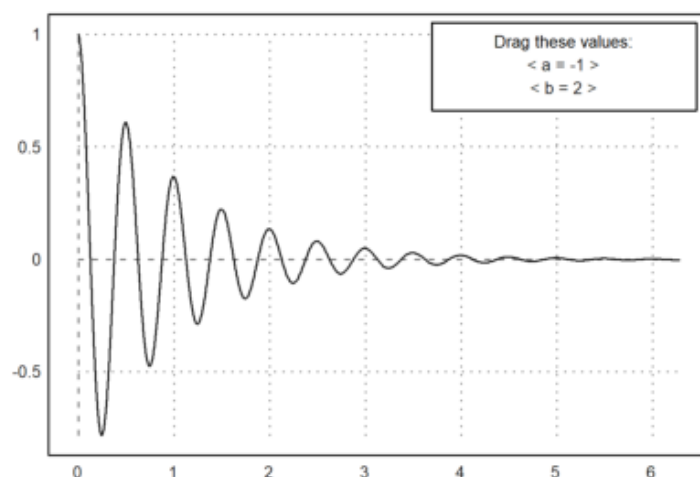
Ada juga fungsi yang memplot fungsi lain berdasarkan vektor parameter, dan memungkinkan pengguna menyesuaikan parameter tersebut.

Pertama, kita membutuhkan fungsi plot.

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

Kemudian, kita membutuhkan nama untuk parameter, nilai awal, dan matriks rentang nx2, atau baris judul. Terdapat slider interaktif yang dapat diatur oleh pengguna. Fungsi dragvalues() menyediakannya.

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

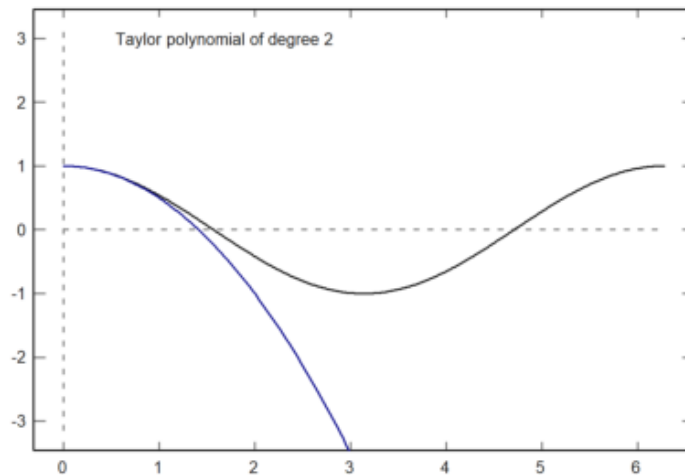


Dimungkinkan untuk membatasi nilai yang diseret ke bilangan bulat. Sebagai contoh, kita menulis fungsi plot, yang memplot polinomial Taylor berderajat n ke fungsi kosinus.

```
>function plotf(n) ...  
plot2d("cos(x)",0,2pi,>square,grid=6);  
plot2d("&#x2191;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 mengizinkan derajat n bervariasi dari 0 hingga 20 dalam 20 stop. Hasil dragvalues() digunakan untuk memplot sketsa dengan n ini, dan untuk memasukkan plot ke dalam buku catatan.

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



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

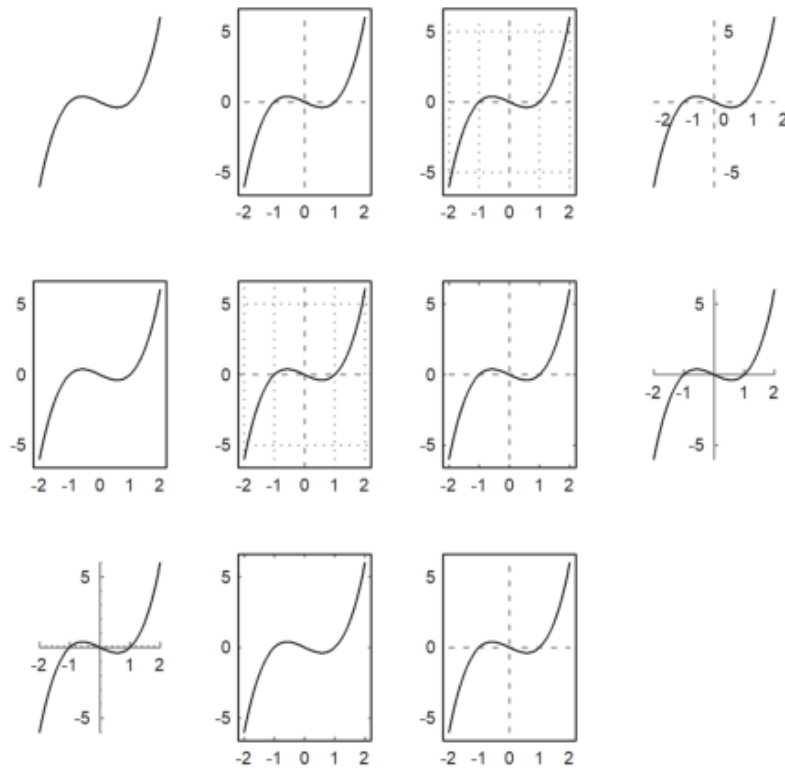
```
>function dragtest ...
    plot2d(none,r=1,title="Drag with the mouse, or press any key!");
    start=0;
    repeat
        {flag,m,time}=mousedrag();
        if flag==0 then return; endif;
        if flag==2 then
            hold on; mark(m[1],m[2]); hold off;
        endif;
    end
endfunction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

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

Gaya Plot 2D

Secara default, EMT menghitung tanda centang sumbu otomatis dan menambahkan label ke setiap tanda centang. Ini dapat diubah dengan parameter grid. Gaya default sumbu dan label dapat diubah. Selain itu, label dan judul dapat ditambahkan secara manual. Untuk mengatur ulang ke gaya default, gunakan `reset()`.

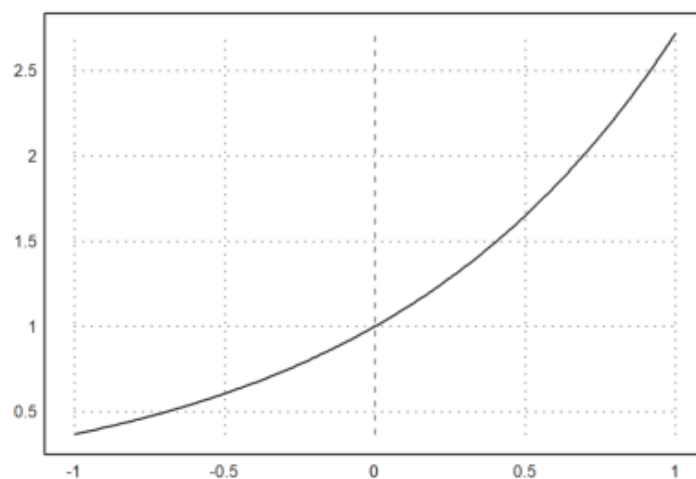
```
>aspect();
>figure(3,4); ...
    figure(1); plot2d("x^3-x",grid=0); ... // no grid, frame or axis
> figure(2); plot2d("x^3-x",grid=1); ... // x-y-axis
> figure(3); plot2d("x^3-x",grid=2); ... // default ticks
> figure(4); plot2d("x^3-x",grid=3); ... // x-y- axis with labels inside
> figure(5); plot2d("x^3-x",grid=4); ... // 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` menonaktifkan frame, dan `framecolor=blue` mengatur frame menjadi warna biru.

Jika Anda menginginkan tanda centang Anda sendiri, Anda dapat menggunakan `style=0`, dan 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
```

Error with Latex formula

y=e^x

texpng:

```
return _texpng(latex,usealias,factor,backcolor,color);
```

Try "trace errors" to inspect local variables after errors.

latex:

```
return loadrgb(texpng(s,=color,=factor));
```

```
Error in:
title(latex("y=e^x")); // title above the plot ...
      ^
```

```
>xlabel(latex("x")); // "x" for x-axis
```

```
Error with Latex formula
x
texpng:
  return _texpng(latex,usealias,factor,backcolor,color);
Try "trace errors" to inspect local variables after errors.
latex:
  return loadrgb(texpng(s,=color,=factor));
Error in:
xlabel(latex("x")); // "x" for x-axis ...
      ^
```

```
>ylabel(latex("y"),>vertical); // vertical "y" for y-axis
```

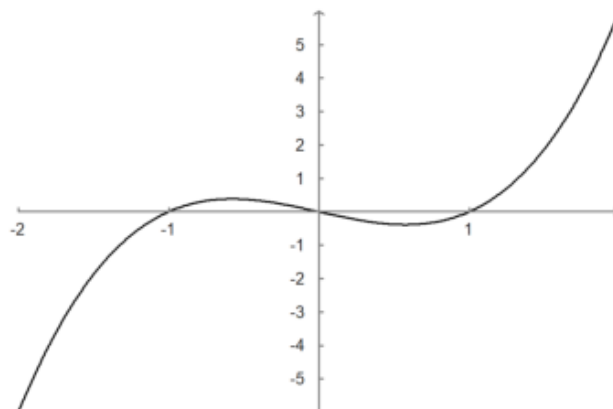
```
Error with Latex formula
y
texpng:
  return _texpng(latex,usealias,factor,backcolor,color);
Try "trace errors" to inspect local variables after errors.
latex:
  return loadrgb(texpng(s,=color,=factor));
Error in:
ylabel(latex("y"),>vertical); // vertical "y" for y-axis ...
      ^
```

```
>label(latex("(0,1)"),0,1,color=blue): // label a point
```

```
Error with Latex formula
(0,1)
texpng:
  return _texpng(latex,usealias,factor,backcolor,color);
Try "trace errors" to inspect local variables after errors.
latex:
  return loadrgb(texpng(s,=color,=factor));
Error in:
label(latex("(0,1)"),0,1,color=blue): // label a point ...
      ^
```

The axis can be drawn separately with `xaxis()` and `yaxis()`.

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

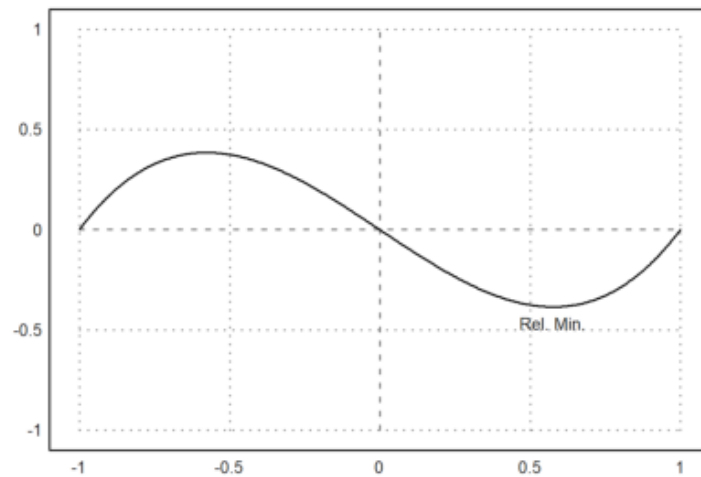


Untuk judul plot dan label sumbu, lihat contoh berikut.

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

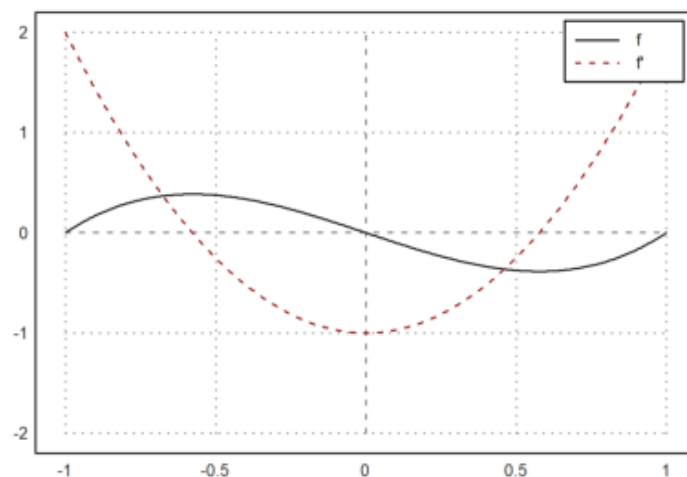
$$x^3 - x$$

```
>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
```

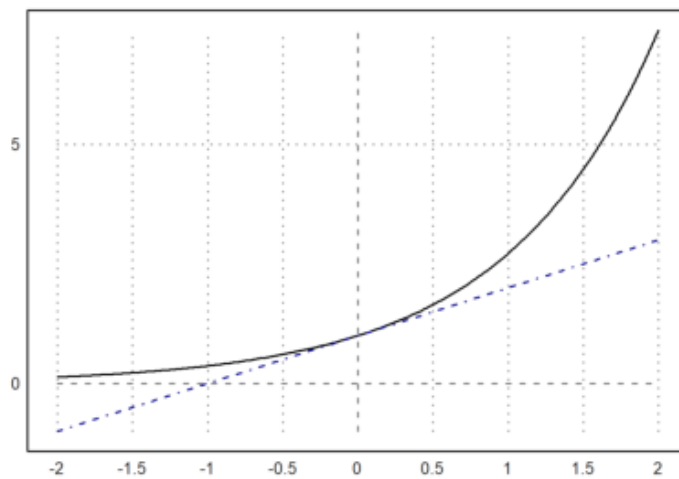


Ada juga kotak teks.

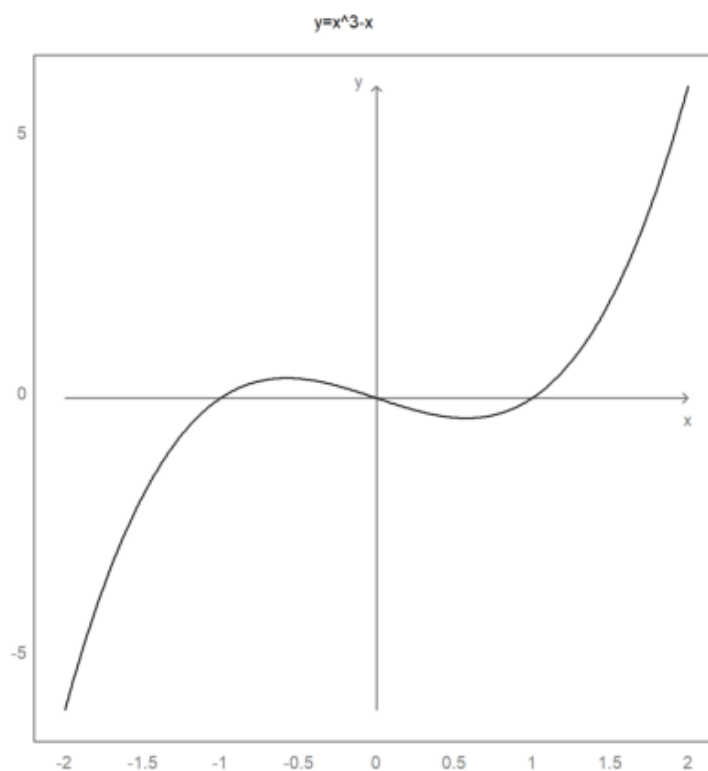
```
>plot2d(&f(x),-1,1,-2,2); // function
>plot2d(&diff(f(x),x),>add,style="--",color=red); // derivative
>labelbox(["f","f'"],["-","--"],[black,red]): // label box
```



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



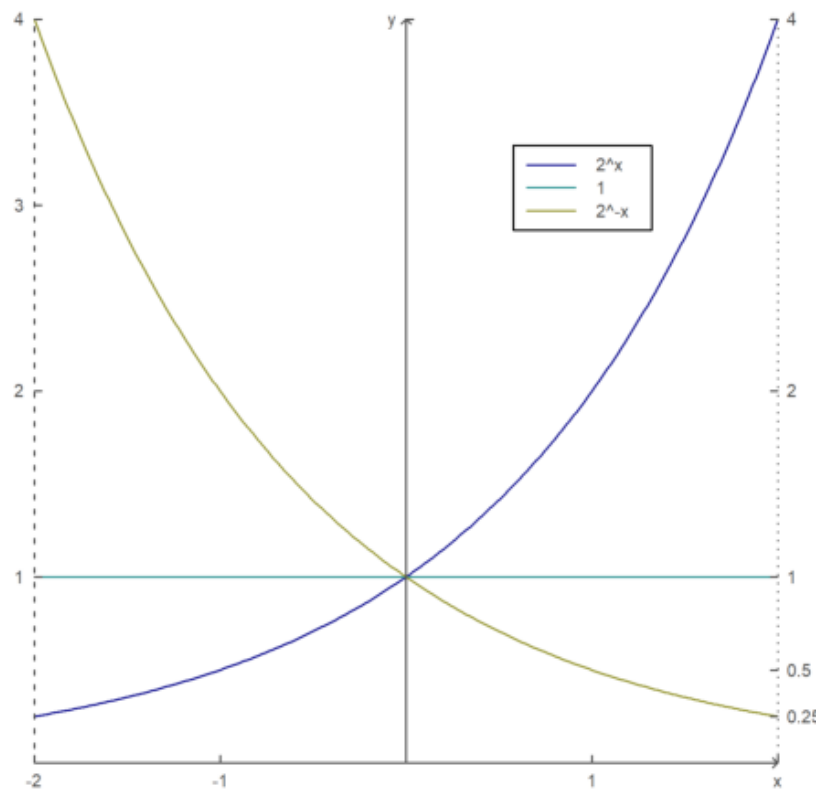
```
>gridstyle("-",color=gray,textcolor=gray,framecolor=gray); ...
plot2d("x^3-x",grid=1); ...
settitle("y=x^3-x",color=black); ...
label("x",2,0,pos="bc",color=gray); ...
label("y",0,6,pos="cl",color=gray); ...
reset():
```



Untuk kontrol yang lebih baik, sumbu x dan sumbu y dapat dilakukan secara manual.

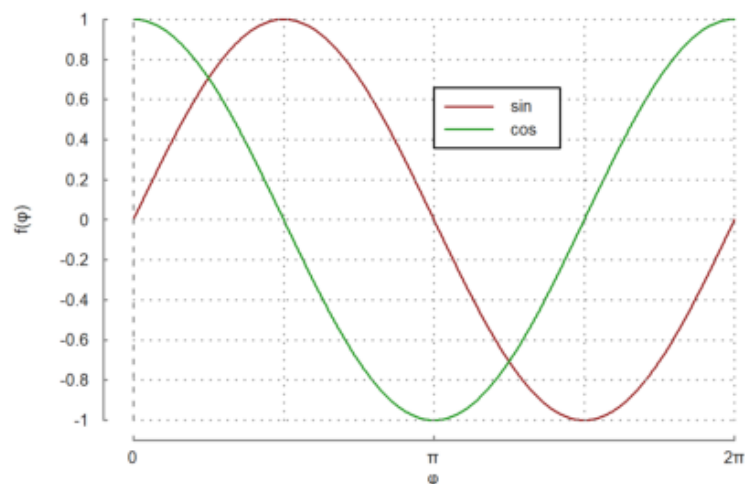
Perintah `fullwindow()` memperluas jendela plot karena kita tidak lagi membutuhkan tempat untuk label di luar jendela plot. Gunakan `shrinkwindow()` atau `reset()` untuk mengatur ulang ke nilai default.

```
>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 di luar area plot.

```
>aspect(1.5);
>plot2d(["sin(x)", "cos(x)"], 0, 2pi, color=[red, green], <grid, <frame); ...
  xaxis(-1.1, (0:2)*pi, xt=["0", u"π", u"2π"], 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); ...
  xlabel(u"φ"); ylabel(u"f(φ)");
```



Memplot Data 2D

Jika x dan y adalah vektor data, data ini akan digunakan sebagai koordinat x dan y dari suatu 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 diatur untuk mempertahankan rasio aspek persegi.

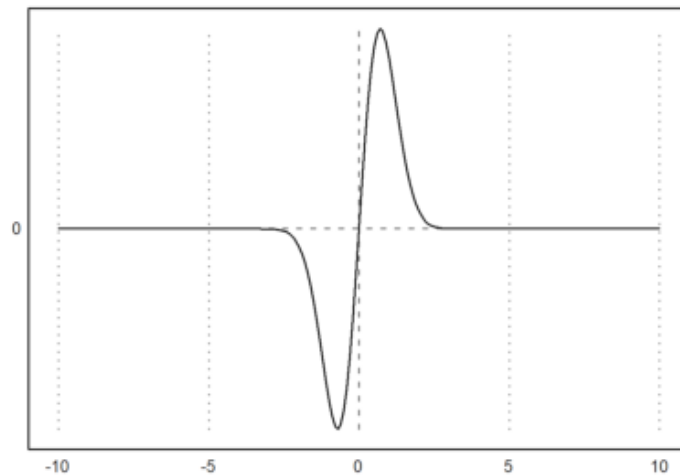
Memplot ekspresi hanyalah singkatan dari plot data. Untuk plot data, Anda memerlukan satu atau beberapa baris nilai x , dan satu atau beberapa baris nilai y . Dari rentang dan nilai x , fungsi `plot2d` akan menghitung data yang akan diplot, secara default dengan evaluasi adaptif fungsi. Untuk plot titik, gunakan `>points`, untuk garis dan titik campuran, gunakan `>addpoints`.

Tetapi Anda dapat memasukkan data secara langsung.

- Gunakan vektor baris untuk x dan y untuk satu fungsi.
- Matriks untuk x dan y 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):
```



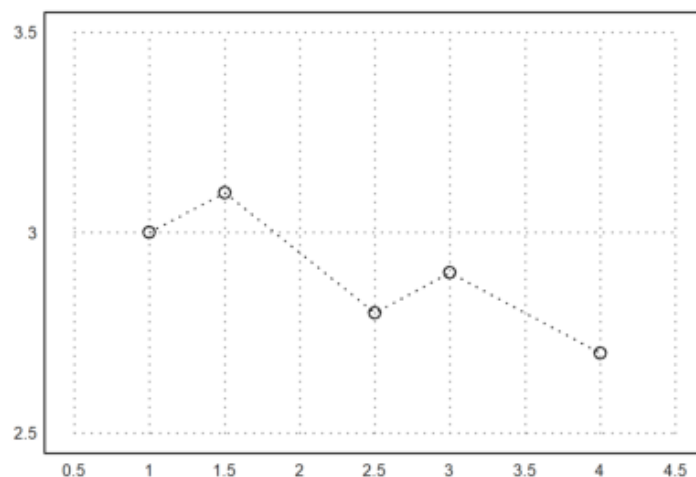
Data juga dapat diplot sebagai titik. Gunakan points=true untuk ini. Plot ini bekerja seperti poligon, tetapi hanya menggambar sudutnya.

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

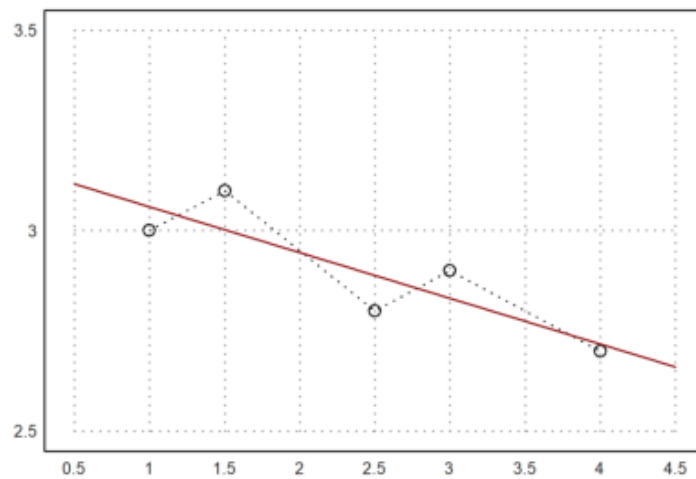
Untuk memplot kumpulan titik, gunakan >points. Jika warnanya adalah vektor warna, setiap titik akan mendapatkan warna yang berbeda. Untuk matriks koordinat dan vektor kolom, warna tersebut berlaku untuk baris-baris matriks.

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
```



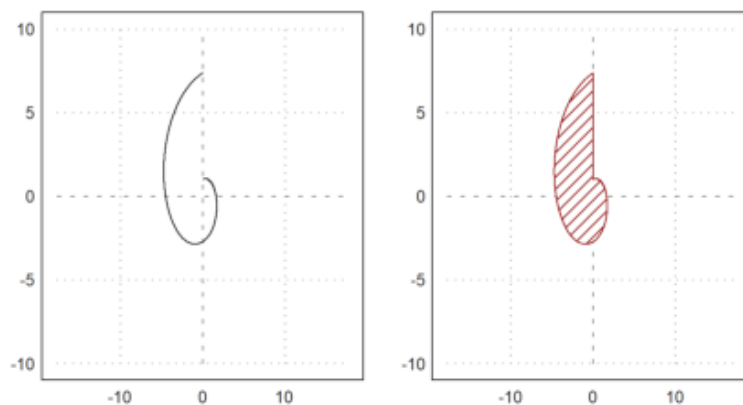
Menggambar Daerah yang Dibatasi Kurva

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

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

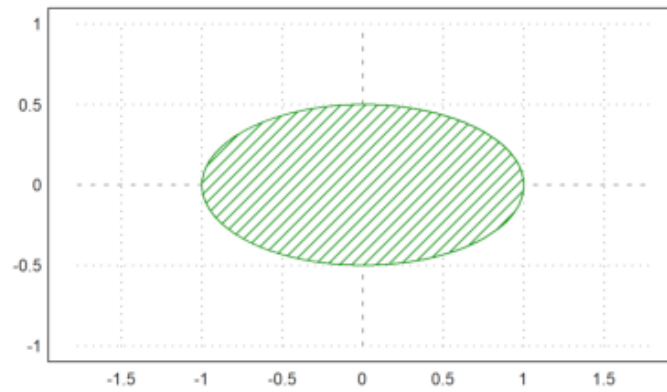
Warna isian ditentukan oleh argumen "`fillcolor`", dan pada `<outline` opsional mencegah penggambaran batas untuk semua gaya kecuali yang default.

```
>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):
```

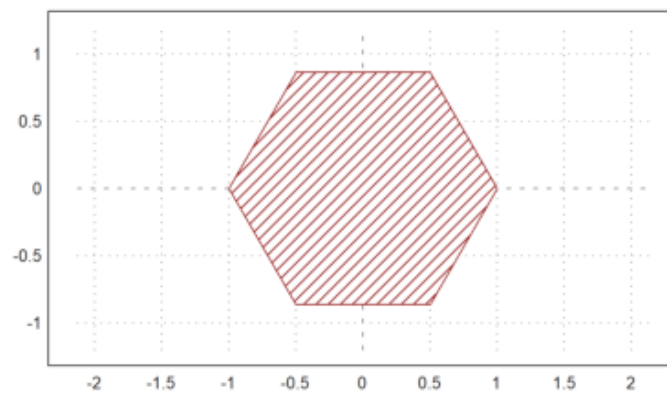


Dalam contoh berikut ini kami memplot elips yang terisi dan dua heksagon yang terisi menggunakan kurva tertutup dengan 6 titik dengan gaya isian yang berbeda.

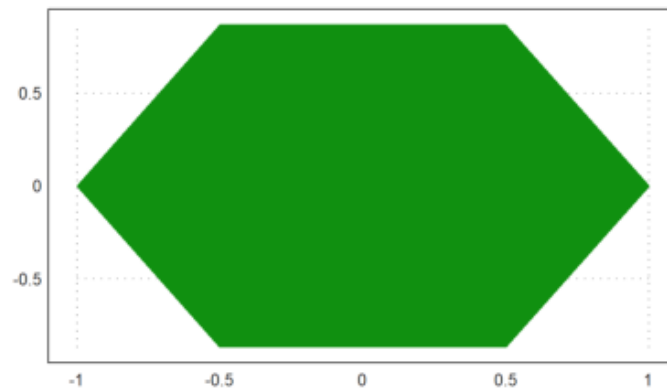
```
>x=linspace(0,2pi,1000); plot2d(sin(x),cos(x)*0.5,r=1,>filled,style="/"):
```



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

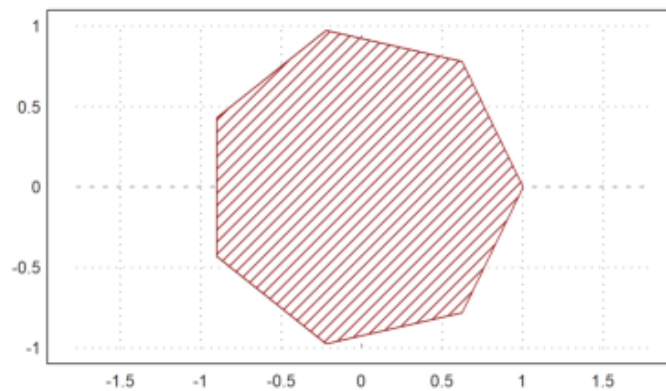


```
>t=linspace(0,2pi,6); plot2d(cos(t),sin(t),>filled,style="#"): :
```



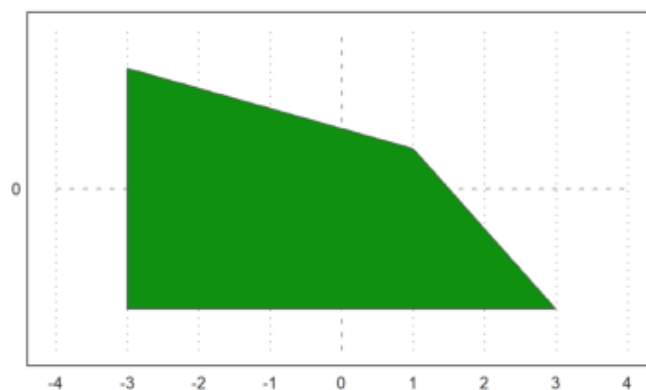
Contoh lain adalah septagon, yang kita buat dengan 7 titik pada lingkaran satuan.

```
>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 $A[k].v \leq 3$ 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):
```



Poin utama dari bahasa matriks adalah memungkinkan pembuatan tabel fungsi dengan mudah.

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

Sekarang kita memiliki vektor nilai x dan y. plot2d() dapat memplot nilai-nilai ini sebagai kurva yang menghubungkan titik-titik. Plot tersebut dapat diisi. Dalam hal ini, ini menghasilkan hasil yang bagus berkat aturan lilitan, yang digunakan untuk pengisian.

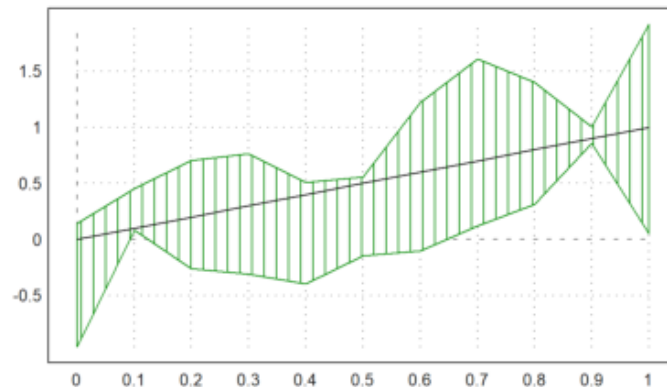
```
>plot2d(x,y,<grid,<frame,>filled):
```



Vektor interval diplot terhadap nilai x sebagai daerah terisi antara nilai interval bawah dan atas.

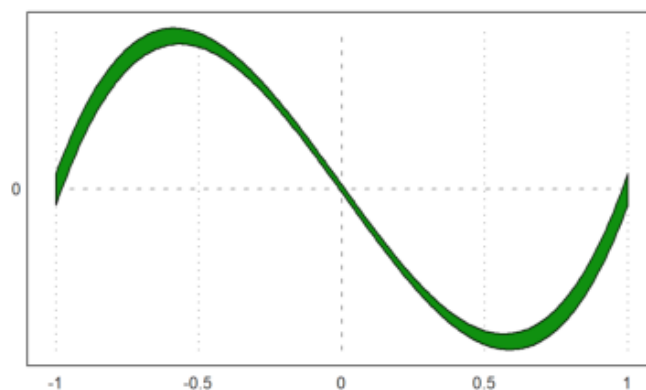
Hal ini dapat berguna untuk memplot kesalahan komputasi. Namun, hal ini juga dapat digunakan untuk memplot kesalahan statistik.

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



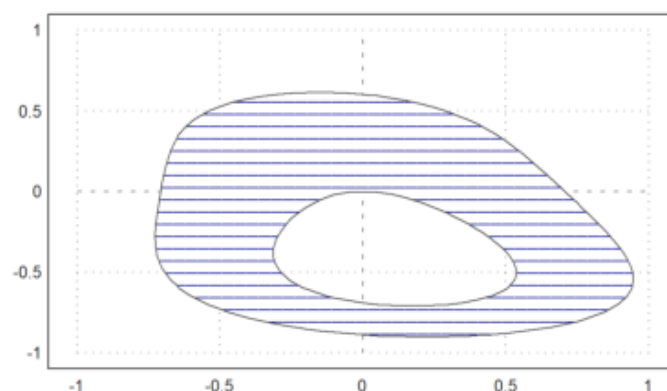
Jika x adalah vektor yang diurutkan, dan y adalah vektor interval, maka plot2d akan memplot rentang interval yang terisi pada bidang. Gaya isianya sama dengan gaya poligon.

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



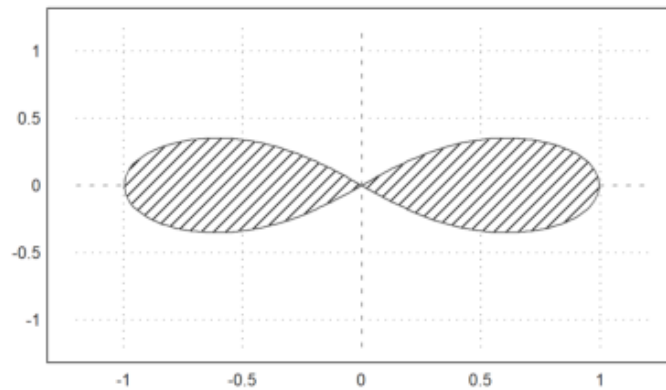
Dimungkinkan untuk mengisi wilayah nilai untuk fungsi tertentu. Untuk ini, level harus berupa matriks 2xn. Baris pertama adalah batas bawah, dan baris kedua berisi batas atas.

```
>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
```

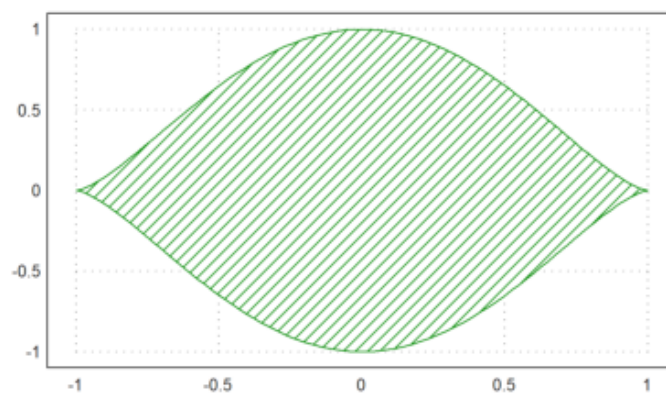


Kita juga dapat mengisi rentang nilai seperti

```
>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="/"):
```



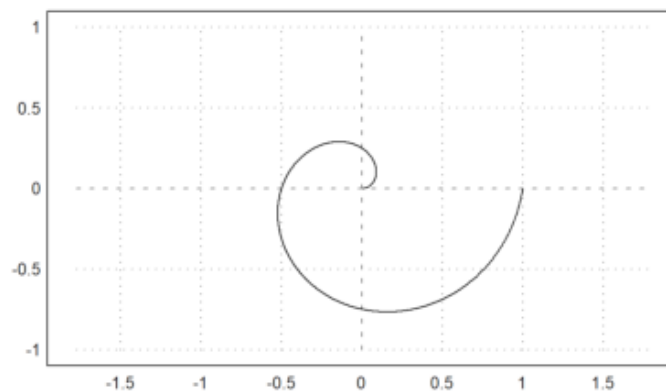
Grafik Fungsi Parametrik

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

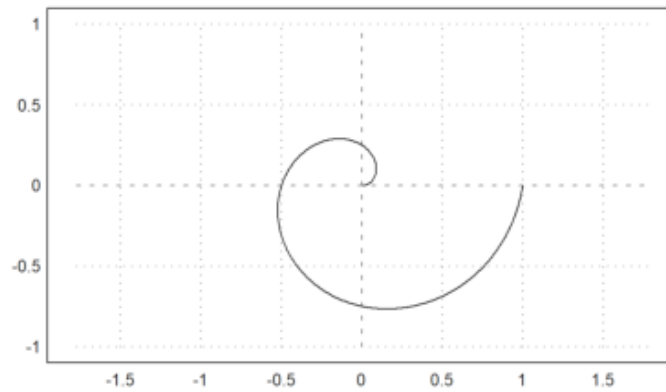
Kita perlu menggunakan banyak titik untuk tampilan yang halus atau fungsi adaptif() untuk mengevaluasi ekspresi (lihat fungsi adaptif() untuk detail lebih lanjut).

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

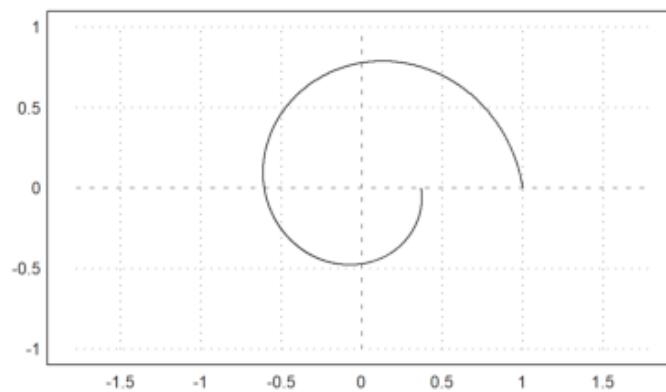


Sebagai alternatif, dimungkinkan untuk menggunakan dua ekspresi untuk kurva. Berikut ini memplot kurva yang sama seperti di atas.

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

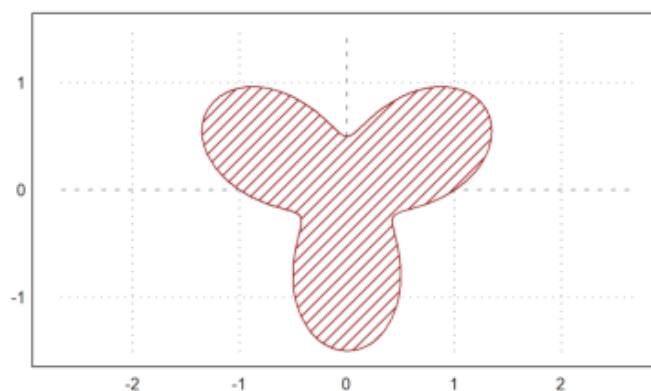


```
>t=linspace(0,1,1000); r=exp(-t); x=r*cos(2*pi*t); y=r*sin(2*pi*t);
>plot2d(x,y,r=1):
```



Pada contoh berikutnya, kita plot kurva
dengan

```
>t=linspace(0,2*pi,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):
```



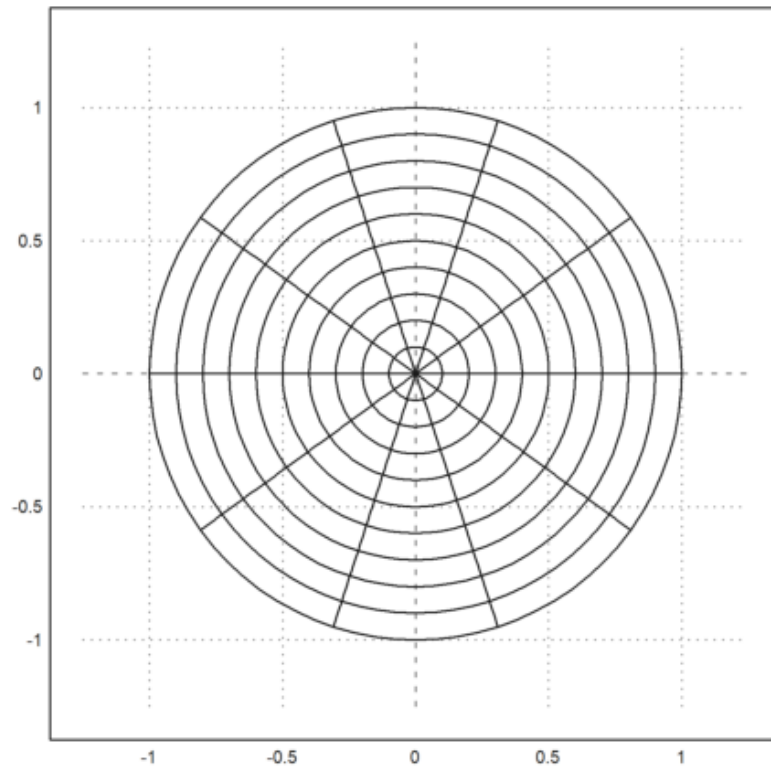
Menggambar Grafik Bilangan Kompleks

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.

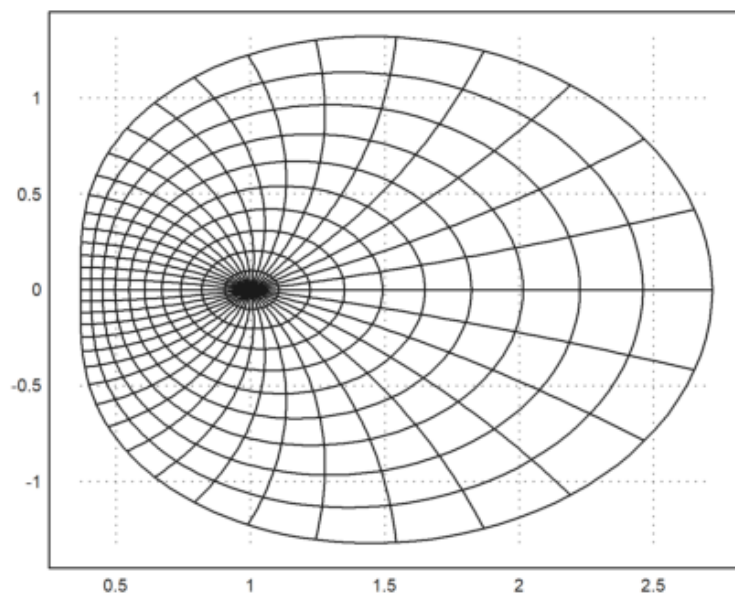
Matriks bilangan kompleks akan secara otomatis diplot sebagai grid pada bidang kompleks.

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

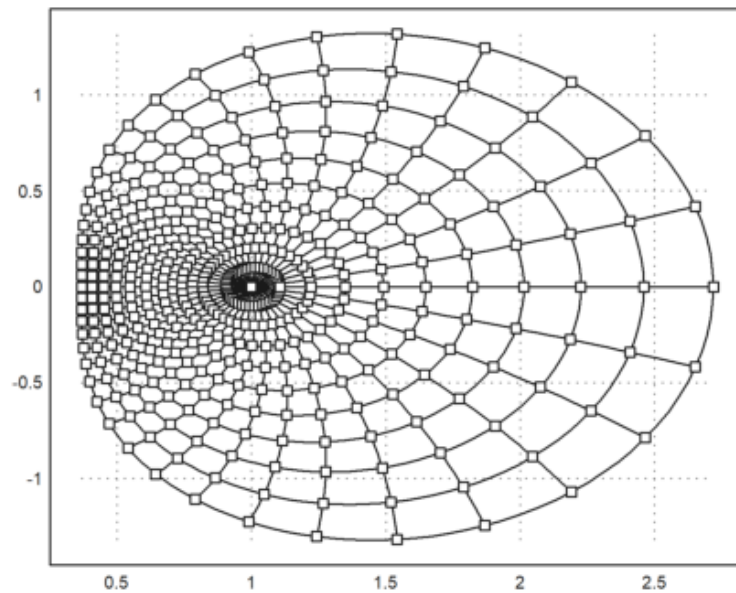
```
>aspect(); r=linspace(0,1,50); a=linspace(0,2pi,80)'; z=r*exp(I*a);...
plot2d(z,a=-1.25,b=1.25,c=-1.25,d=1.25,cgrid=10):
```



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



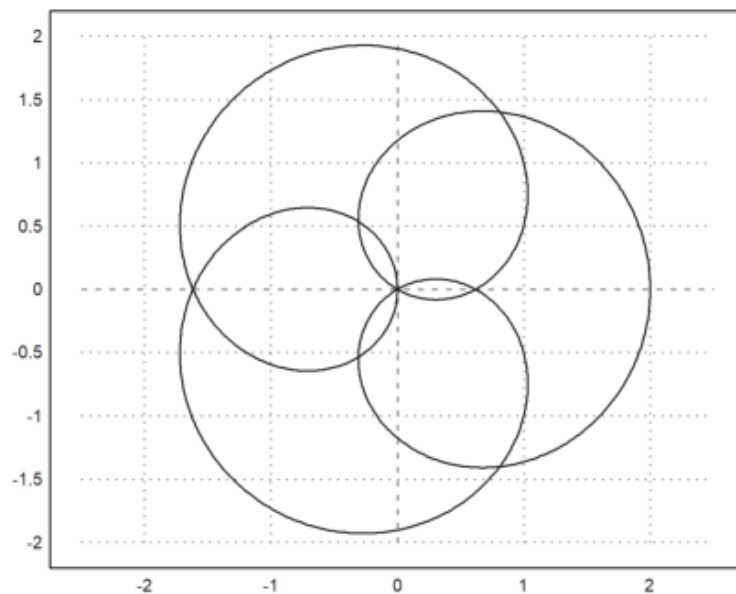
```
>r=linspace(0,1,10); a=linspace(0,2pi,40)'; z=r*exp(I*a);
>plot2d(exp(z),>points,>add):
```



Vektor bilangan kompleks secara otomatis diplot sebagai kurva pada bidang kompleks dengan bagian riil dan bagian imajiner.

Dalam contoh ini, kita memplot lingkaran satuan dengan

```
>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.

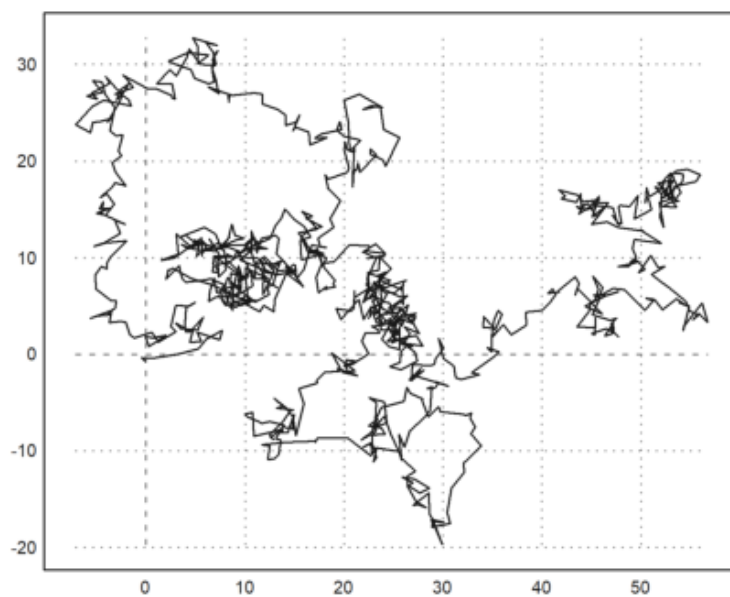
Penjumlahan kumulatif dari nilai-nilai berdistribusi normal 0-1 menghasilkan pergerakan acak (random walk).

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

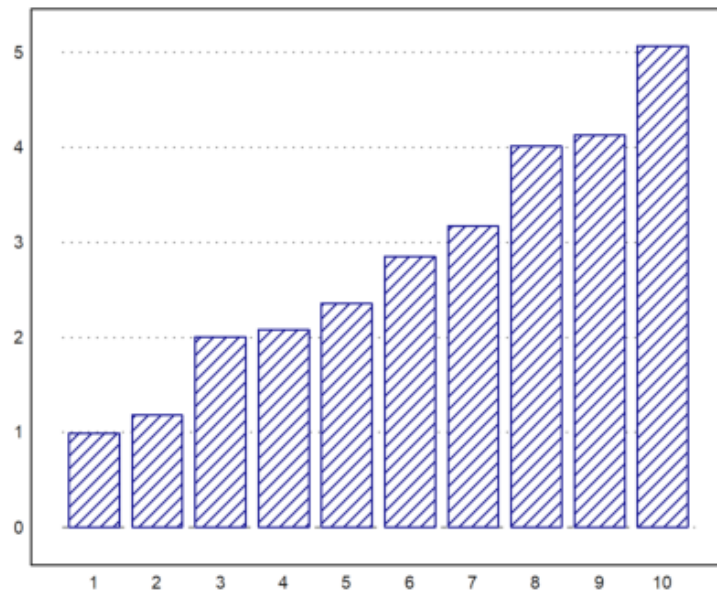


Menggunakan dua baris menunjukkan jalan dalam dua dimensi.

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

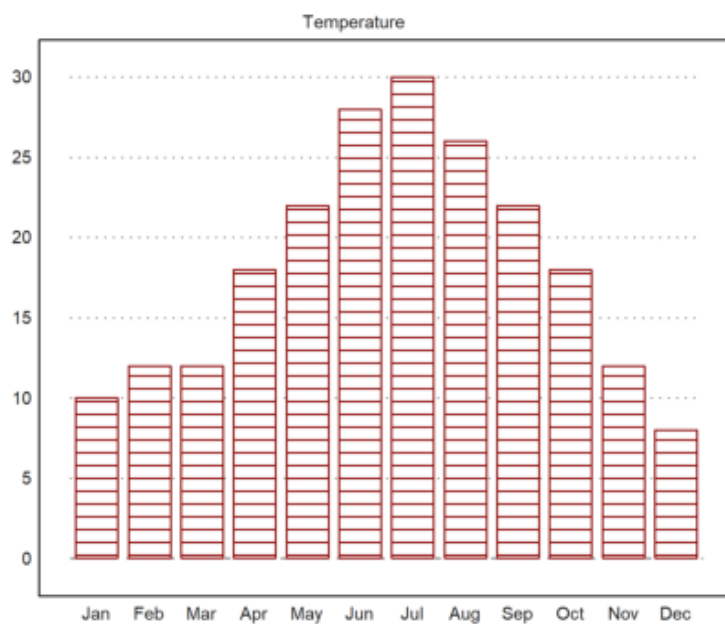


```
>columnspc(cumsum(random(10)),style="/",color=blue):
```

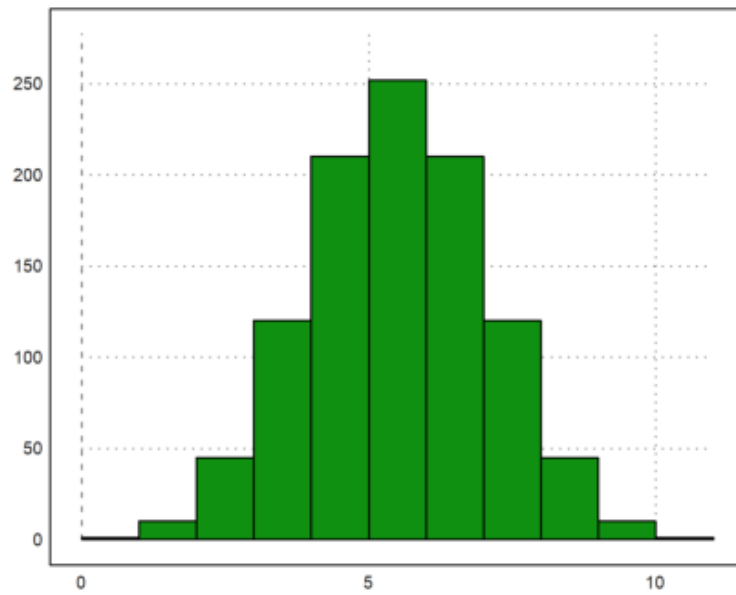


la juga dapat menampilkan string sebagai label.

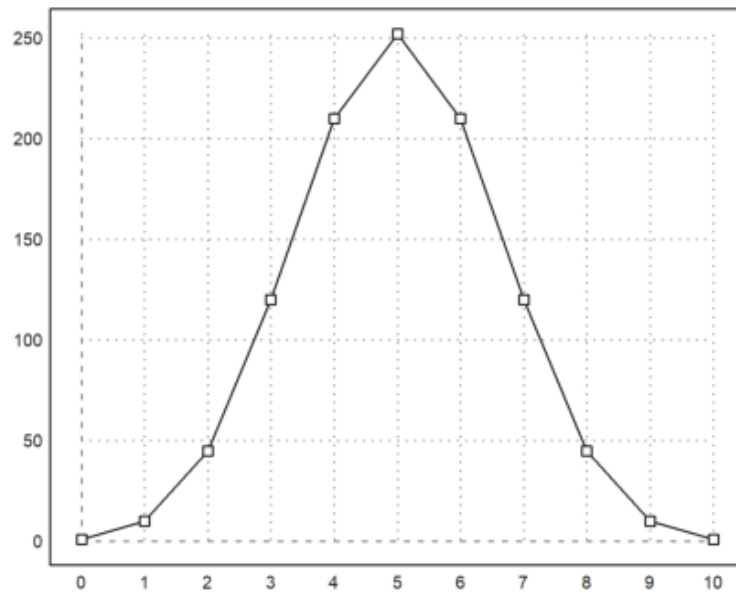
```
>months=["Jan", "Feb", "Mar", "Apr", "May", "Jun", ...
         "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"];
>values=[10,12,12,18,22,28,30,26,22,18,12,8];
>columnsplot(values,lab=months,color=red,style="-");
>title("Temperature"):
```



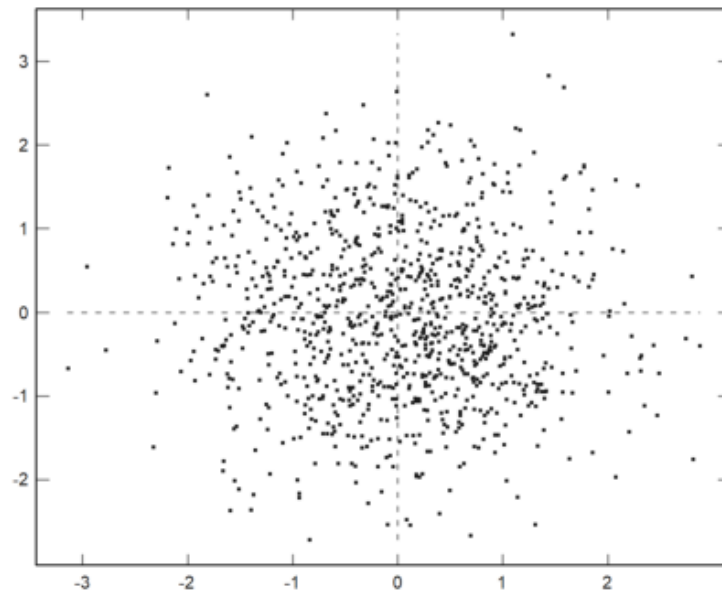
```
>k=0:10;
>plot2d(k,bin(10,k),>bar):
```



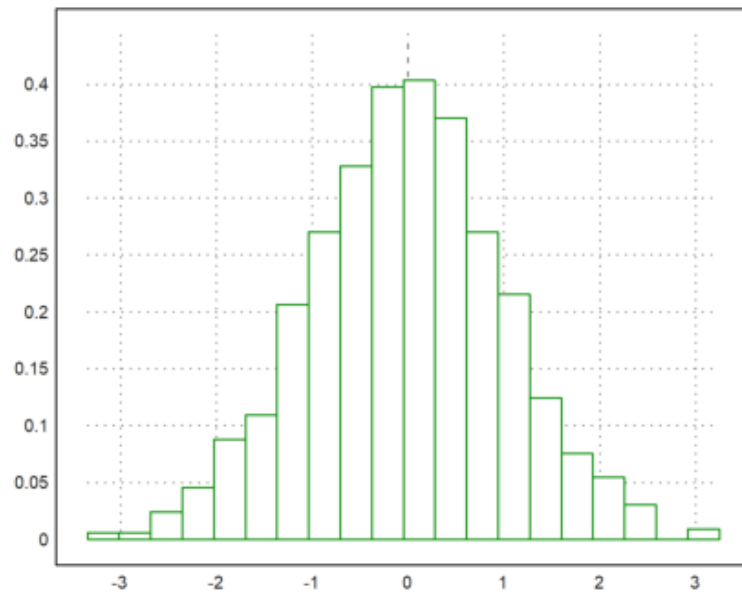
```
>plot2d(k,bin(10,k)); plot2d(k,bin(10,k),>points,>add):
```



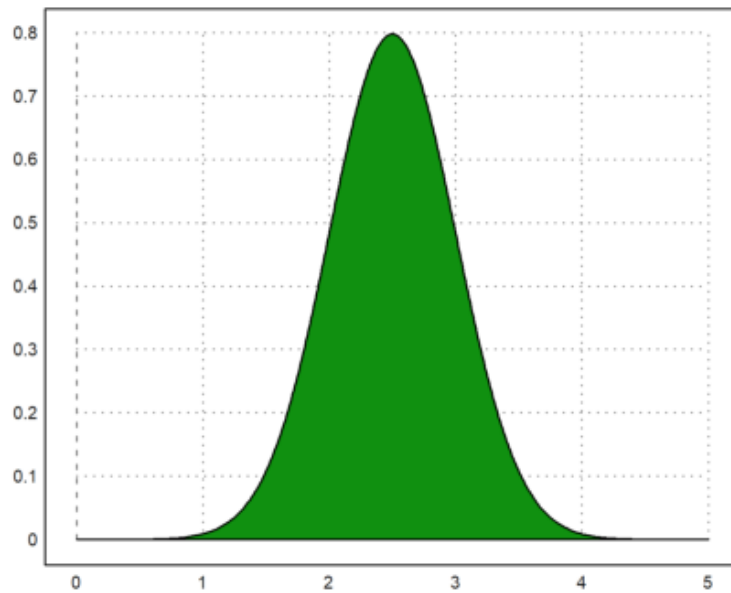
```
>plot2d(normal(1000),normal(1000),>points,grid=6,style=".."):
```



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

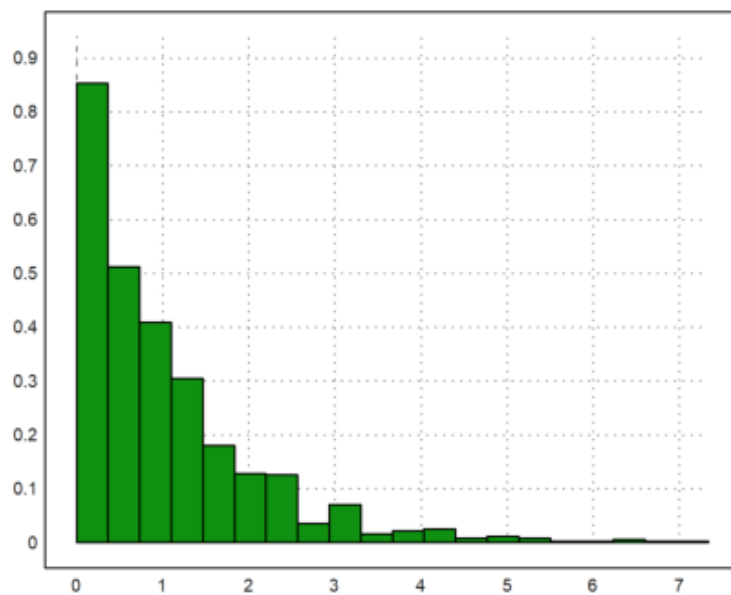


```
>plot2d("qnormal",0,5;2.5,0.5,>filled):
```



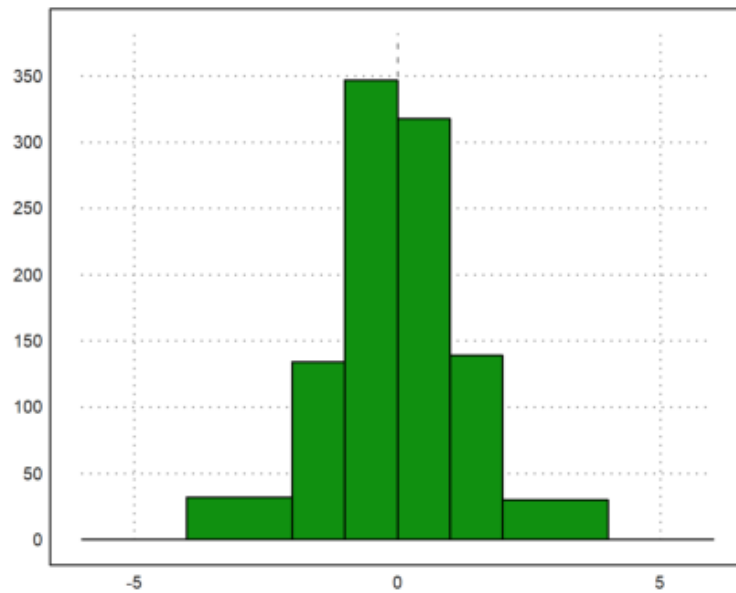
Untuk memplot distribusi statistik eksperimental, Anda dapat menggunakan `distribution=n` dengan `plot2d`.

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



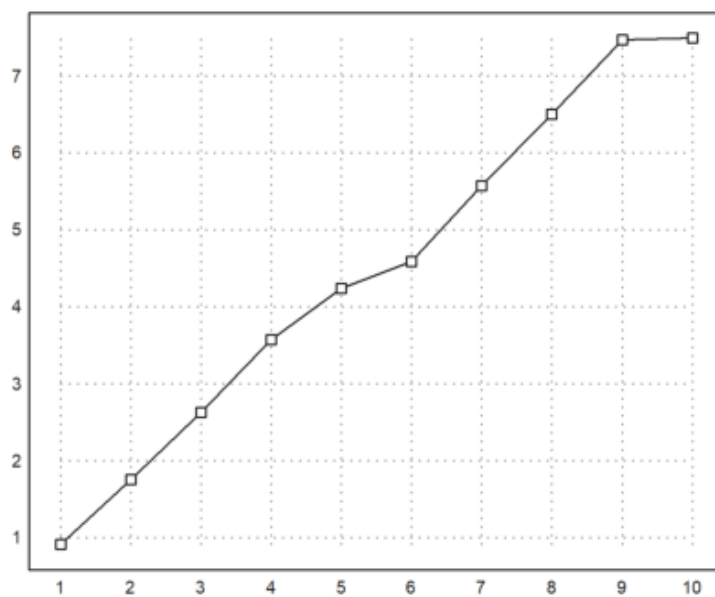
Atau Anda dapat menghitung distribusi dari data dan memplot hasilnya dengan `>bar` di `plot3d`, atau dengan plot kolom.

```
>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):
```

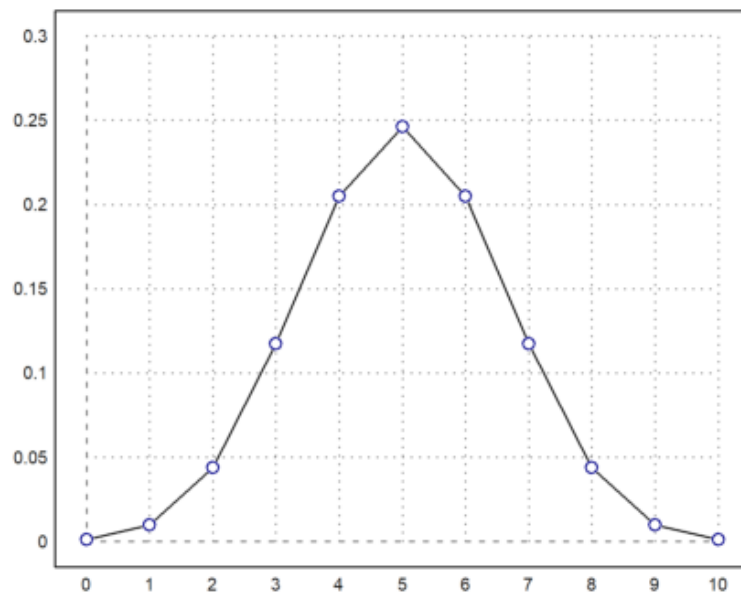


Fungsi `statplot()` mengatur gaya dengan string sederhana.

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



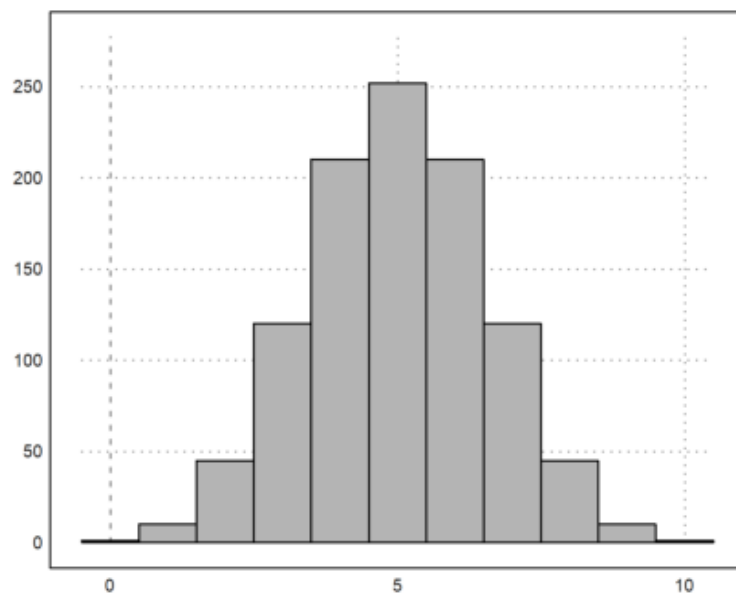
```
>n=10; i=0:n; ...
  plot2d(i,bin(n,i)/2^n,a=0,b=10,c=0,d=0.3); ...
  plot2d(i,bin(n,i)/2^n,points=true,style="ow",add=true,color=blue):
```



Selain itu, data dapat diplot sebagai batang. Dalam hal ini, x harus diurutkan dan satu elemen lebih panjang dari y. Batang akan memanjang dari $x[i]$ hingga $x[i+1]$ dengan nilai $y[i]$. Jika x berukuran sama dengan y, x akan memanjang satu elemen dengan spasi terakhir.

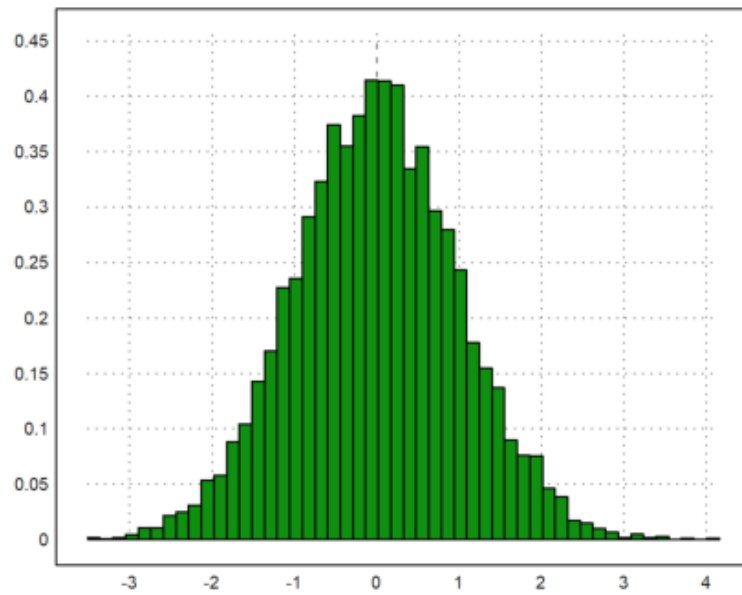
Gaya isian dapat digunakan seperti di atas.

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

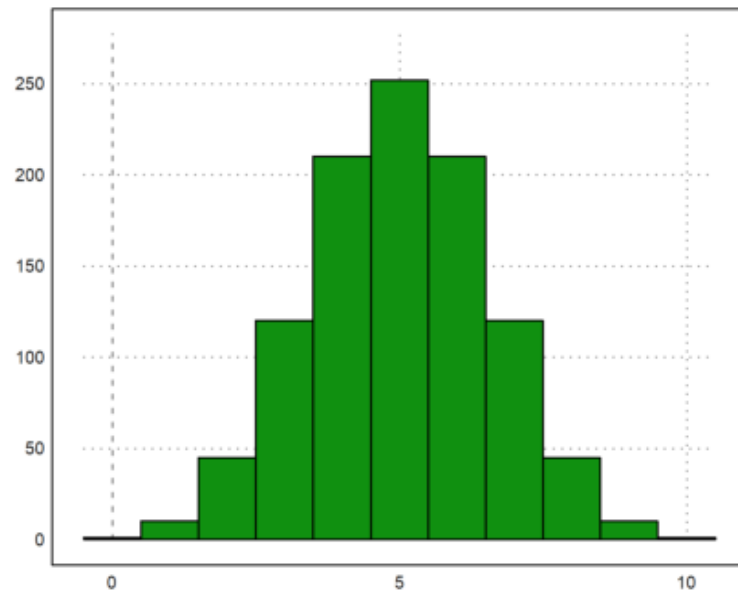


Data untuk diagram batang (batang=1) dan histogram (histogram=1) dapat diberikan secara eksplisit dalam xv dan yv, atau dapat dihitung dari distribusi empiris dalam xv dengan distribusi > (atau distribusi=n). Histogram nilai xv akan dihitung secara otomatis dengan histogram >. Jika >genap ditentukan,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

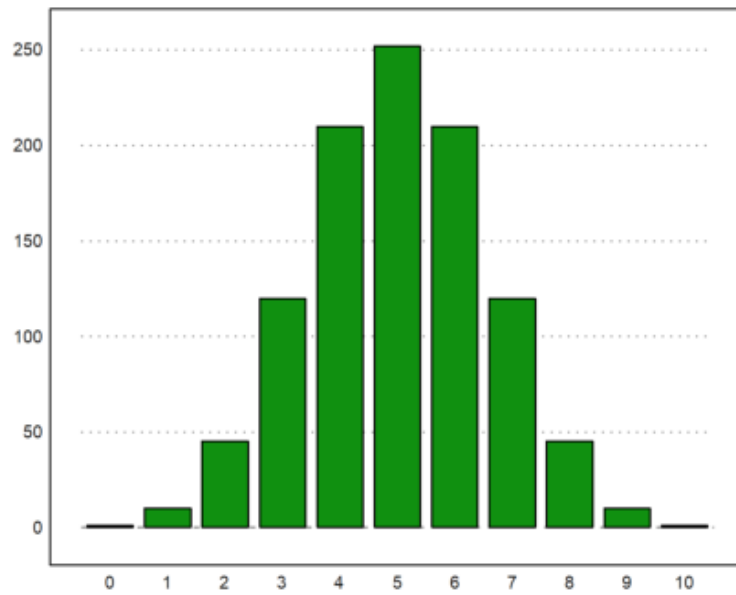
```
>plot2d(normal(10000),distribution=50):
```



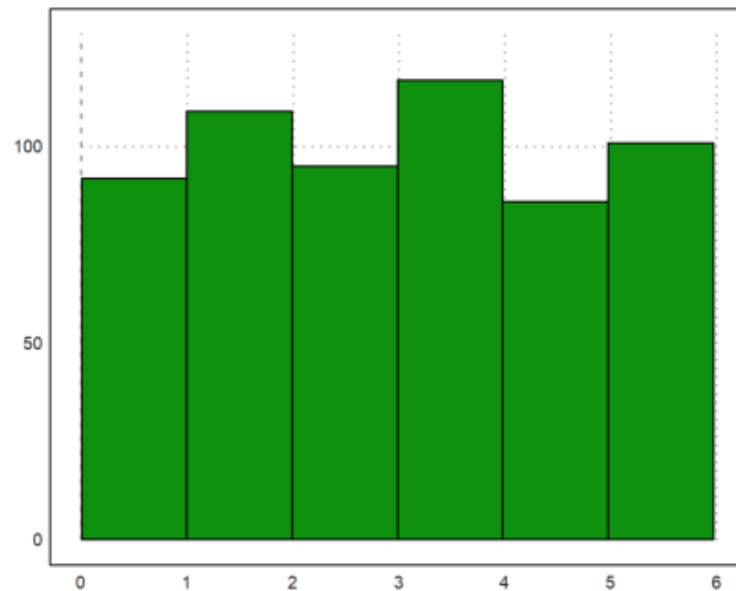
```
>k=0:10; m=bin(10,k); x=(0:11)-0.5; plot2d(x,m,>bar):
```



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

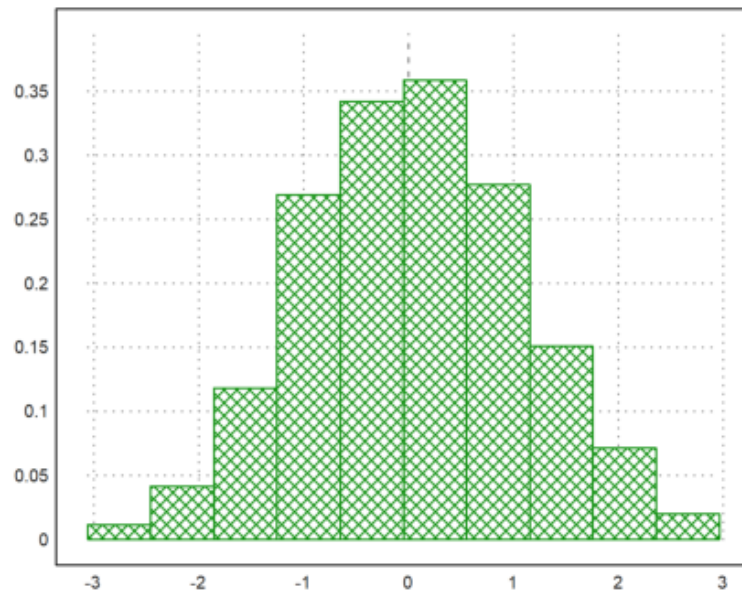


```
>plot2d(random(600)*6,histogram=6):
```



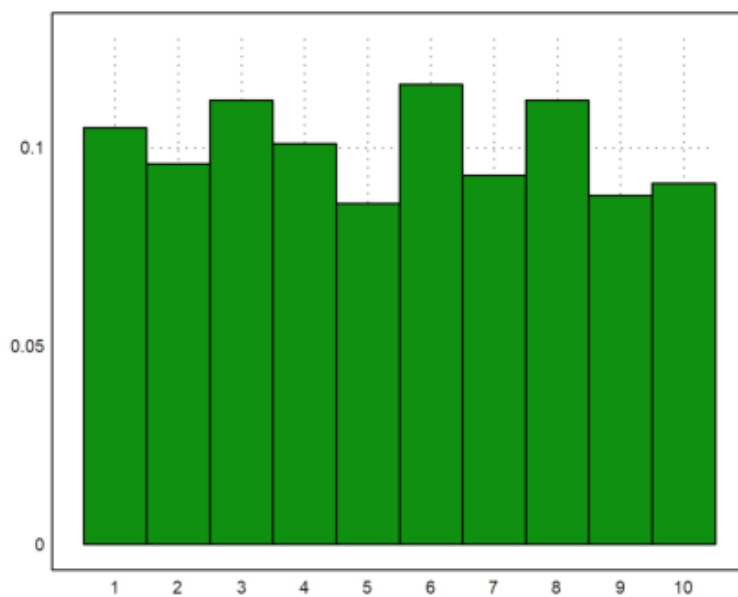
Untuk distribusi, ada parameter `distribution=n`, yang menghitung nilai secara otomatis dan mencetak distribusi relatif dengan `n` sub-interval.

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



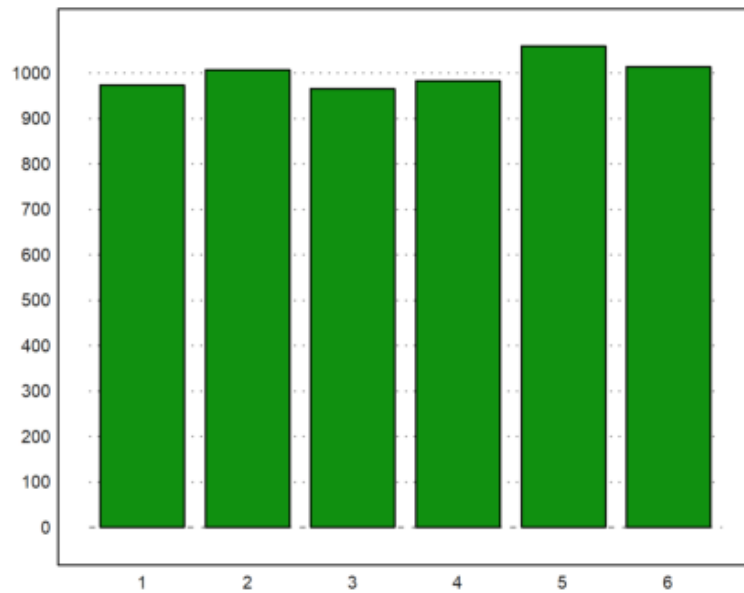
Dengan parameter `even=true`, ini akan menggunakan interval integer.

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

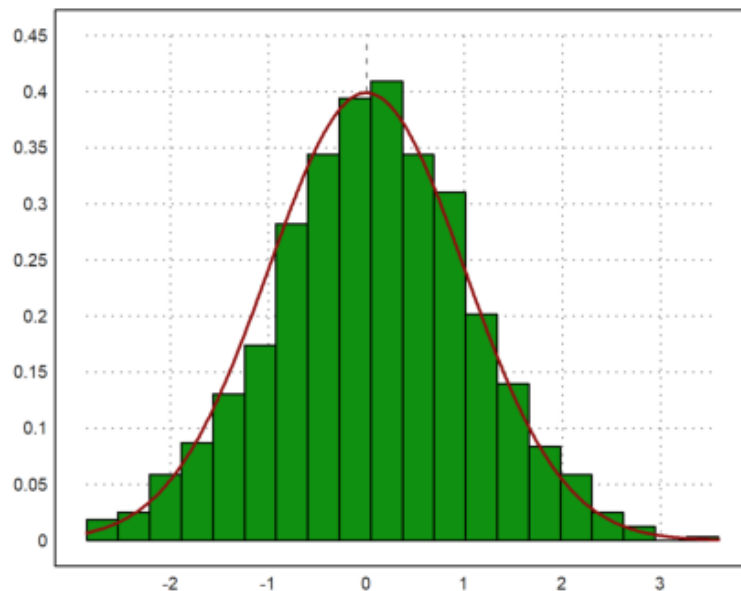


Perlu diketahui bahwa ada banyak plot statistik yang mungkin berguna. Lihat tutorial tentang statistik.

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

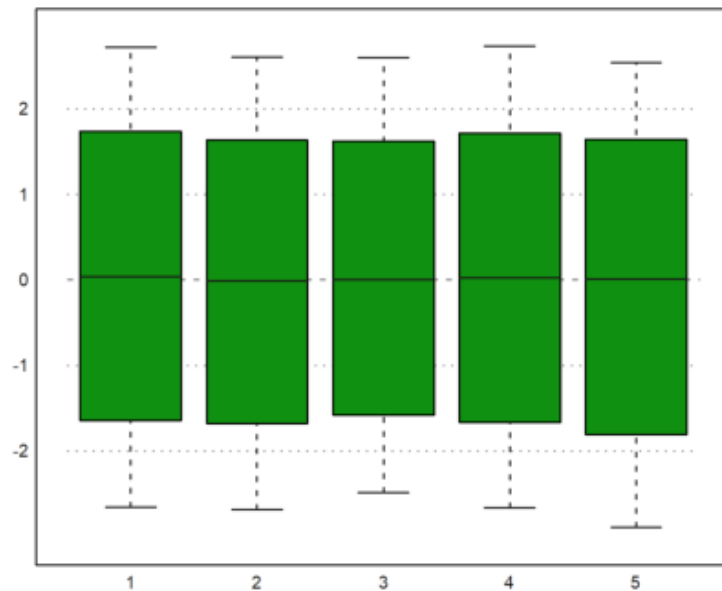


```
>plot2d(normal(1,1000),>distribution); ...
  plot2d("qnormal(x)",color=red,thickness=2,>add):
```



Terdapat juga banyak plot khusus untuk statistik. Boxplot menunjukkan kuartil dari distribusi ini dan banyaknya outlier. Menurut definisi, outlier dalam boxplot adalah data yang melebihi 1,5 kali rentang 50% tengah plot.

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



Fungsi Implisit

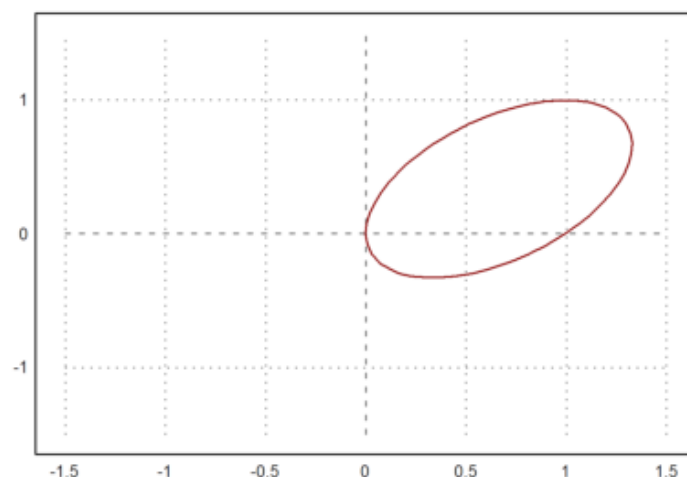
Plot implisit menunjukkan garis level yang menyelesaikan $f(x,y)=\text{level}$, di mana "level" dapat berupa nilai tunggal atau vektor nilai. Jika $\text{level}=\text{"auto"}$, akan ada garis level n_c , yang akan menyebar secara merata antara nilai minimum dan maksimum fungsi. Warna yang lebih gelap atau lebih terang dapat ditambahkan dengan >hue untuk menunjukkan nilai fungsi. Untuk fungsi implisit, xv harus berupa fungsi atau ekspresi parameter x dan y , atau, sebagai alternatif, xv dapat berupa matriks nilai.

Euler dapat menandai garis level

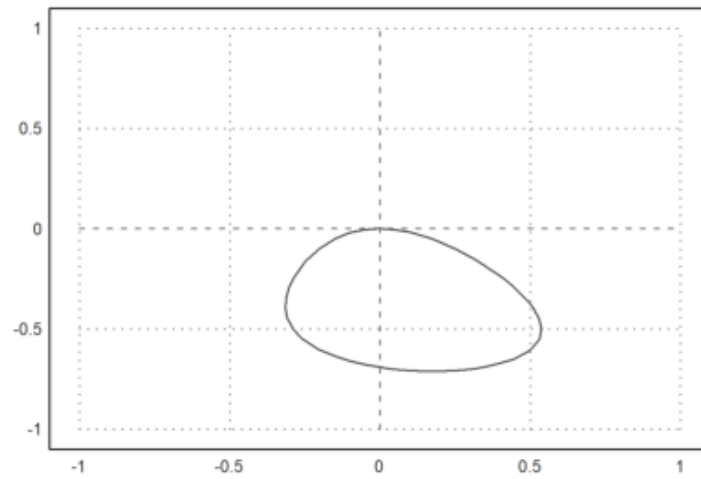
dari fungsi apa pun.

Untuk menggambar himpunan $f(x,y)=c$ untuk satu atau lebih konstanta c , Anda dapat menggunakan `plot2d()` dengan plot implisitnya di bidang. Parameter untuk c adalah $\text{level}=c$, di mana c dapat berupa vektor garis level. Selain itu, skema warna dapat digambar di latar belakang untuk menunjukkan nilai fungsi untuk setiap titik dalam plot. Parameter "n" menentukan kehalusan plot.

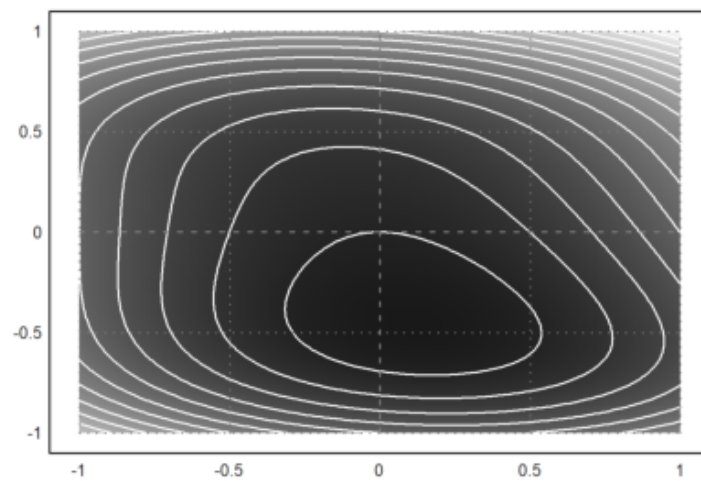
```
>aspect(1.5);
>plot2d("x^2+y^2-x*y-x", r=1.5, level=0, contourcolor=red):
```



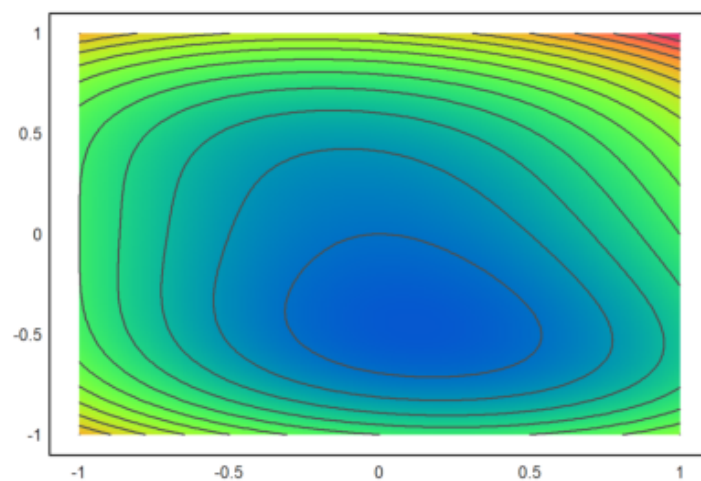
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr, level=0): // Solutions of f(x,y)=0
```



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

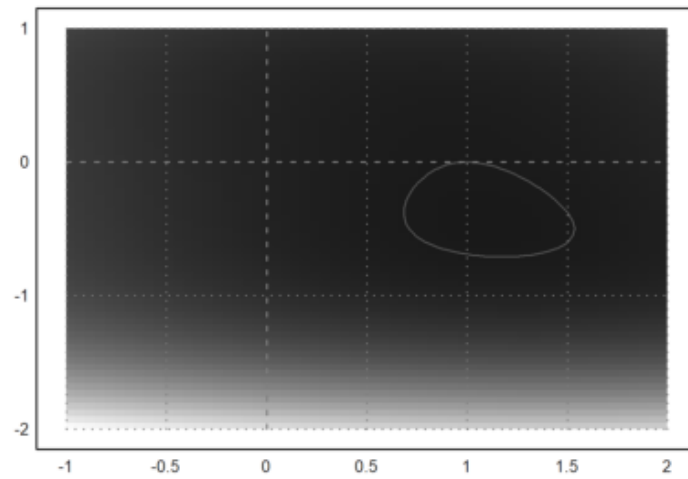


```
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=200,grid=4): // nicer
```

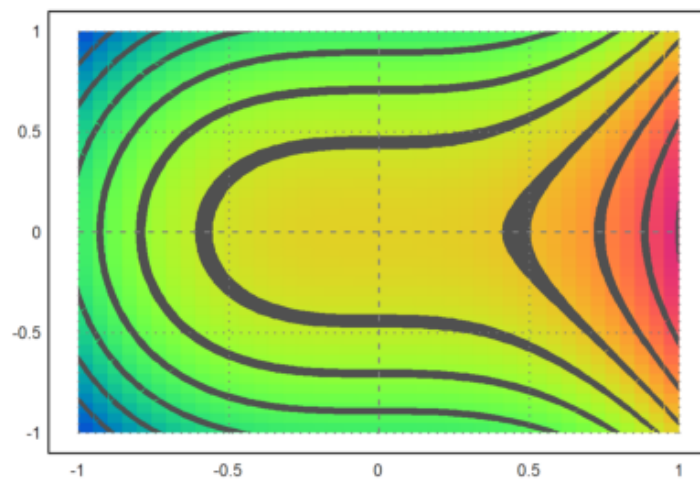


Ini juga berlaku untuk plot data. Namun, Anda harus menentukan rentang untuk label sumbu.

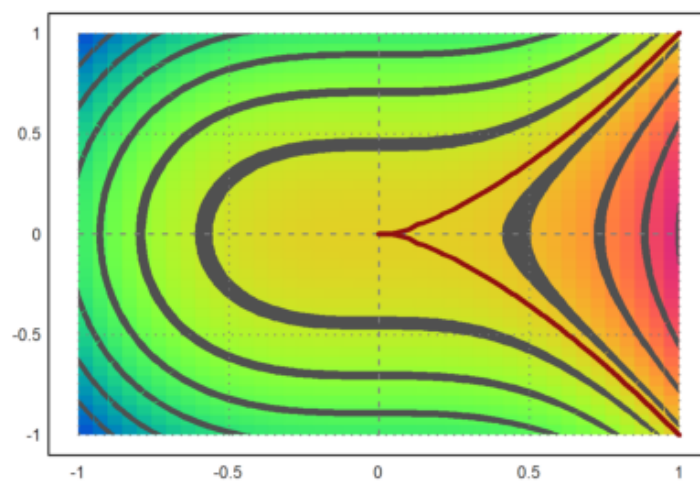
```
>x=-2:0.05:1; y=x'; z=expr(x,y);
>plot2d(z,level=0,a=-1,b=2,c=-2,d=1,>hue):
```



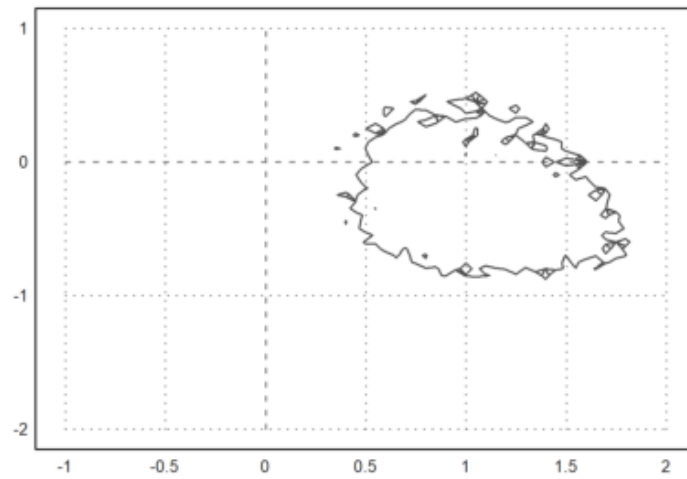
```
>plot2d("x^3-y^2",>contour,>hue,>spectral):
```



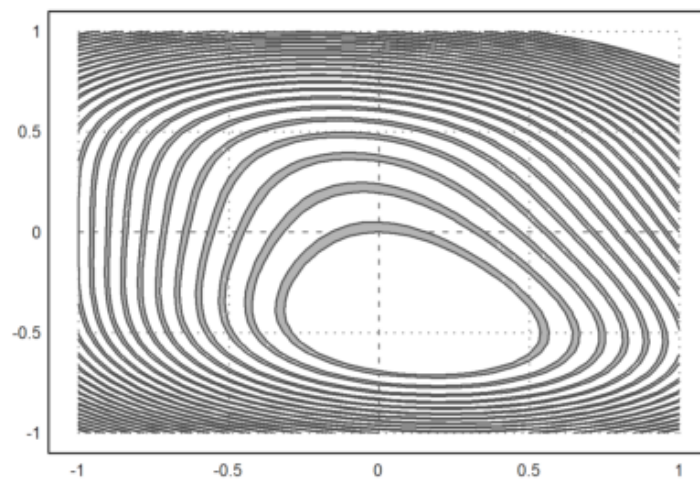
```
>plot2d("x^3-y^2",level=0,contourwidth=3,>add,contourcolor=red):
```



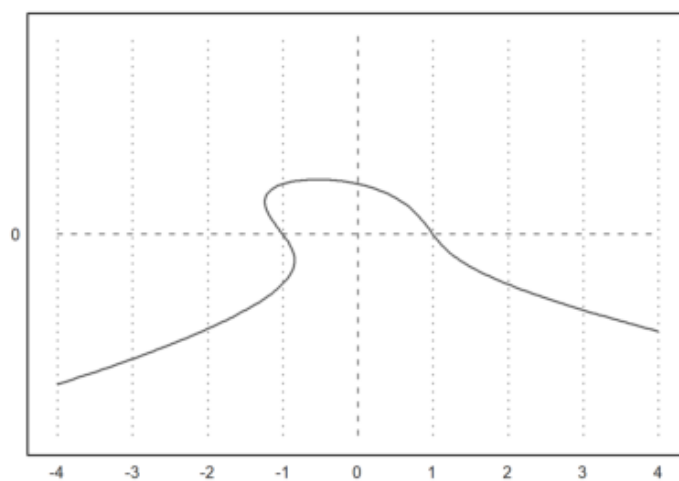
```
>z=z+normal(size(z))*0.2;
>plot2d(z,level=0.5,a=-1,b=2,c=-2,d=1):
```



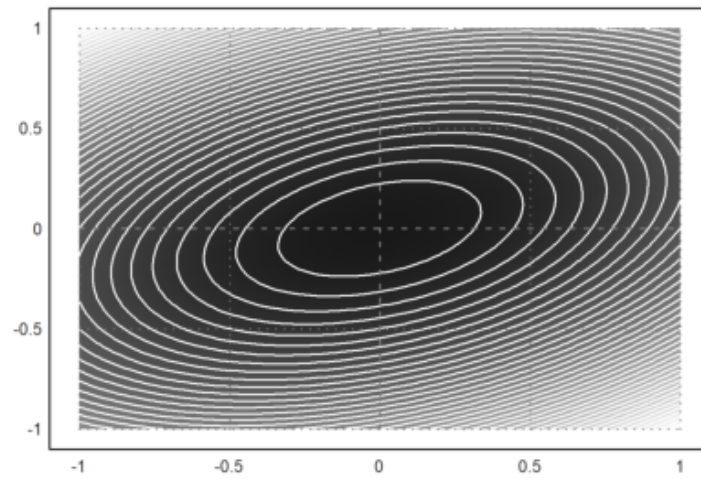
```
>plot2d(expr,level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
```



```
>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):
```

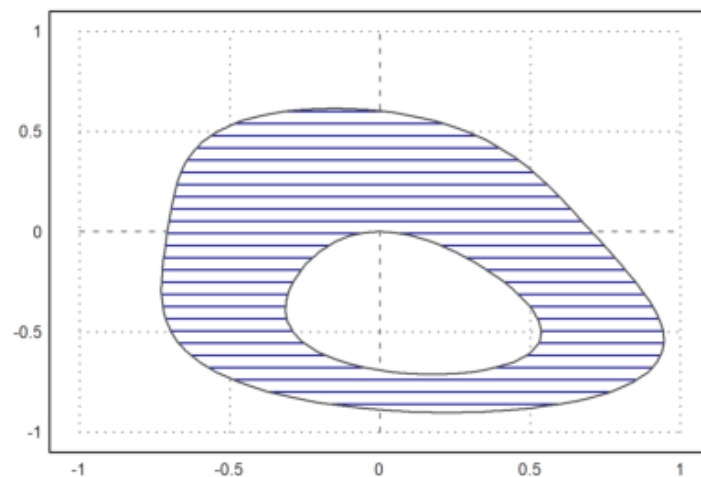


Dimungkinkan juga untuk mengisi himpunan

dengan rentang level.

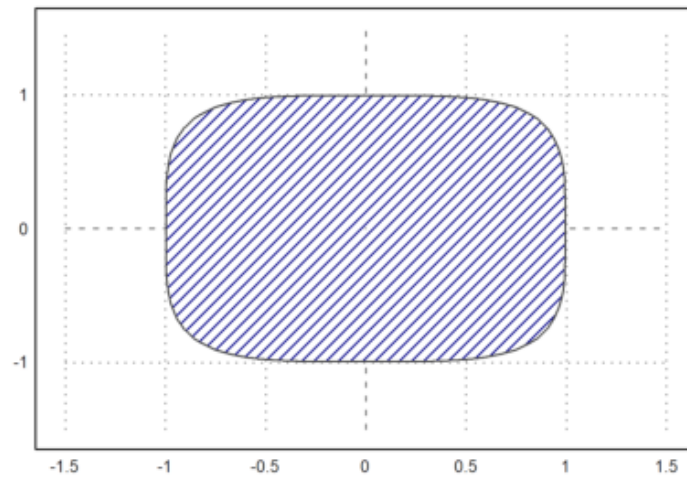
Dimungkinkan untuk mengisi wilayah nilai untuk fungsi tertentu. Untuk ini, level harus berupa matriks 2xn. Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

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

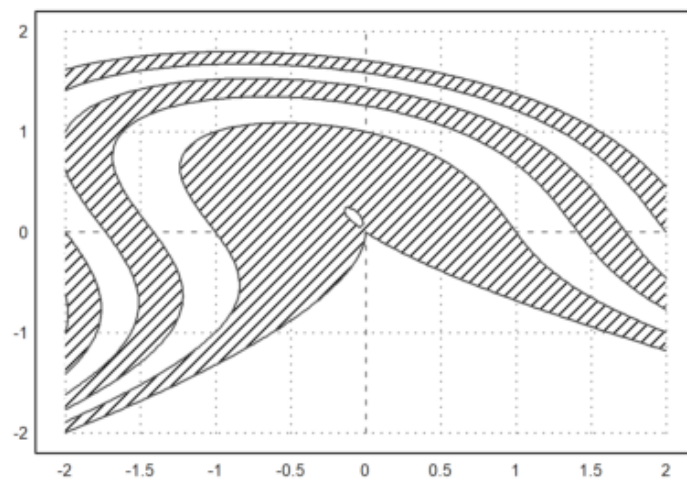


Plot implisit juga dapat menunjukkan rentang level. Level harus berupa matriks 2xn interval level, dengan baris pertama berisi awal dan baris kedua berisi akhir setiap interval. Sebagai alternatif, vektor baris sederhana dapat digunakan untuk level, dan parameter dl memperluas nilai level ke interval.

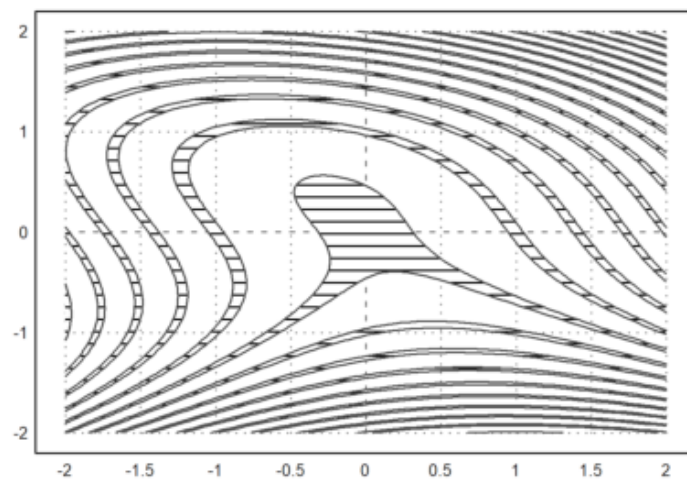
```
>plot2d("x^4+y^4", r=1.5, level=[0;1], color=blue, style="/"): 
```



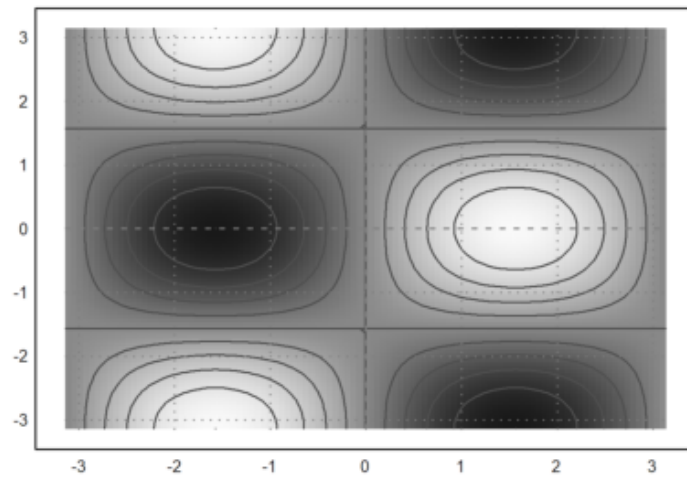
```
>plot2d("x^2+y^3+x*y", level=[0,2,4;1,3,5], style="/", r=2, n=100) :
```



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



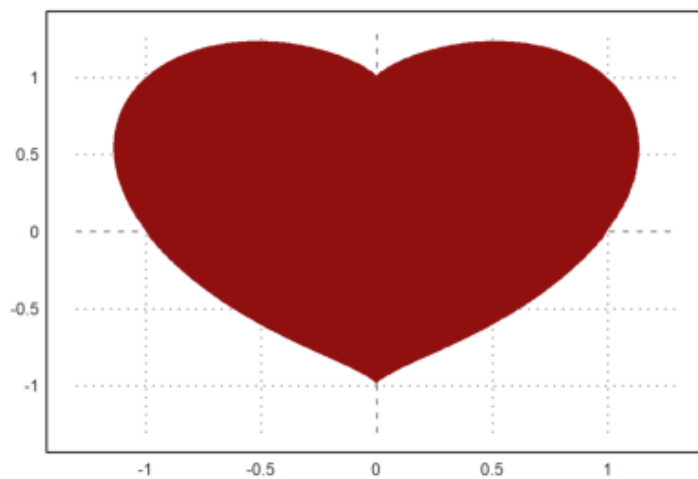
```
>plot2d("sin(x)*cos(y)", r=pi, >hue, >levels, n=100) :
```



Anda juga dapat menandai suatu wilayah

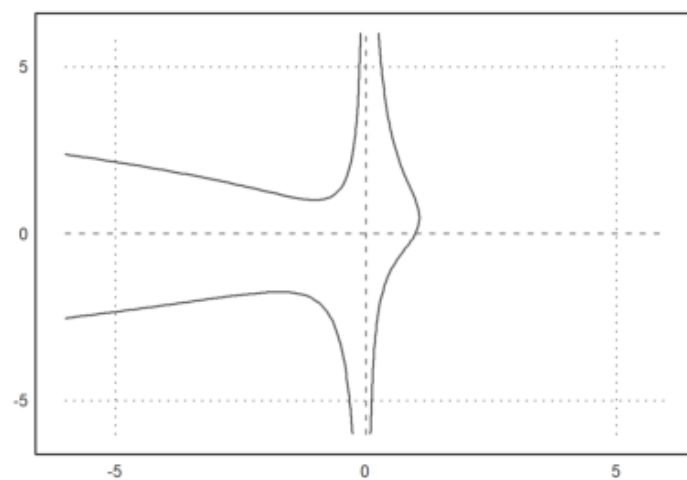
Hal ini dilakukan dengan menambahkan level dengan dua baris.

```
>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

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



```

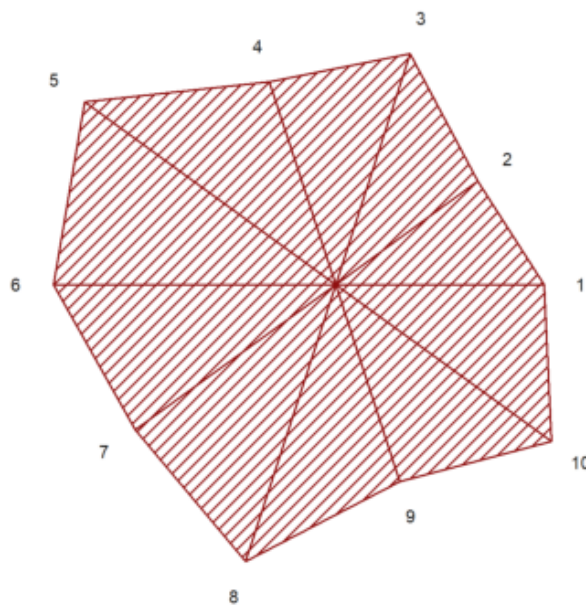
>function starplot1 (v, style="/", color=green, lab=none) ...
    if !holding() then clg; endif;
    w=window(); window(0,0,1024,1024);
    h=holding(1);
    r=max(abs(v))*1.2;
    setplot(-r,r,-r,r);
    n=cols(v); t=linspace(0,2pi,n);
    v=v|v[1]; c=v*cos(t); s=v*sin(t);
    cl=barcolor(color); st=barstyle(style);
    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

```

Tidak ada tanda centang pada grid atau sumbu di sini. Selain itu, kami menggunakan jendela penuh untuk plot.

Kami memanggil reset sebelum menguji plot ini untuk mengembalikan pengaturan grafis default. Ini tidak perlu dilakukan jika Anda yakin plot Anda berfungsi.

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



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

Dalam fungsi berikut, kita melakukan plot impuls logaritmik. Plot2d dapat melakukan 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

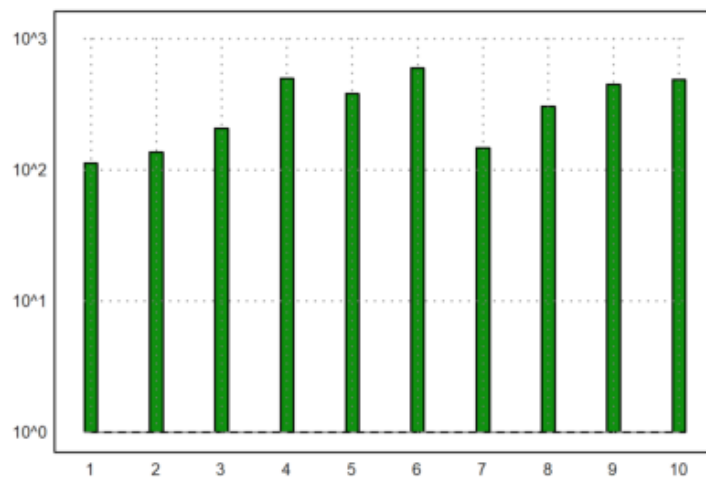
```

Mari kita mengujinya dengan nilai yang didistribusikan secara eksponensial.

```

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

```



Mari kita animasikan kurva 2D menggunakan plot langsung. Perintah `plot(x,y)` hanya memplot kurva ke dalam jendela plot. `setplot(a,b,c,d)` mengatur jendela ini.

Fungsi `wait(0)` memaksa plot untuk muncul di jendela grafik. Jika tidak, penggambaran ulang akan dilakukan dalam interval waktu yang jarang.

```

>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 tombol apa saja untuk menghentikan animasi ini.

```

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

```

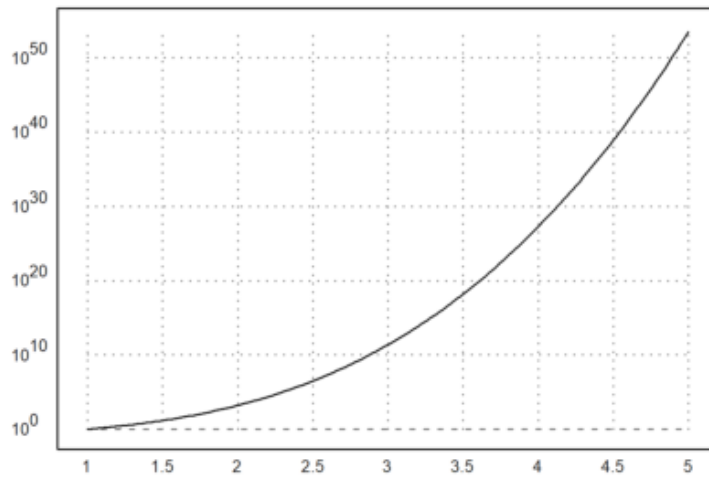
Plot Logaritma

EMT menggunakan parameter "logplot" untuk skala logaritma.

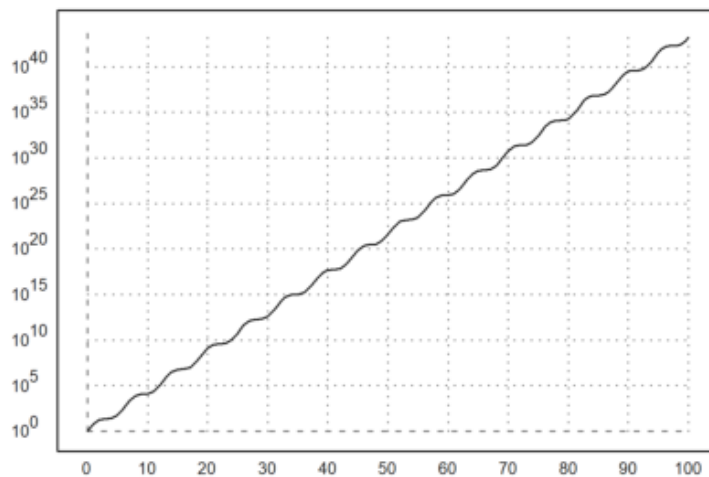
Plot logaritma dapat diplot menggunakan skala logaritma pada y dengan `logplot=1`, atau menggunakan skala logaritma pada x dan y dengan `logplot=2`, atau pada x dengan `logplot=3`.

- `logplot=1`: y-logaritma
- `logplot=2`: x-y-logaritma
- `logplot=3`: x-logaritma

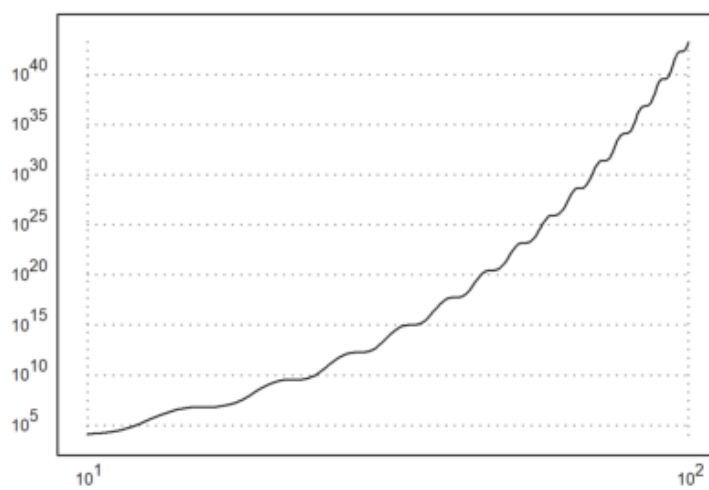
```
>plot2d("exp(x^3-x)*x^2",1,5,logplot=1):
```



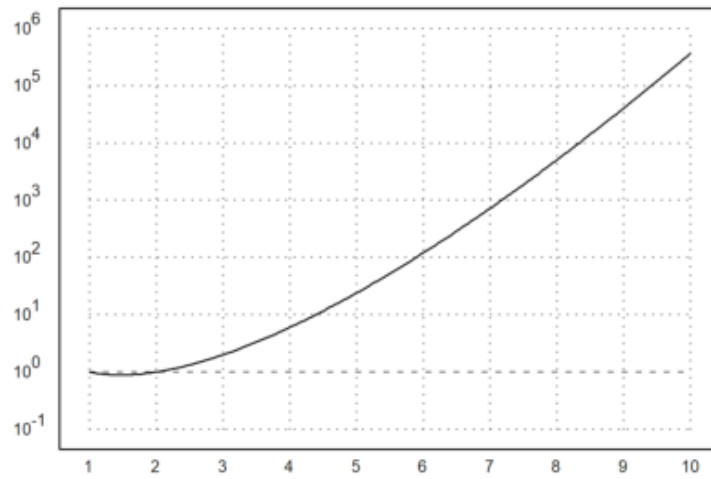
```
>plot2d("exp(x+sin(x))",0,100,logplot=1):
```



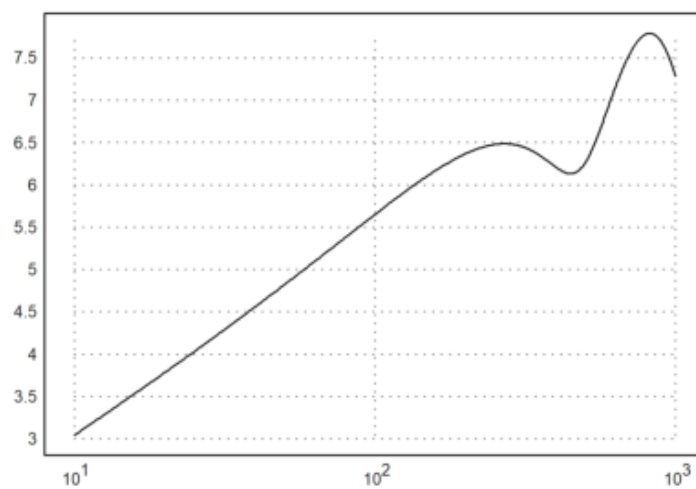
```
>plot2d("exp(x+sin(x))",10,100,logplot=2):
```



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

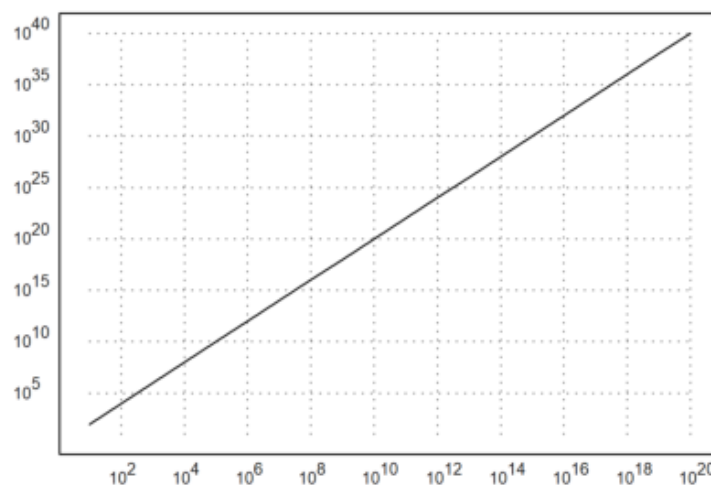


```
>plot2d("log(x*(2+sin(x/100)))",10,1000,logplot=3):
```



Ini juga berlaku untuk plot data.

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



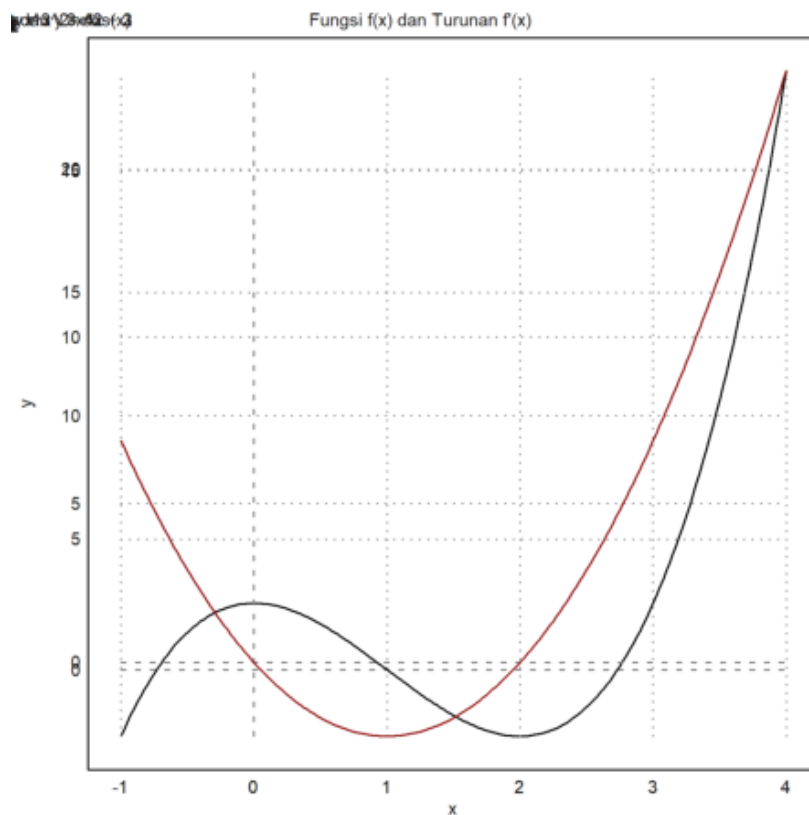
```
> x = linspace(0,6,200);
> y = x^2 - 6*x + 8;
> plot2d(x,y);
> title("Fungsi Kuadrat y = x^2 - 6x + 8");
```

```
> xlabel("x");
> ylabel("y");

> function f(x) &= x^3 - 3*x^2 + 2;
> function df(x) &= diff(f(x),x); // turunan
> x = linspace(-1,4,200);
> plot2d(x,f(x));
> hold on;
> plot2d(x,df(x),color=red);
> hold off;
> title("Fungsi f(x) dan Turunan f'(x)");
> xlabel("x");
> ylabel("y");
> legend("f(x)", "f'(x)");
```

```
Function legend not found.
Try list ... to find functions!
Error in:
  legend("f(x)", "f'(x)"); ...
```

```
> figure(1); plot2d("x^2-4*x+3",grid=5);
> title("Fungsi Kuadrat y = x^2 - 4x + 3");
> figure(2); plot2d("x^3-3*x^2+2",grid=5);
> title("Fungsi Kubik y = x^3 - 3x^2 + 2");
> figure(3); plot2d("sin(x)",0,2*pi,grid=7);
> hold on;
> plot2d("cos(x)",0,2*pi,"green");
> hold off;
> title("Grafik y = sin(x) dan y = cos(x)");
>figure(1):
```



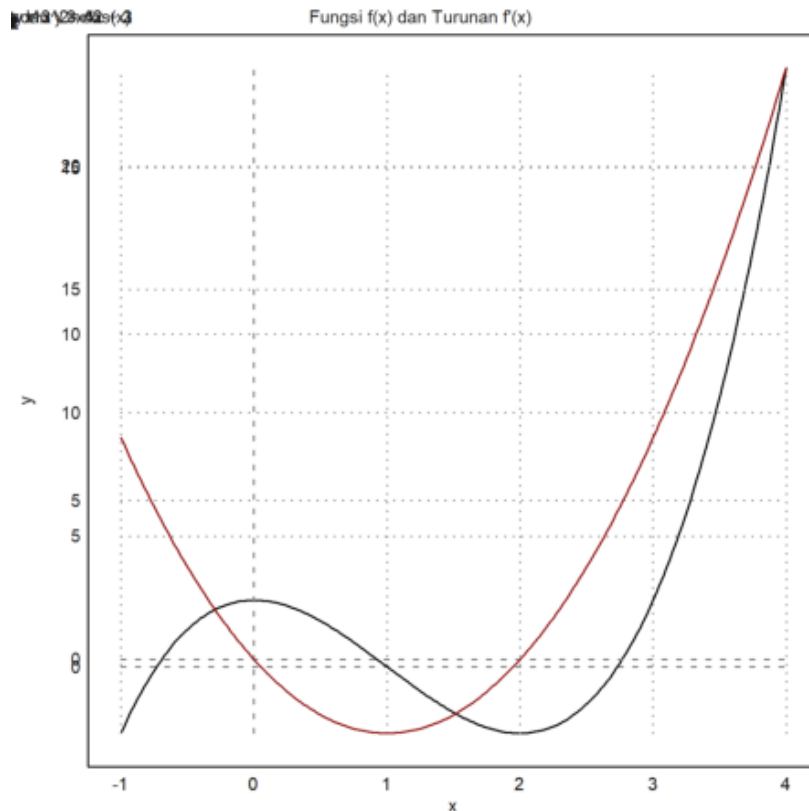
```
>figure(5); plot2d("x^3-x",grid=4); // label saja
>figure(6); plot2d("x^3-x",grid=5); // default
>figure(7); plot2d("x^3-x",grid=6); // axes only
>figure(8); plot2d("x^3-x",grid=7); // axes + ticks di sumbu
>figure(9); plot2d("x^3-x",grid=8); // axes + ticks rapat
>figure(10); plot2d("x^3-x",grid=9); // ticks kecil di dalam
>figure(11); plot2d("x^3-x",grid=10); // axes only, tanpa ticks
```

```

Space between commands expected!
Found: // axes only, tanpa ticks (character 47)
You can disable this in the Options menu.
Error in:
figure(11); plot2d("x^3-x",grid=10);// axes only, tanpa ticks ...
^

```

>figure (5):



```

> x = linspace(-2,6,200);
> y = x^2 - 4*x + 3;
> plot2d(x,y);
> title("Grafik Kuadrat y = x^2 - 4x + 3");
> xlabel("x"); ylabel("y");
> grid(1);

```

```

Function grid not found.
Try list ... to find functions!
Error in:
grid(1); ...
^

```

grid pada EMT punya banyak opsi tampilan.
Bisa dipakai untuk menyesuaikan grafik sesuai kebutuhan (misalnya untuk laporan lebih rapi).

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
 "[]", "<>", ".", "...", "...",
 "x", "+", "|", "-", "o"
 "[#]", "<>#", "o#" (filled shapes)
 "[w]", "<>w", "ow" (non-transparent)
 For lines use
 "-", "--", "-.", ".", "-.-", "-.-", "->"
 For filled polygons or bar plots use
 "#", "#O", "O", "/", "\", "V",
 "+", "|", "-", "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 $2 \times n$ 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 [-2,2] 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.