

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).

Plot Dasar

Ada fungsi plot yang sangat mendasar. Terdapat koordinat layar, yang selalu berkisar dari 0 hingga 1024 di setiap sumbu, tidak peduli apakah layarnya persegi atau tidak. Terdapat koordinat plot, yang dapat diatur dengan `setplot()`. Pemetaan antara koordinat tergantung pada jendela plot saat ini. Sebagai contoh, default `shrinkwindow()` menyisakan ruang untuk label sumbu dan judul plot.

Pada contoh, kita hanya menggambar beberapa garis acak dalam berbagai warna. Untuk detail mengenai fungsi-fungsi ini, pelajari fungsi-fungsi inti dari EMT.

```
>clg; // membersihkan layar
>window(0,0,1024,1024); // menggunakan semua jendela
>setplot(0,1,0,1); // mengatur koordinat plot
>hold on; // memulai mode penimpaan
>n=100; X=random(n,2); Y=random(n,2); // mendapatkan poin acak
>colors=rgb(random(n),random(n),random(n)); // mendapatkan warna acak
>loop 1 to n; color(colors[#]); plot(X[#],Y[#]); end; // plot
>hold off; // mengakhiri mode penimpaan
>insimg; // memasukan ke catatan
```



```
>reset; // mereset grafik sebelumnya agar tidak tertimpa
```

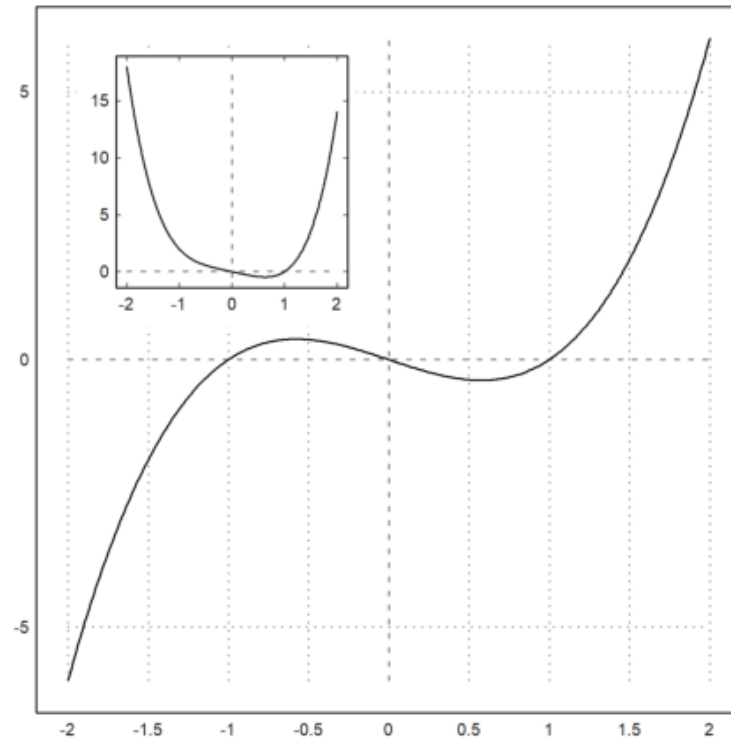
Anda harus menahan grafik, karena perintah `plot()` akan menghapus jendela plot.

Untuk menghapus semua yang telah kita lakukan, kita 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 gunakan perintah `insimg()` untuk menampilkan gambar hasil plot.

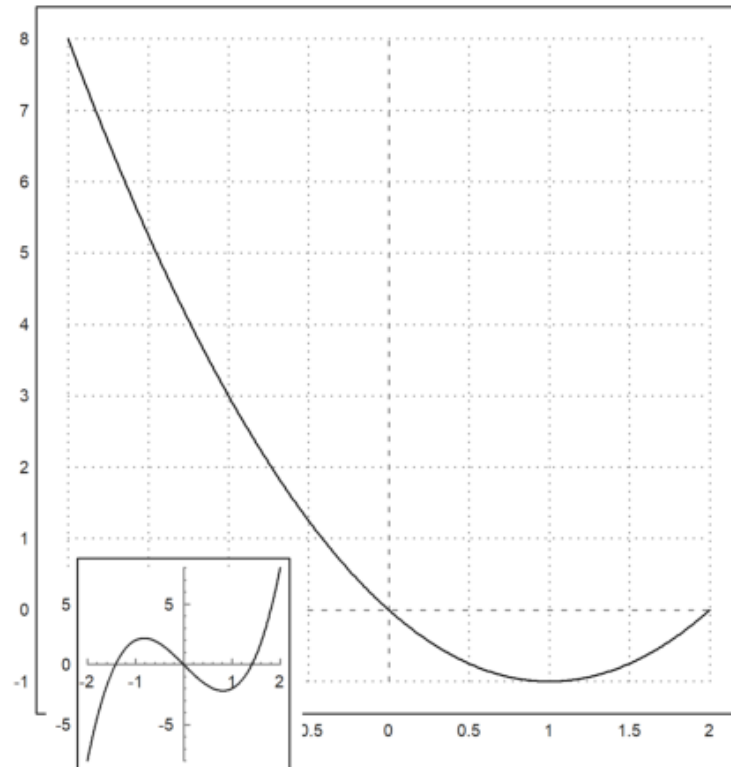
Sebagai contoh lain, kita menggambar sebuah plot sebagai inset pada plot yang lain. Hal ini dilakukan dengan mendefinisikan jendela plot yang lebih kecil. Perhatikan bahwa jendela ini tidak menyediakan ruang untuk label sumbu di luar jendela plot. Kita harus menambahkan beberapa margin untuk hal ini sesuai kebutuhan. Perhatikan bahwa kita menyimpan dan mengembalikan jendela penuh, dan menahan plot saat ini sementara kita membuat 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);  
>reset;  
>plot2d("x^2-2x");  
>xw=150; yw=750; ww=275; hw=275;  
>ow=window();  
>window(xw,yw,xw+ww,yw+hw);
```

```
>hold on;  
>barclear(xw-40,yw+20,ww+55,ww-15);  
>plot2d("2*x^3-4x",grid=8):
```

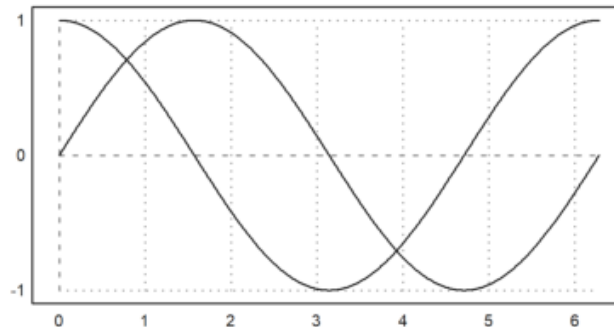


Plot dengan beberapa angka dicapai dengan cara yang sama. Ada fungsi utility figure() untuk ini.

Plot default menggunakan jendela plot persegi. Anda dapat mengubahnya dengan fungsi `aspect()`. Jangan lupa untuk mengatur ulang aspeknya nanti. Anda juga dapat mengubah default ini di menu dengan “Set Aspect” ke rasio aspek tertentu atau ke ukuran jendela grafik saat ini.

Tetapi Anda juga dapat mengubahnya untuk satu plot. Untuk ini, ukuran area plot saat ini diubah, dan jendela diatur sedemikian rupa sehingga label memiliki ruang yang cukup.

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



```
>aspect();  
>reset;  
>aspect(4); //rasio panjang dan lebar 4:1
```

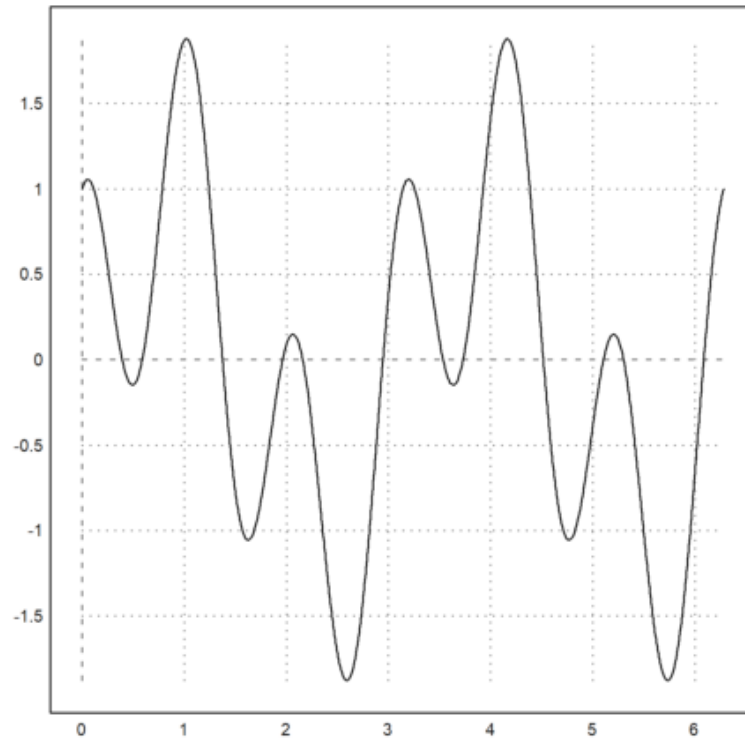
gambarlah plot

$$\cos(6x) + \sin(2x)$$

dengan rentang

$$0 \leq x \leq 2\pi$$

```
>plot2d("cos(6x)+sin(2x)",0,2pi):
```



Fungsi reset () mengembalikan default plot, termasuk rasio aspek.

Plot 2D di Euler

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

Dimungkinkan untuk memplot di Maxima menggunakan Gnuplot atau di Python menggunakan Math Plot Lib.

Euler dapat memplot plot 2D dari

- ekspresi
- fungsi, variabel, atau kurva berparameter,
- vektor nilai x-y,
- awan titik-titik di bidang,
- kurva implisit dengan level atau wilayah level.
- Fungsi yang kompleks

Gaya plot mencakup berbagai gaya untuk garis dan titik, plot batang, dan plot berbayang.

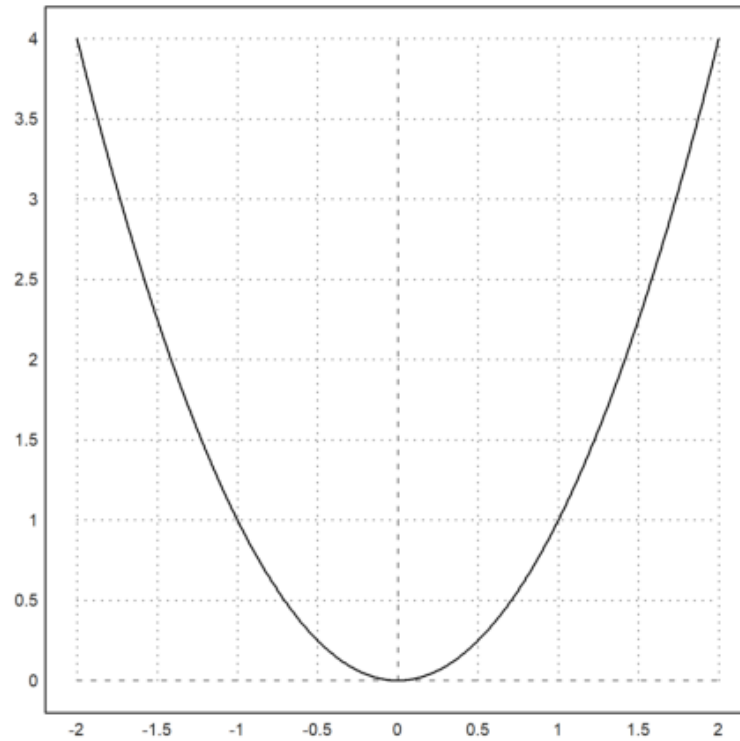
Plot Ekspresi atau Variabel

Ekspresi tunggal dalam “x” (misalnya “4*x^2”) atau nama fungsi (misalnya “f”) menghasilkan grafik fungsi.

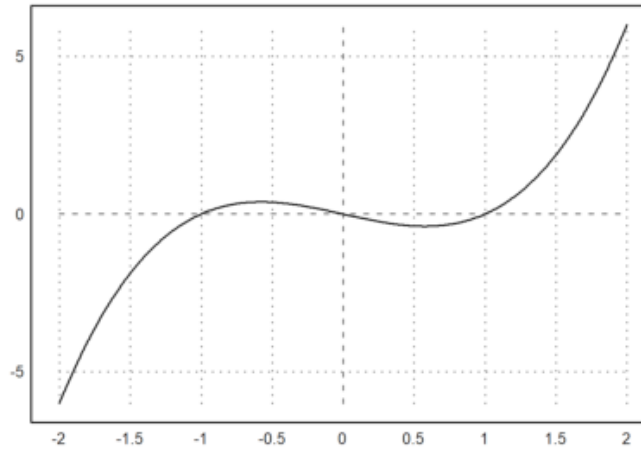
Berikut ini adalah contoh paling dasar, yang menggunakan rentang default dan menetapkan rentang y yang tepat agar sesuai dengan plot fungsi.

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

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



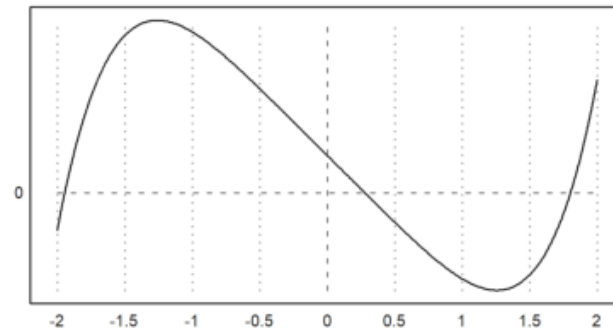
```
>aspect(1.5); plot2d("x^3-x"):
```



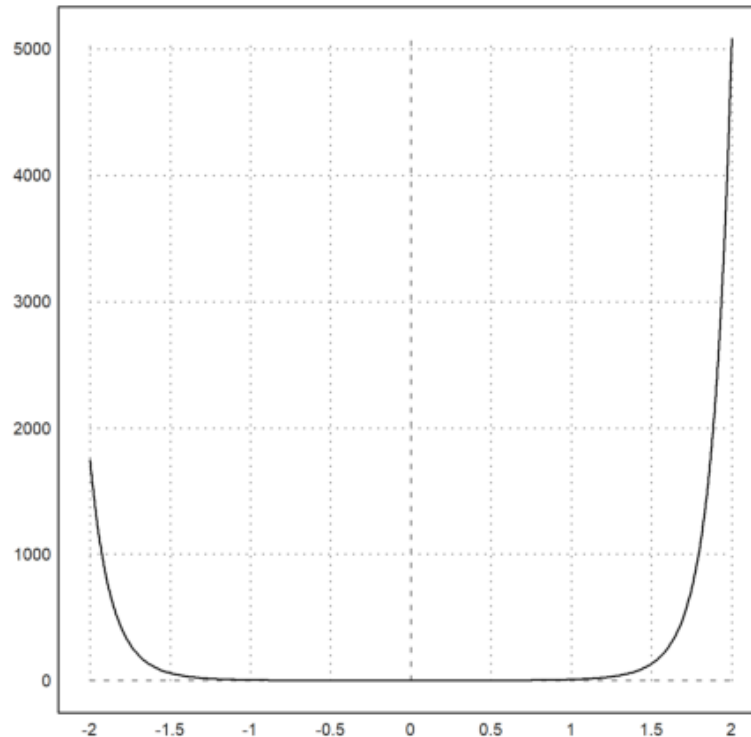
gambarlah plot berikut

$$2x^5 - 25x + 7$$

```
>aspect(2); plot2d("2*x^5-25*x+7"): // tanda titik dua(:) memunculkan grafik
```



```
>a:=5.6; plot2d("exp(-a*x^2)/a"); insimg(30); // menampilkan gambar hasil plot setinggi 25 baris
>reset; ...
>a:=7.5; plot2d("exp(2*x^2+2*x/a)");
```

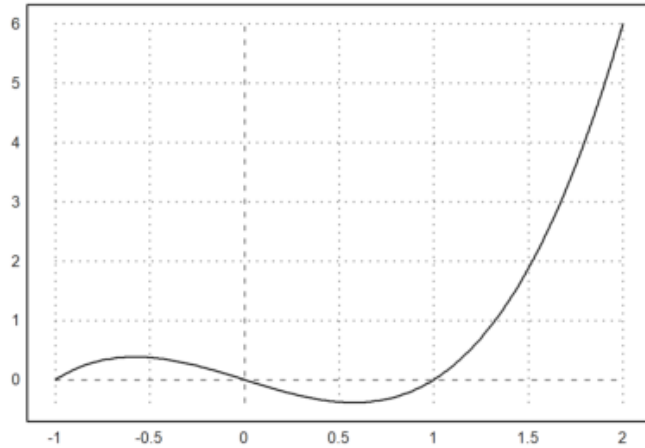


Dari beberapa contoh sebelumnya Anda dapat melihat bahwa aslinya gambar plot menggunakan sumbu X dengan rentang nilai dari -2 sampai dengan 2. Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan nilai-nilai batas X (dan Y) di belakang ekspresi yang digambar.

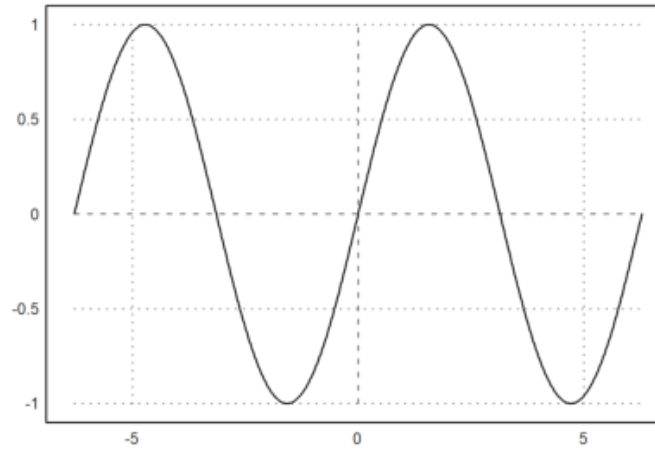
Rentang plot ditetapkan dengan parameter yang ditetapkan berikut ini

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

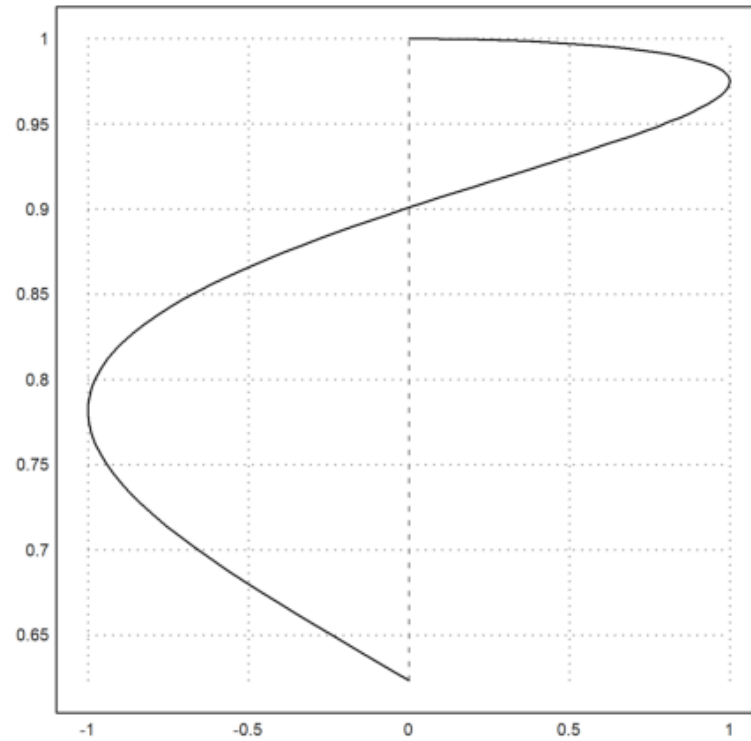
```
>plot2d("x^3-x",-1,2):
```



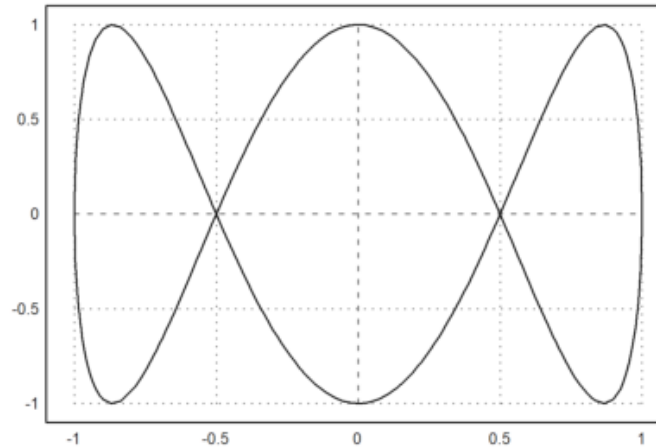
```
>plot2d("sin(x)",-2*pi,2*pi): // plot sin(x) pada interval [-2pi, 2pi]
```



```
>reset; plot2d("sin(x)","cos(x/7)",xmin=0,xmax=2pi): //memplot dengan xmin=0 dan xmax=2pi
```

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



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

Dalam opsi, plot dapat diatur untuk muncul

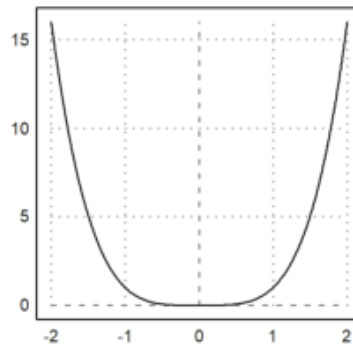
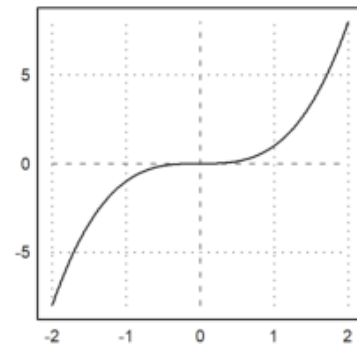
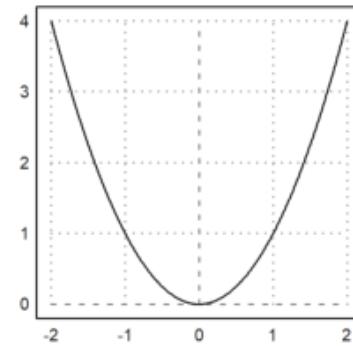
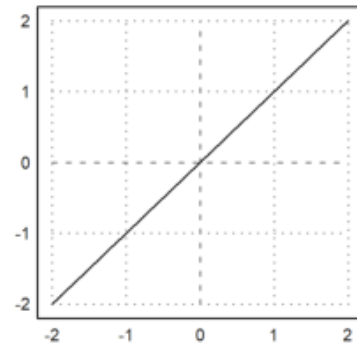
- di jendela terpisah yang dapat diubah ukurannya,
- di jendela buku catatan.

Lebih banyak gaya yang dapat dicapai dengan perintah plot tertentu.

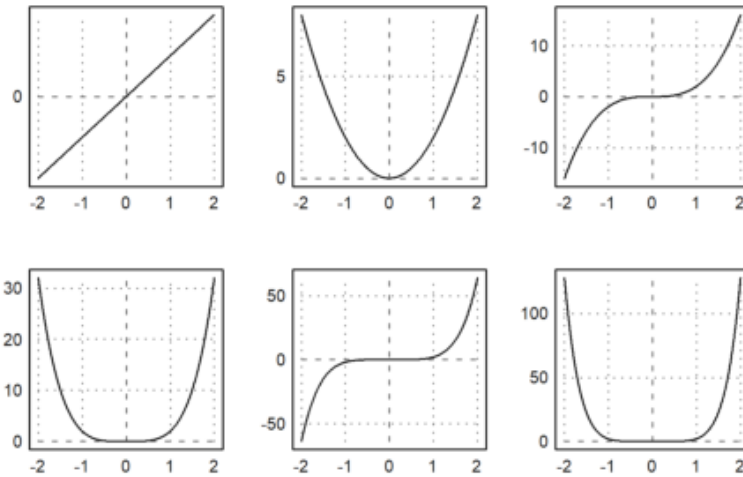
Dalam hal apa pun, tekan tombol tabulator untuk melihat plot, jika disembunyikan.

Untuk membagi jendela menjadi beberapa plot, gunakan perintah `figure()`. Pada contoh, kita memplot x^1 sampai x^4 ke dalam 4 bagian jendela. `figure(0)` akan mereset jendela default.

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



```
>reset; ...
>figure(3,3); ...
>for n=1 to 6; figure(n); plot2d("2*x^"+n); end; ...
>figure(0):
```

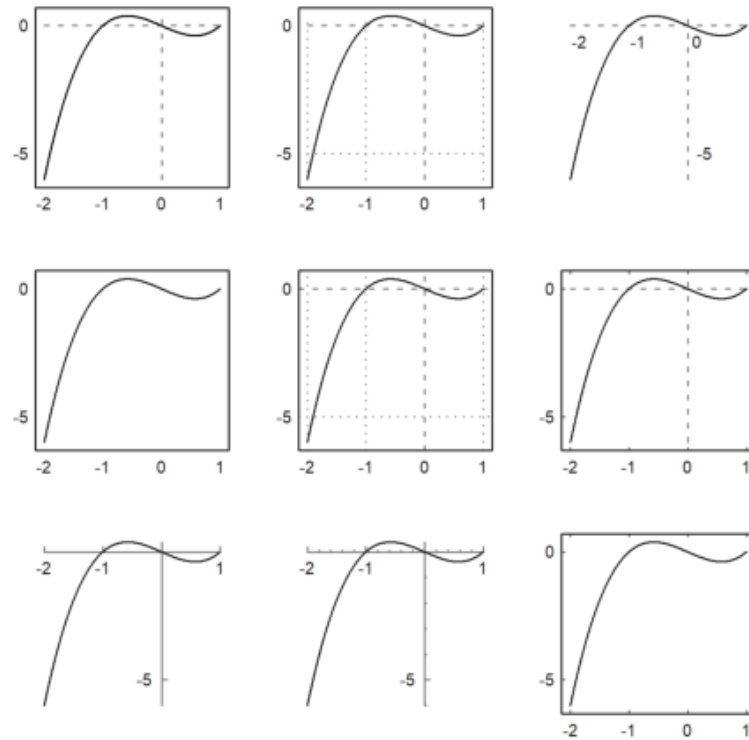


Pada `plot2d()`, terdapat beberapa gaya alternatif yang tersedia dengan `grid=x`. Sebagai gambaran umum, kami menampilkan berbagai gaya grid dalam satu gambar (lihat di bawah ini untuk perintah `figure()`). Gaya `grid=0` tidak disertakan. Gaya ini tidak menampilkan grid dan frame.

```

>reset; ...
>figure(3,3); ...
>for k=1:9; figure(k); plot2d("x^3-x",-2,1,grid=k); end; ...
>figure(0): // figure(0) untuk menampilkan grafik dijendela langsung

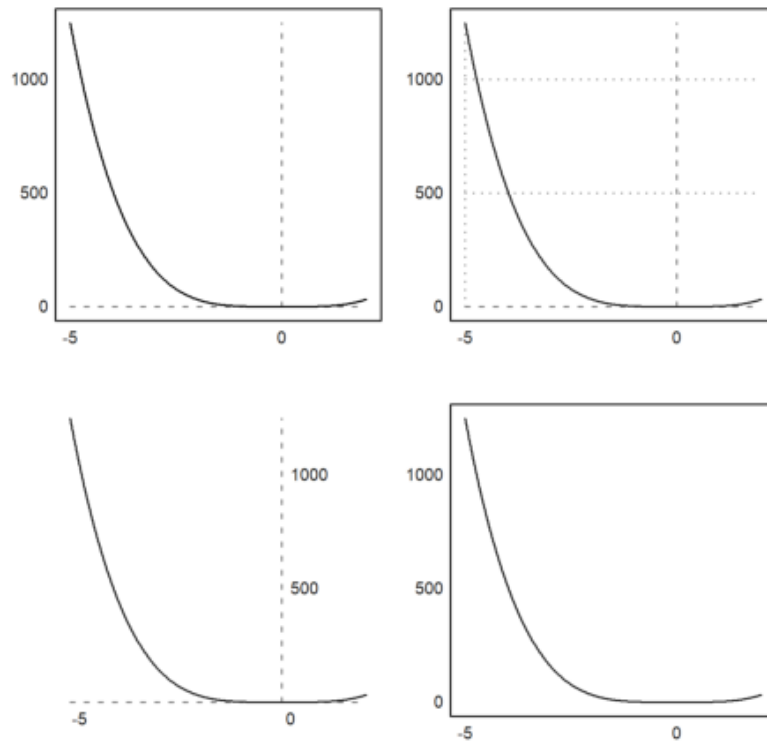
```



```

>reset; ...
>figure(2,2); ...
>for k=1:4; figure(k); plot2d("2*x^4",-5,2,grid=k); end; ...
>figure(0):

```

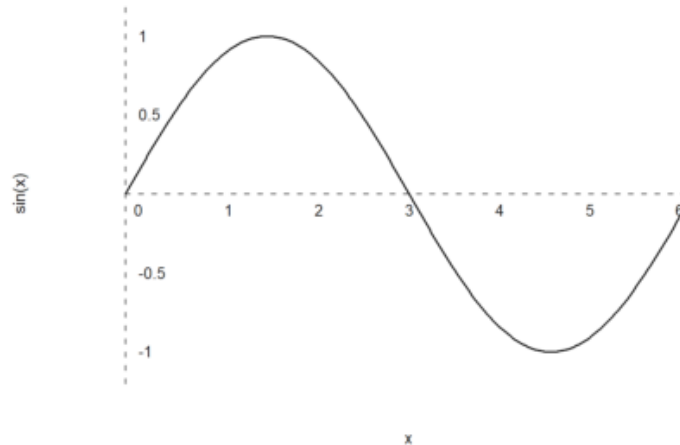


Jika argumen untuk `plot2d()` adalah sebuah ekspresi yang diikuti oleh empat angka, angka-angka ini adalah rentang x dan y untuk plot.

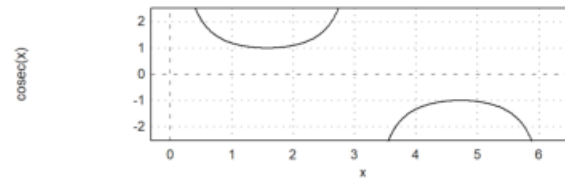
Atau, a , b , c , d dapat ditentukan sebagai parameter yang ditetapkan sebagai $a=\dots$ dst.

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

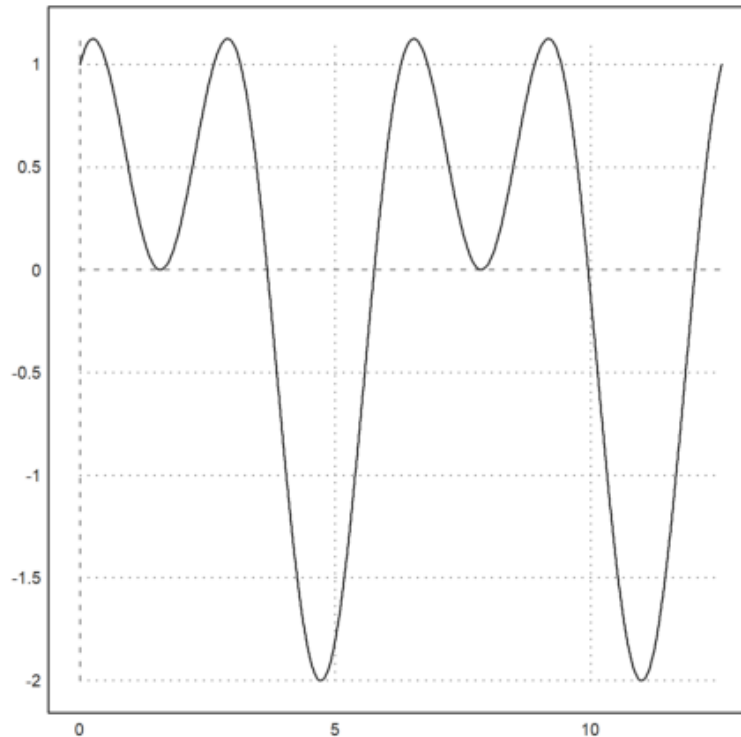
```
>aspect(1.5); plot2d("sin(x)",0,2pi,-1.2,1.2,grid=3,xl="x",yl="sin(x)");
```



```
>reset; aspect(3.2); plot2d("cosec(x)",0,2pi,-2.3,2.3,grid=5,xl="x",yl="cosec(x)");
```



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

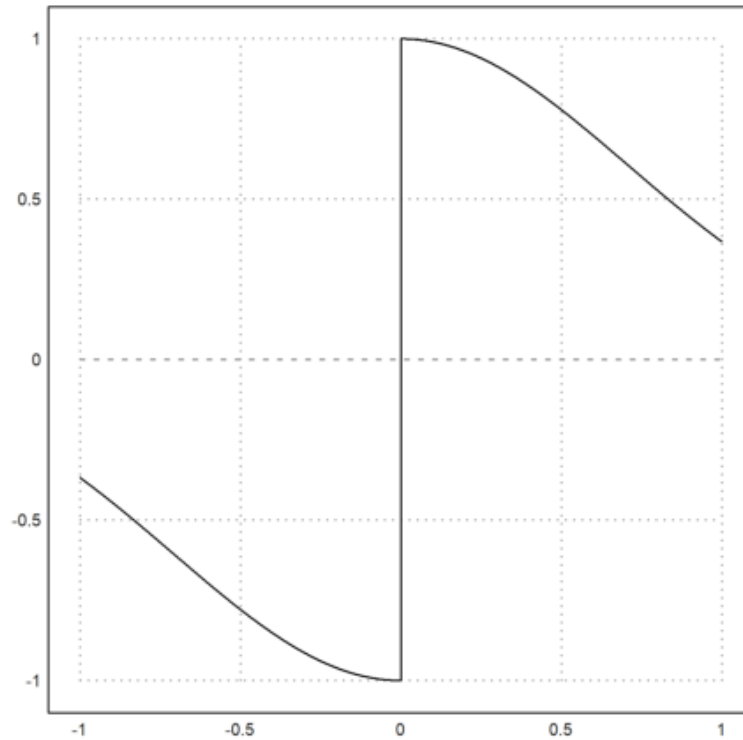



Gambar yang dihasilkan dengan menyisipkan plot ke dalam jendela teks disimpan dalam direktori yang sama dengan notebook, secara default dalam subdirektori bernama “images”. Gambar-gambar tersebut juga digunakan oleh ekspor HTML.

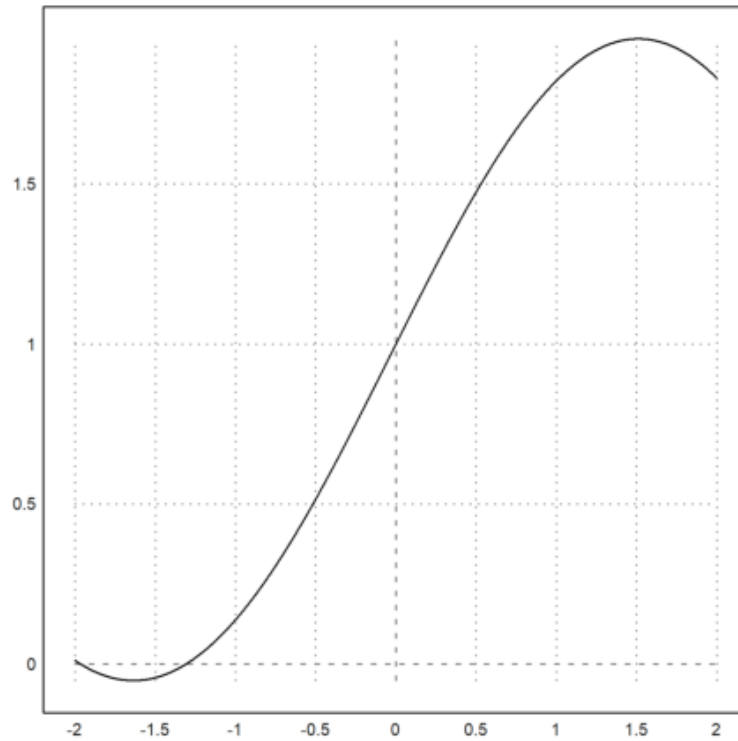
Anda cukup menandai gambar mana saja dan menyalinnya ke clipboard dengan Ctrl-C. Tentu saja, Anda juga dapat mengeksport grafik saat ini dengan fungsi-fungsi pada menu File.

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

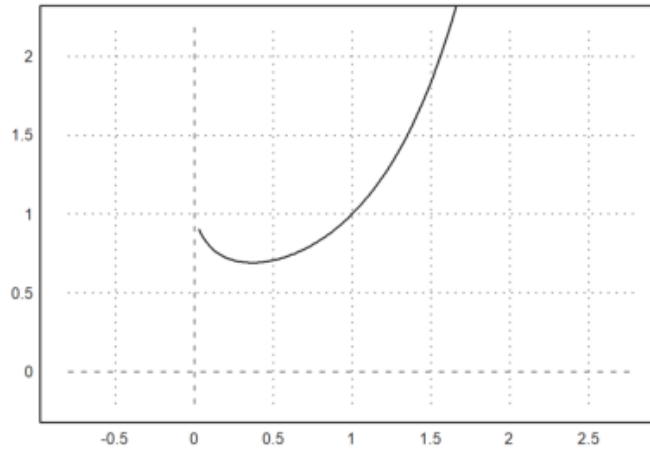
```
>reset; plot2d("sign(x)*exp(-x^2)",-1,1,<adaptive,n=10000):
```



```
>reset; plot2d("cos(x/5)+sin(x)",-2,2,<adaptive,n=2500):
```

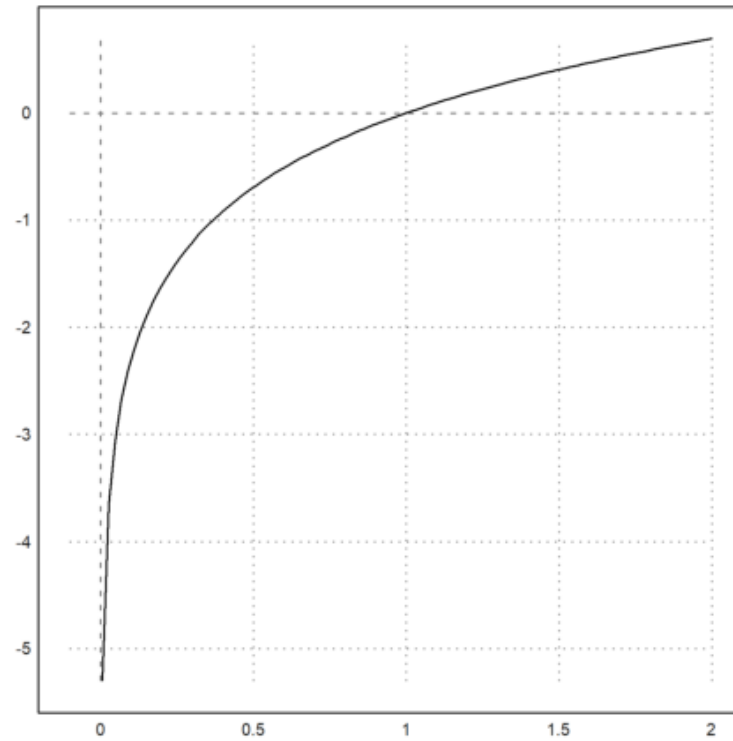


```
>plot2d("x^x",r=1.2,cx=1,cy=1):
```

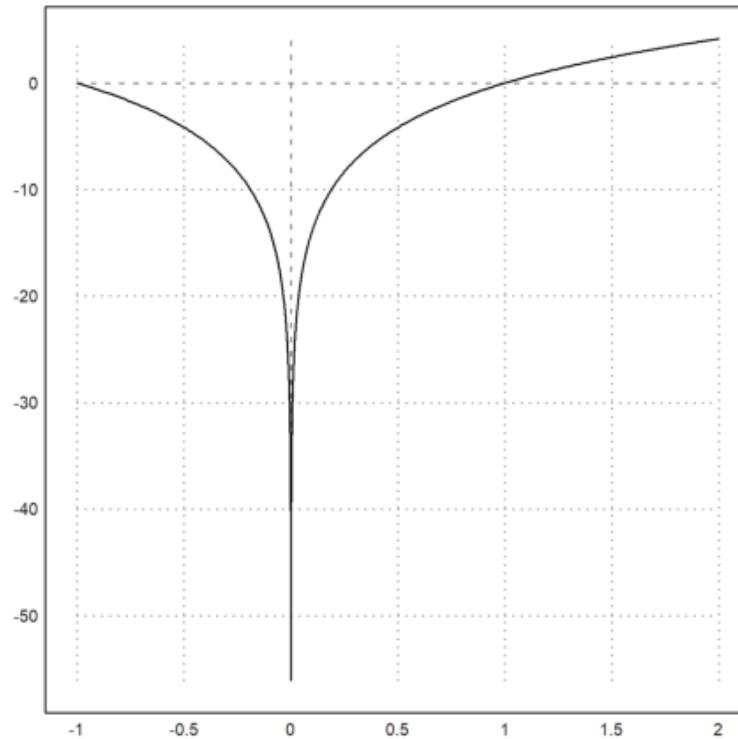


Perhatikan bahwa x^x tidak didefinisikan untuk $x \leq 0$. Fungsi `plot2d` menangkap kesalahan ini, dan mulai memplot segera setelah fungsi didefinisikan. Hal ini berlaku untuk semua fungsi yang mengembalikan NAN di luar jangkauan definisinya.

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

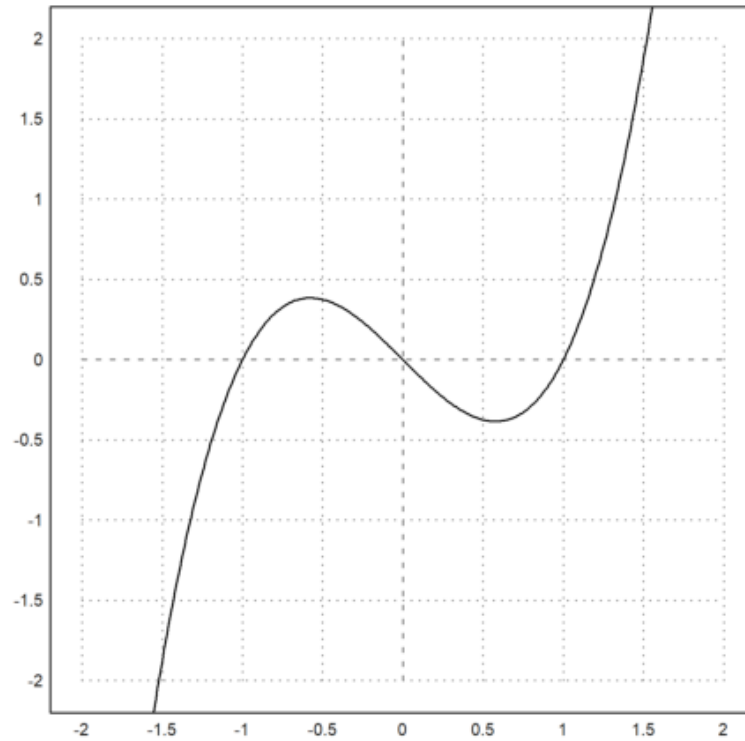


```
>reset; plot2d("3*log(x^2)",-1,2):
```

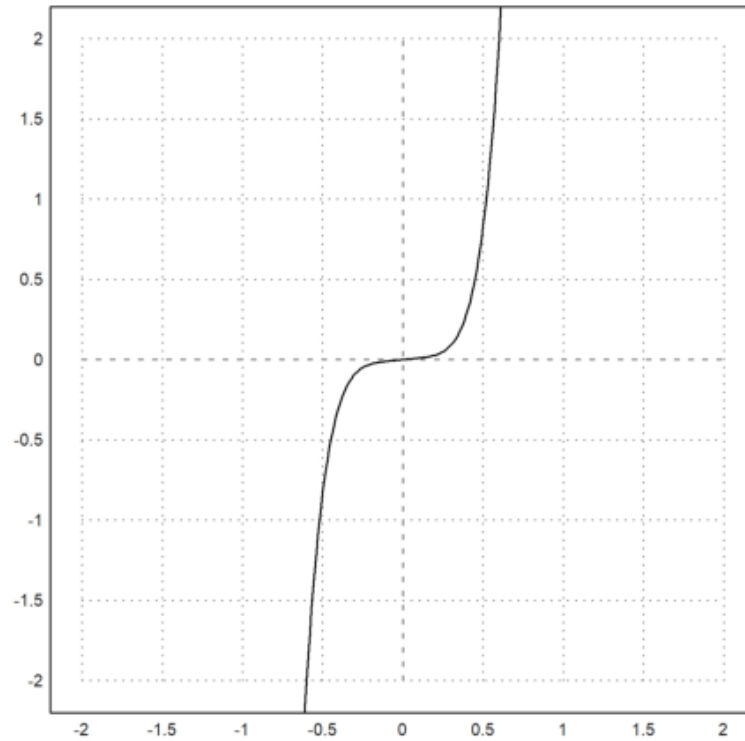


Parameter `square=true` (atau `>square`) memilih rentang y secara otomatis sehingga hasilnya adalah jendela plot persegi. Perhatikan bahwa secara default, Euler menggunakan ruang persegi di dalam jendela plot.

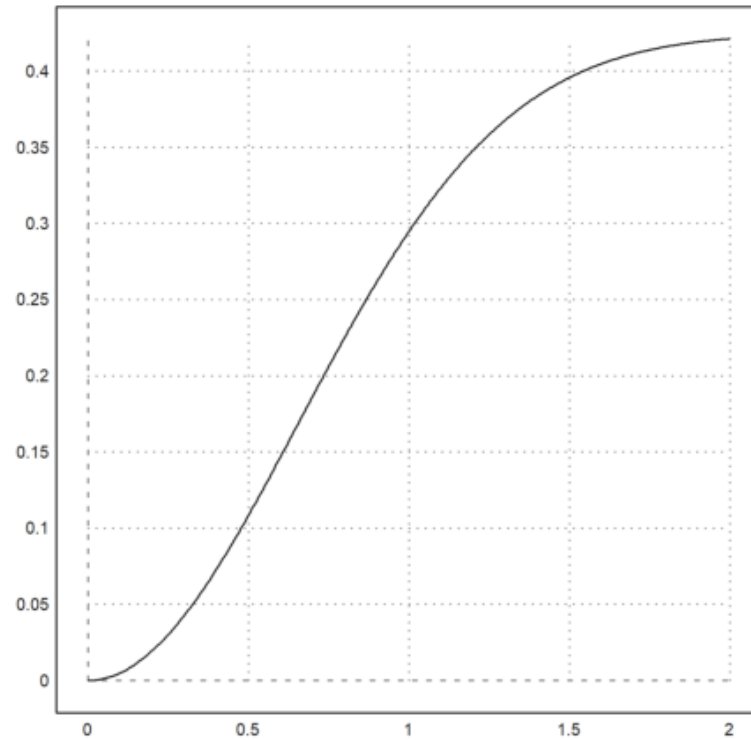
```
>reset; plot2d("x^3-x",>square):
```



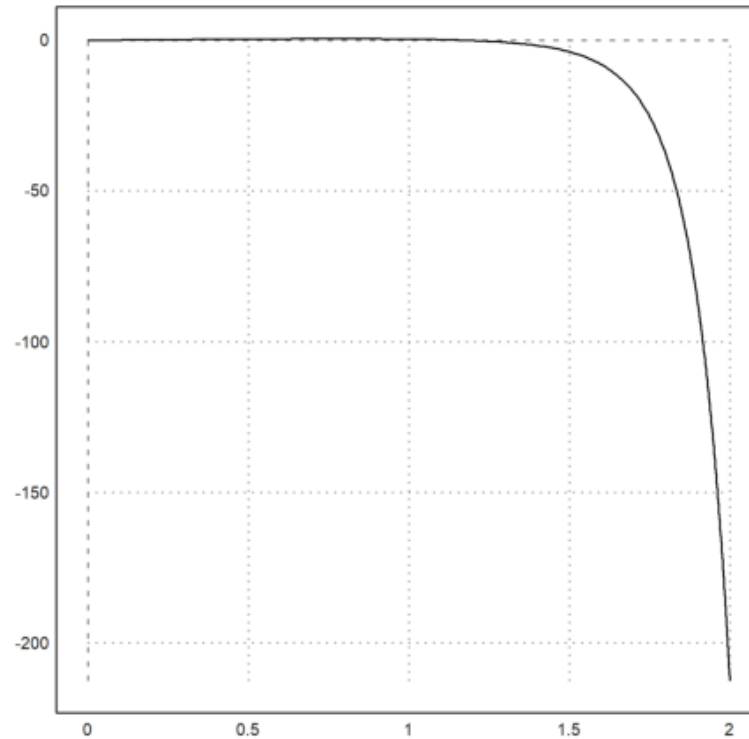
```
>reset; plot2d("25*x^5+x/10",>square):
```



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

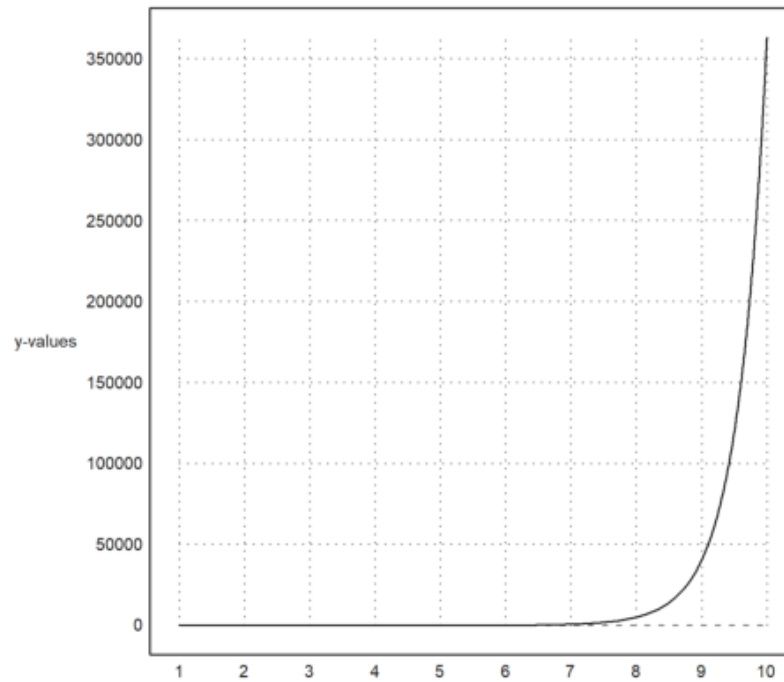



```
>plot2d(''integrate("cos(2*x)*exp(x^3)",0,x)''',0,2):
```

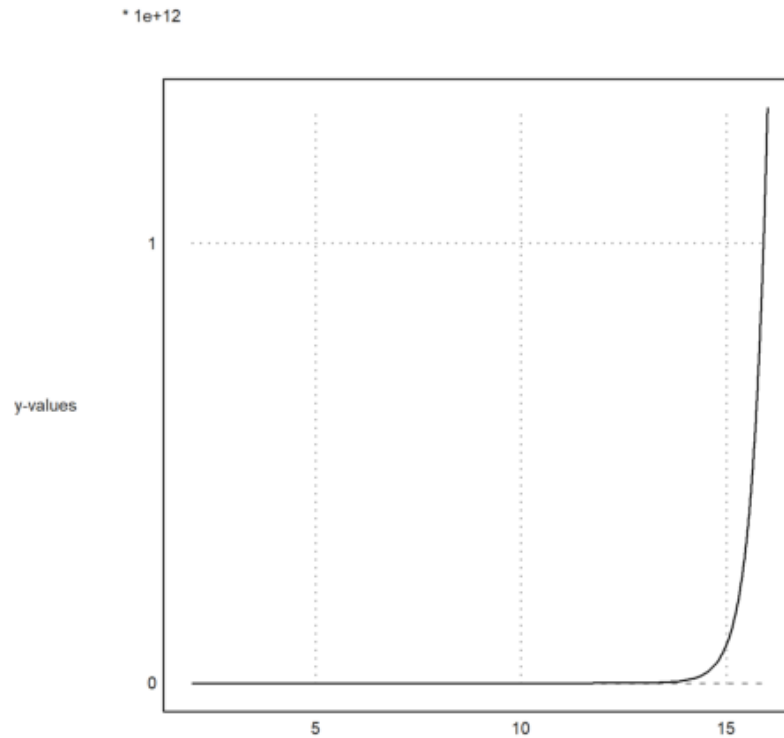


Jika Anda membutuhkan lebih banyak ruang untuk label-y, panggil `shrinkwindow()` dengan parameter lebih kecil, atau tetapkan nilai positif untuk “lebih kecil” pada `plot2d()`.

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

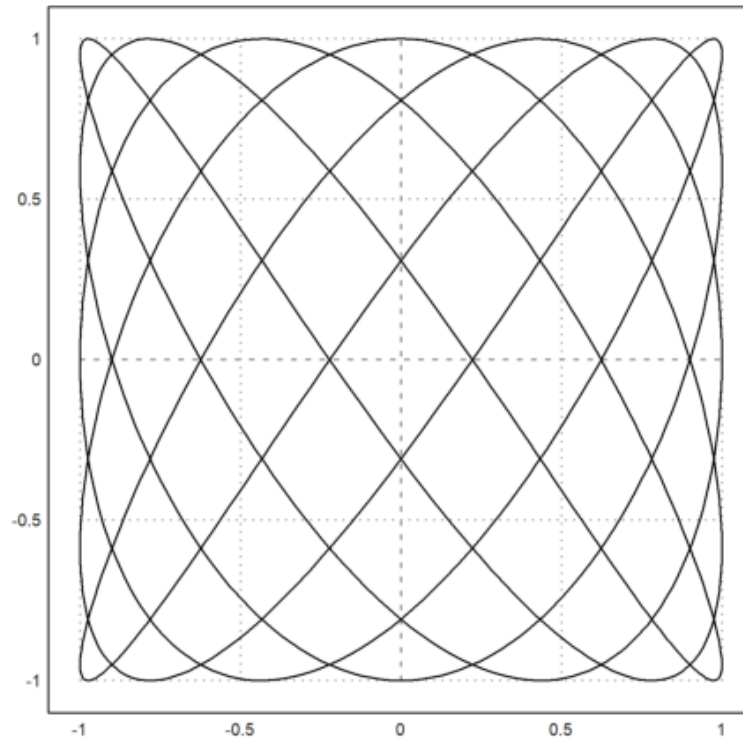


```
>reset; plot2d("gamma(x)",2,16,yl="y-values",smaller=8,<vertical):
```

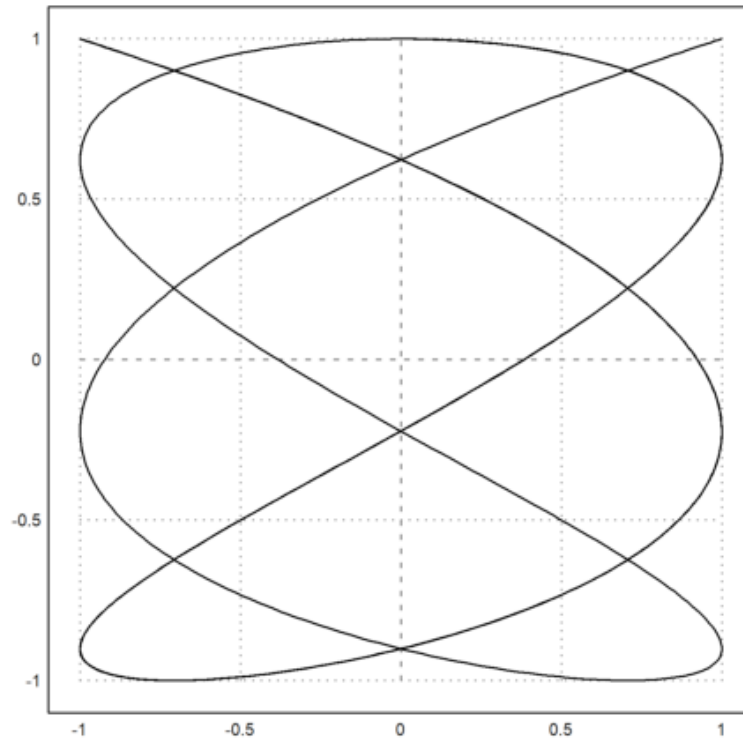


Ekspresi simbolik juga dapat digunakan, karena disimpan sebagai ekspresi string sederhana.

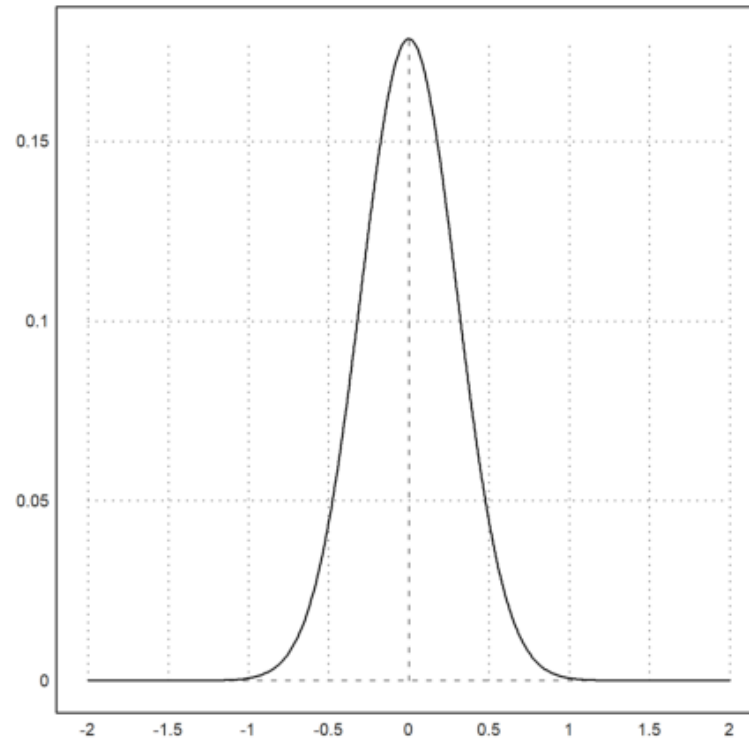
```
>reset; x=linspace(0,2pi,1000); plot2d(sin(5x),cos(7x)):
```



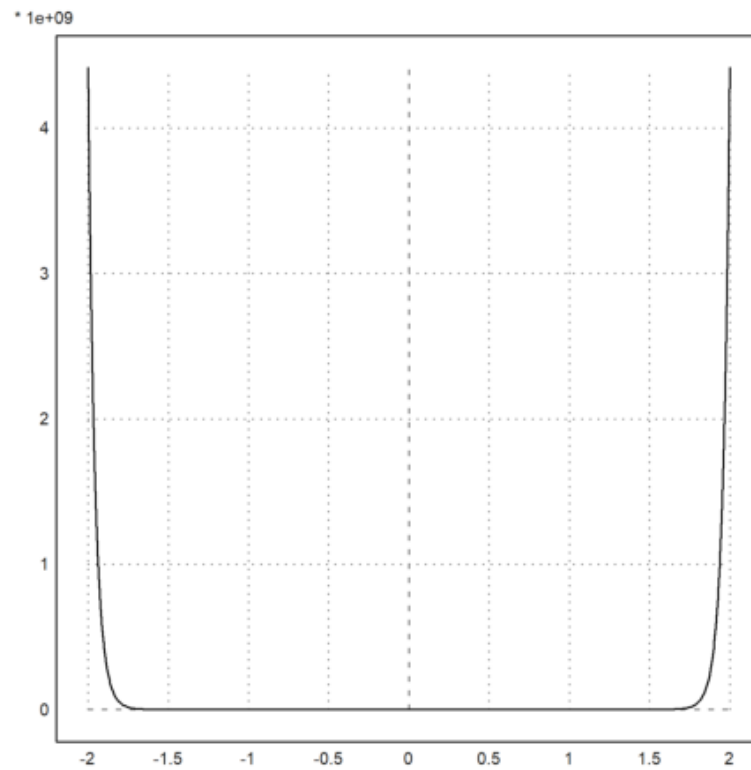
```
>reset; x=linspace(0,3pi,2500); plot2d(sin(7x),cos(4x)):
```



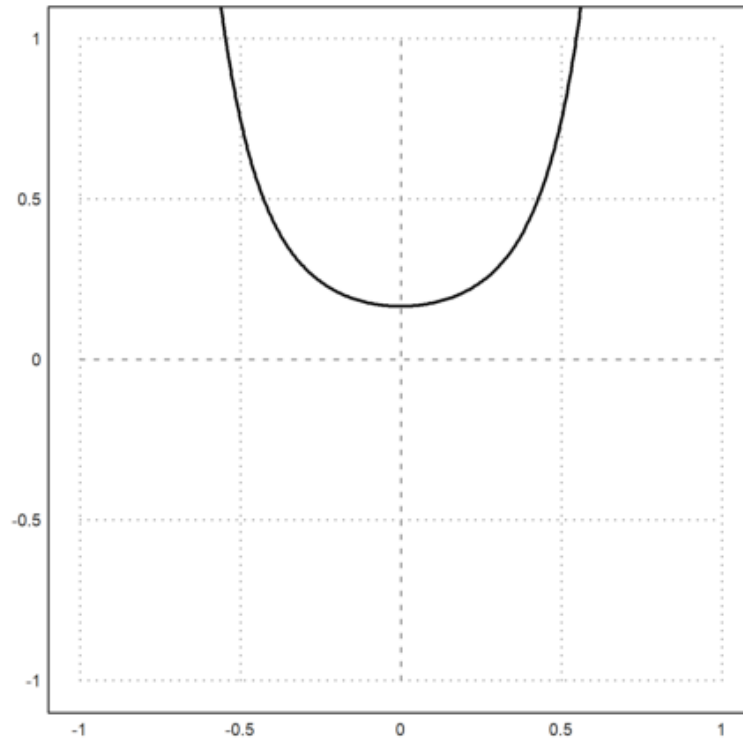
```
>reset; a:=5.6; expr &= exp(-a*x^2)/a; // define expression  
>plot2d(expr,-2,2): // plot from -2 to 2
```



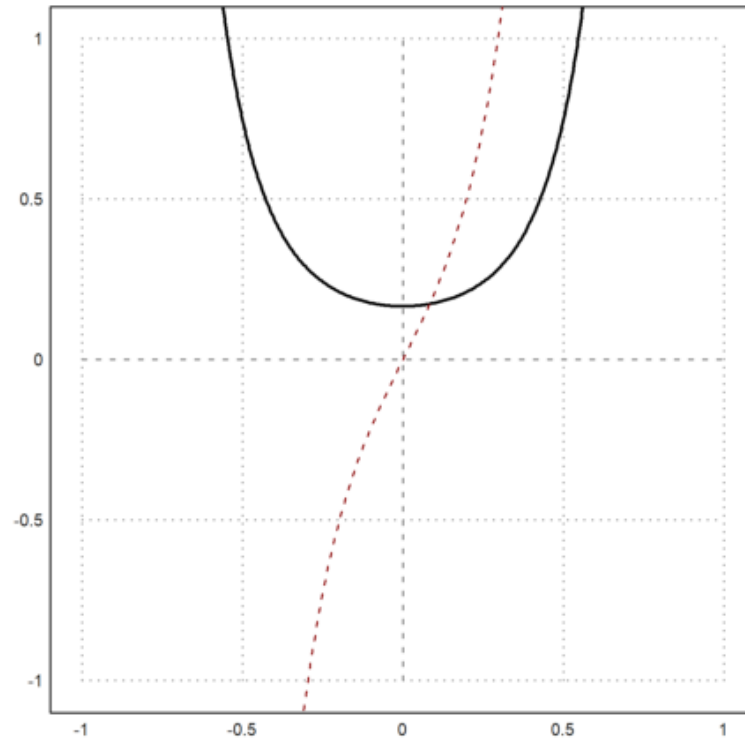
```
>reset; b:=6; expr &= exp(b*x^2)/b; ...  
>plot2d(expr,-2,2):
```



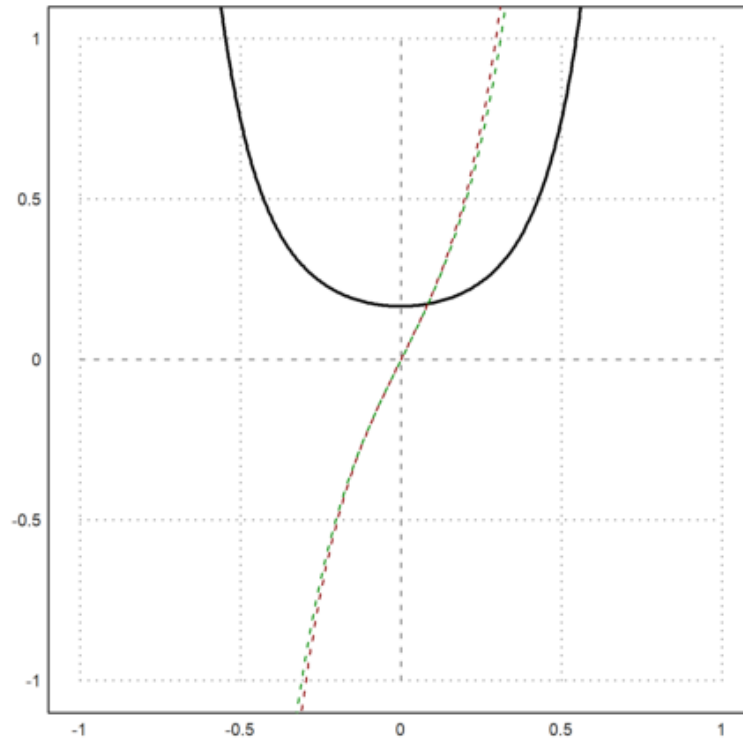
```
>reset; plot2d(expr,r=1,thickness=2): // plot di sekitar persegi (0,0)
```

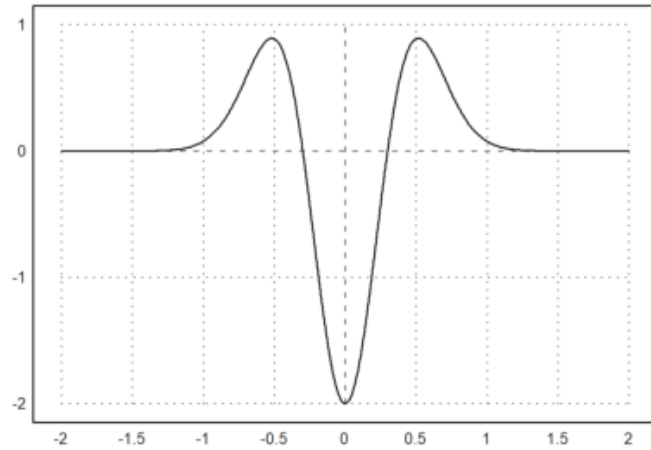
```
>plot2d(&diff(expr,x),>add,style="--",color=red): // add another plot
```



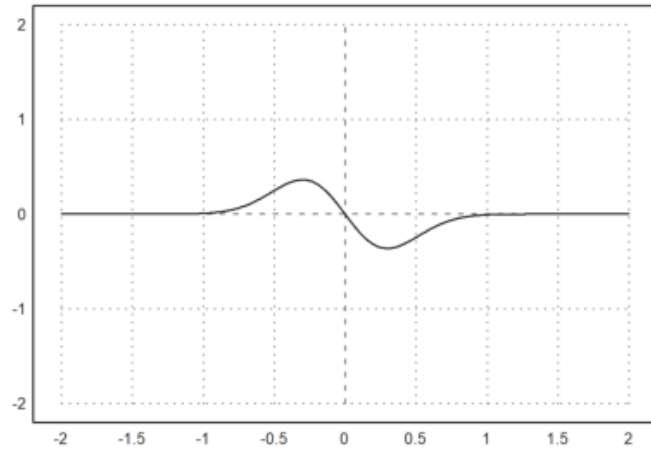
```
>reset; plot2d(&diff(expr,x),>add,style="--",color=green):
```



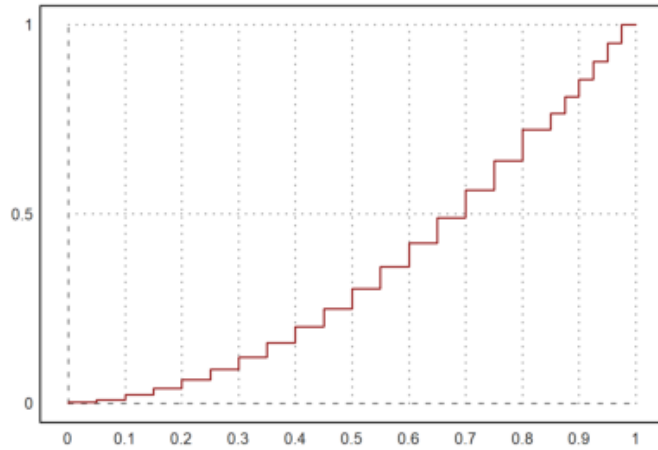
```
>plot2d(&diff(expr,x,2),a=-2,b=2,c=-2,d=1): // plot in rectangle
```



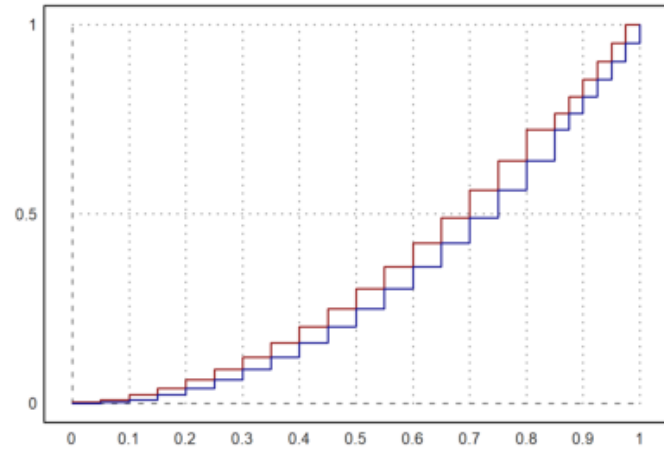
```
>plot2d(&diff(expr,x),a=-2,b=2,>square): // keep plot square
```



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



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

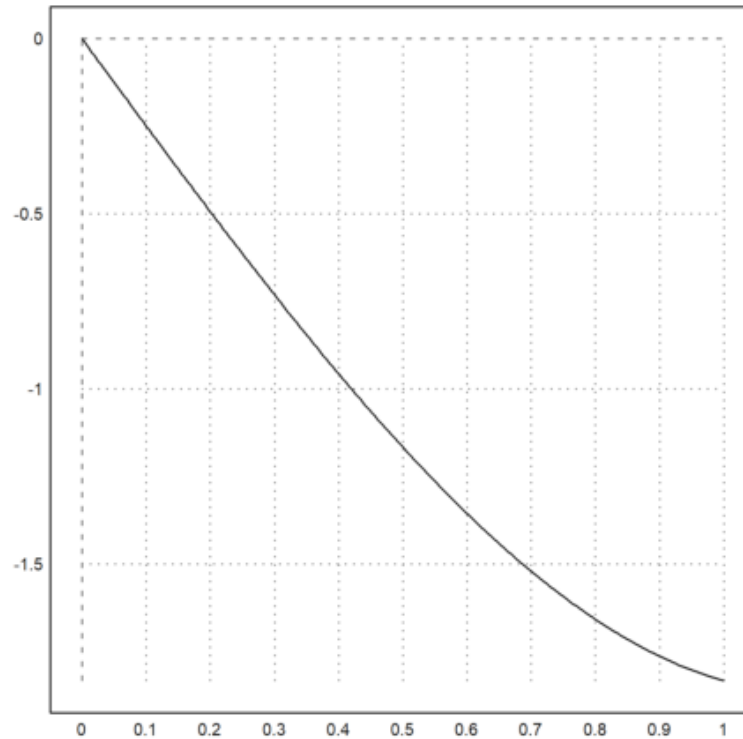


Fungsi dalam satu Parameter

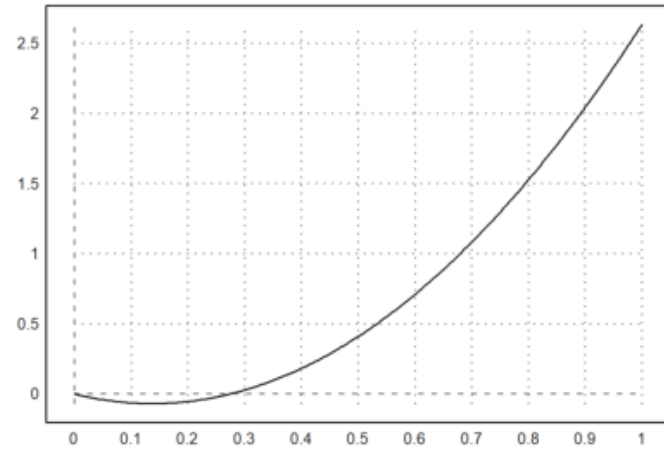
Fungsi plot yang paling penting untuk plot planar adalah `plot2d()`. Fungsi ini diimplementasikan dalam bahasa Euler dalam file “`plot.e`”, yang dimuat pada awal program.

Berikut adalah beberapa contoh penggunaan fungsi. Seperti biasa dalam EMT, fungsi yang bekerja untuk fungsi atau ekspresi lain, Anda dapat mengoper parameter tambahan (selain `x`) yang bukan variabel global ke fungsi dengan parameter titik koma atau dengan koleksi panggilan.

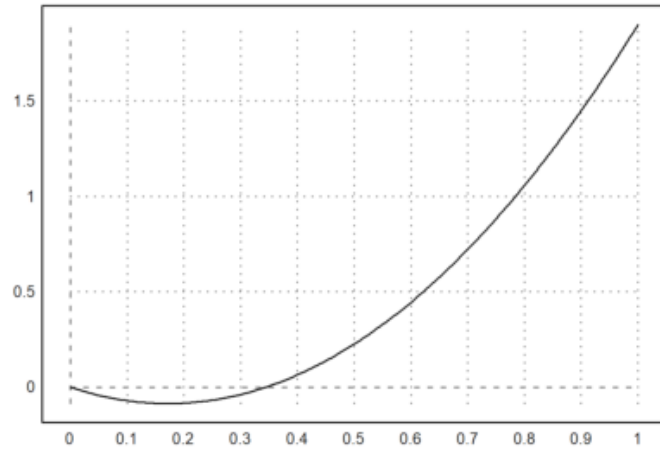
```
>function f(x,b) := x^3/b-b*x-x;  
>b=1.5; plot2d("f",0,1;b):
```

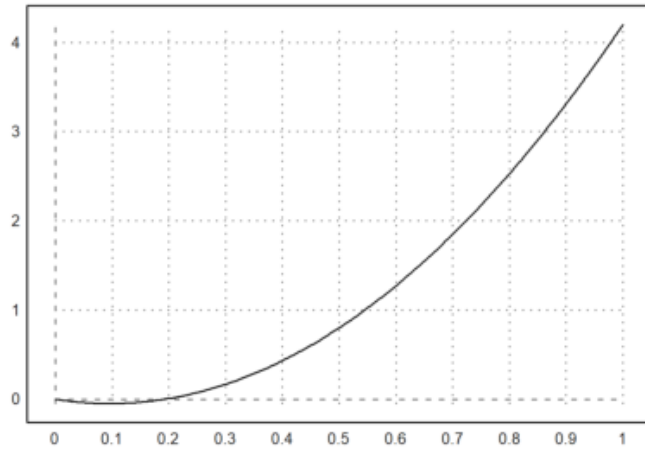
```
>function f(x,a) := x^2/a+a*x^2-x; // mendefinisikan fungsi  
>a=0.3; plot2d("f",0,1;a): // plot dengan a=0.3
```



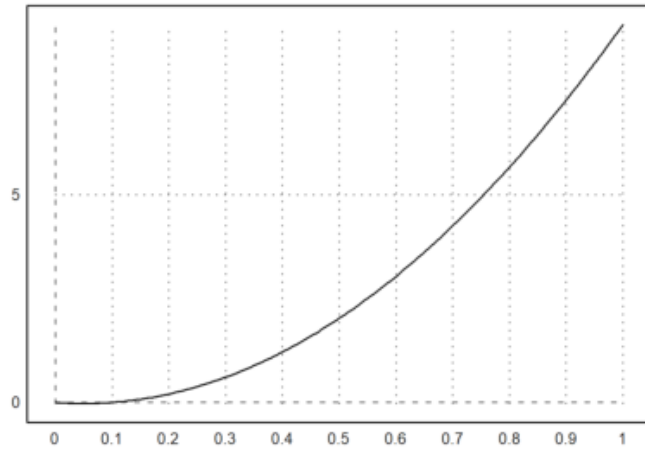
```
>plot2d("f",0,1;0.4): // plot dengan a=0.4
```



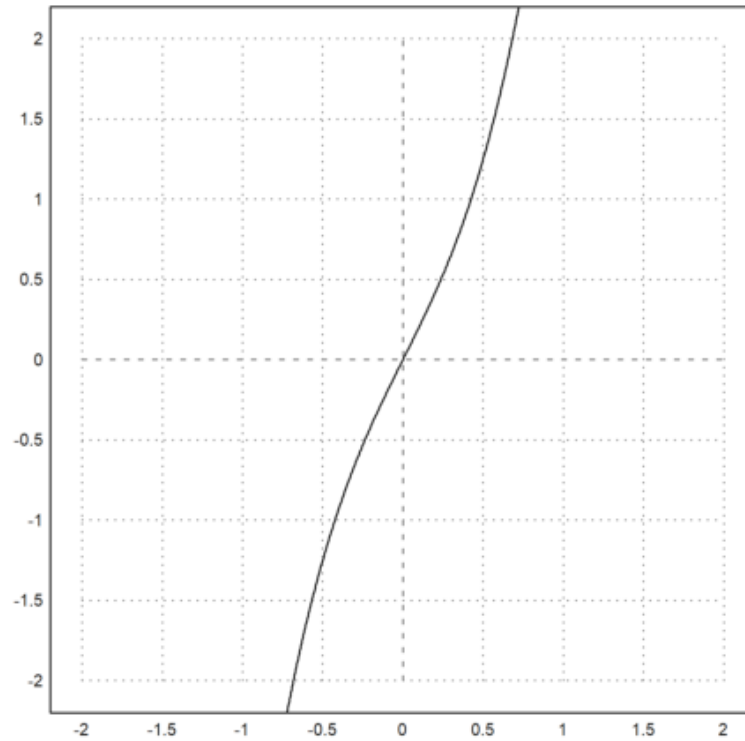
```
>plot2d({"f",0.2},0,1): // plot dengan a=0.2
```



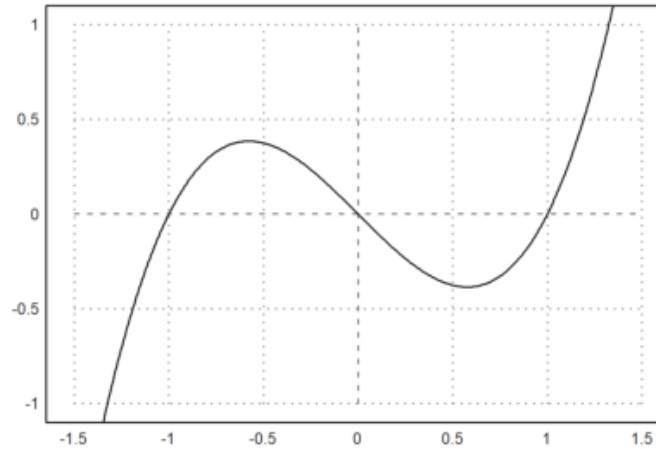
```
>plot2d({"f(x,b)",b=0.1},0,1): // plot dengan 0.1
```



```
>function f(x) := 2*x^3+3*x-x;... // mendefinisikan fungsi f(x)
>plot2d("f",r=2):
```



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



Berikut ini adalah ringkasan dari fungsi yang diterima

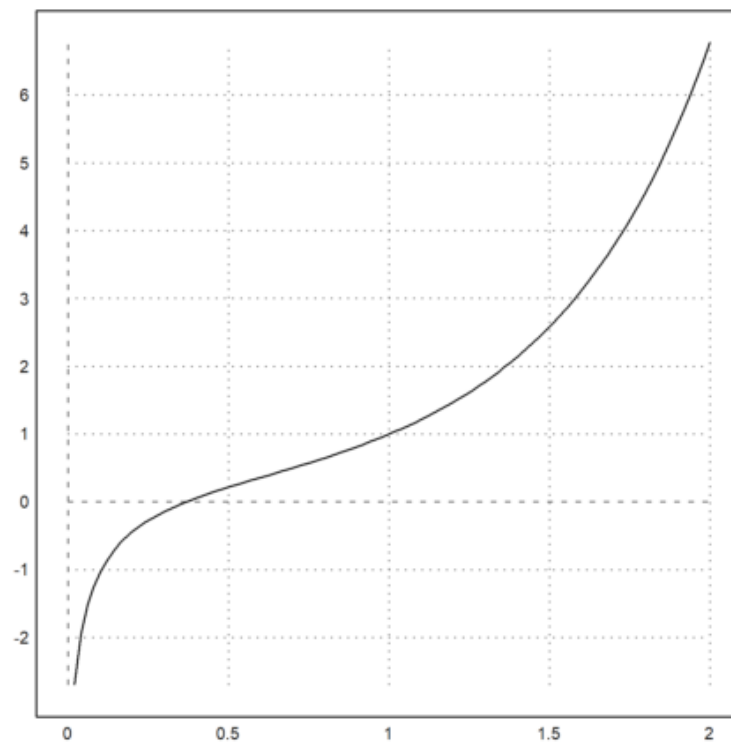
- ekspresi atau ekspresi simbolik dalam x
- fungsi atau fungsi simbolik dengan nama sebagai "f"
- fungsi-fungsi simbolik hanya dengan nama f

Fungsi `plot2d()` juga menerima fungsi simbolik. Untuk fungsi simbolik, nama saja sudah cukup.

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

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

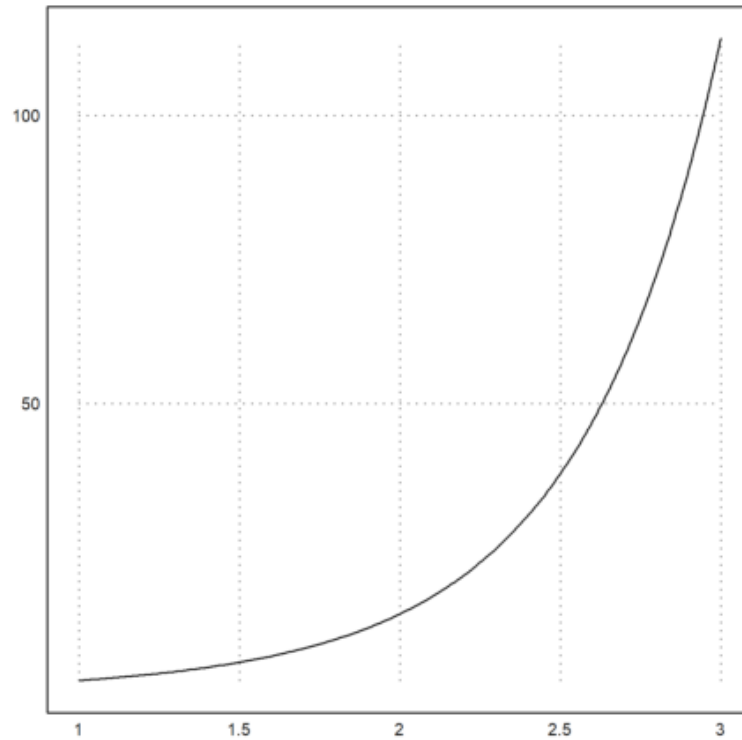
```
>plot2d(f,0,2):
```



```
>function f(x) &= diff(2*x^x,x)
```


$$\frac{x}{2} (\log(x) + 1)$$

```
>plot2d(f,1,3):
```

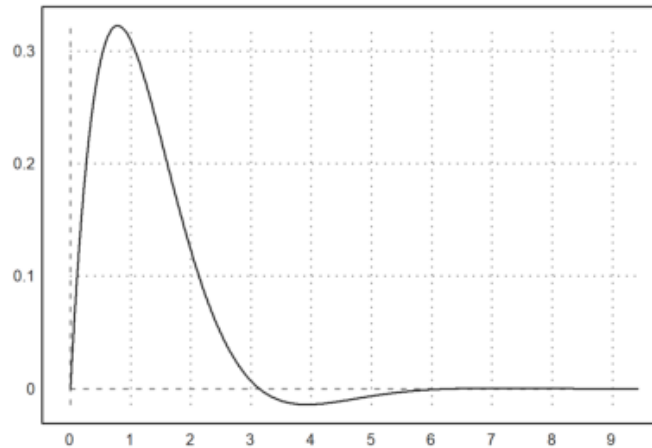


Tentu saja, untuk ekspresi atau ungkapan simbolik, nama variabel sudah cukup untuk memplotnya.

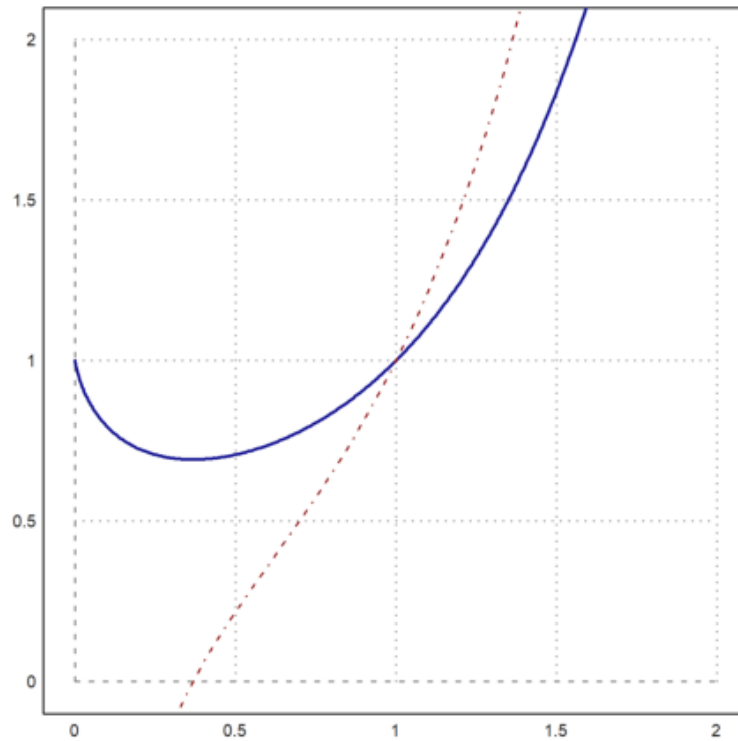
```
>expr &= sin(x)*exp(-x)
```

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

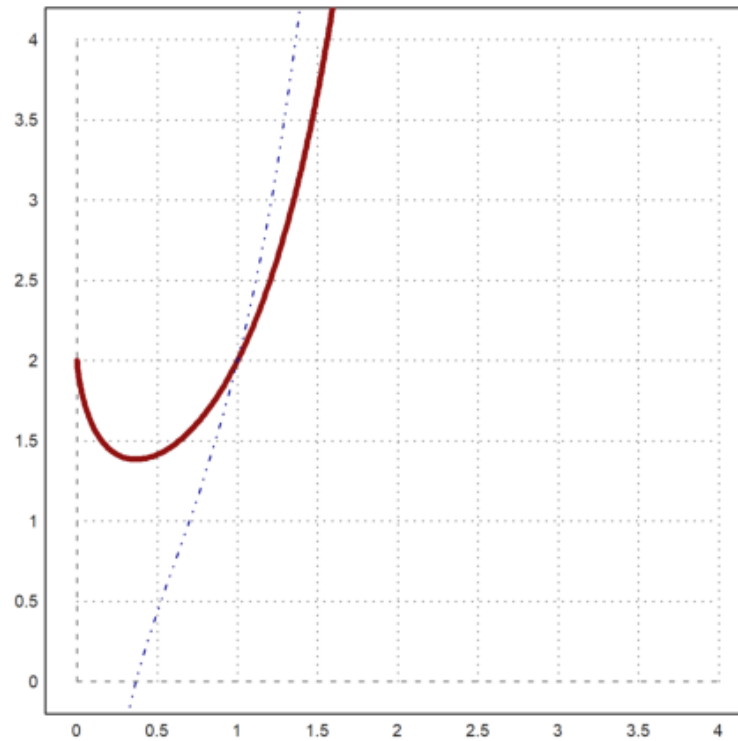
```
>plot2d(expr,0,3pi):
```



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



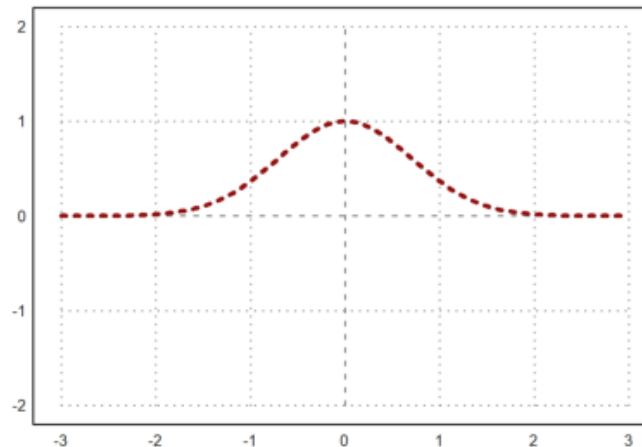
```
>function f(x) &= 2*x^x;  
>plot2d(f,r=2,cx=2,cy=2,color=red,thickness=4);  
>plot2d(&diff(f(x),x),>add,color=blue,style=".-."):
```



- style = "...". Pilih dari "-", "—", "·", "·", "·", "·".
- color: Lihat di bawah untuk warna.
- ketebalan: Default adalah 1.

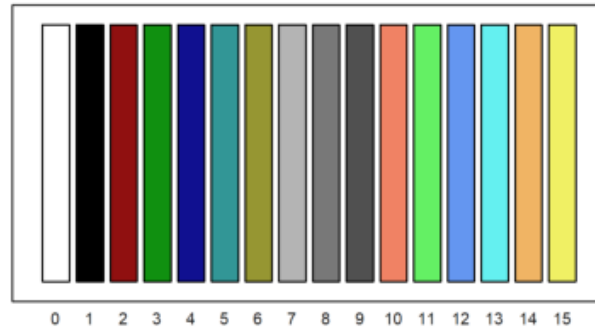
- 0..15: indeks warna default.
- konstanta warna: putih, hitam, merah, hijau, biru, cyan, zaitun, abu-abu muda, abu-abu, abu-abu tua, oranye, hijau muda, pirus, biru muda, oranye muda, kuning
- rgb (merah, hijau, biru): parameter adalah real dalam $[0,1]$.

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



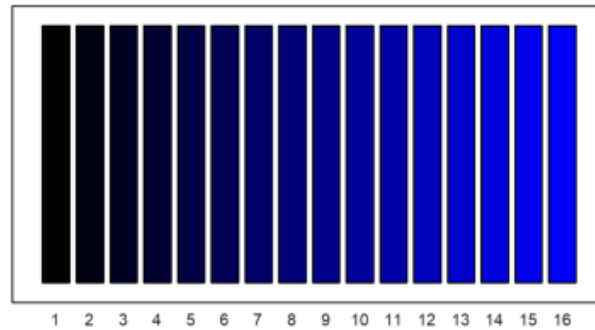
Berikut ini adalah pemandangan warna EMT yang sudah ditetapkan sebelumnya.

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

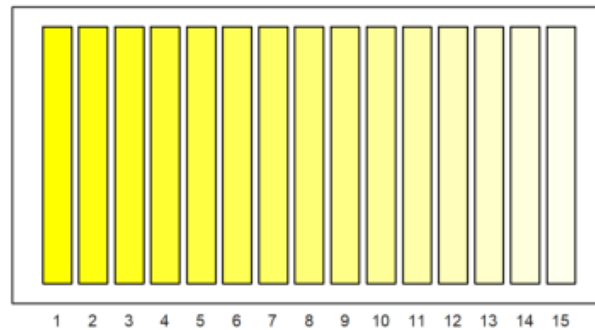


Tapi kamu juga bisa menggunakan warna lain

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



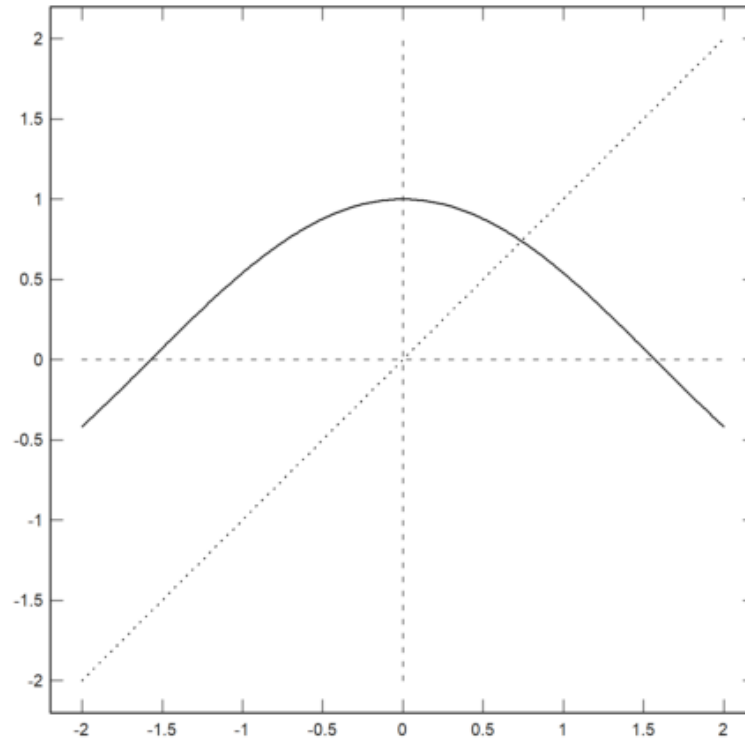
```
>aspect(2); columnsplot(ones(1,15),grid=0,color=rgb(1,1,linspace(0,1,15))):
```



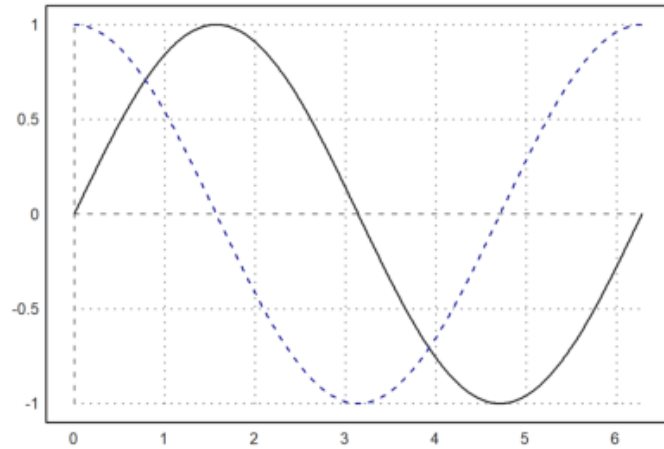
Menggambar Beberapa Kurva pada bidang koordinat yang sama

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

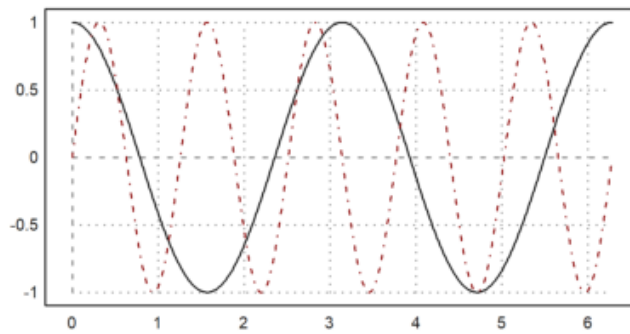
```
>aspect(); plot2d("cos(x)",r=2,grid=6); plot2d("x",style=".",>add):
```

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

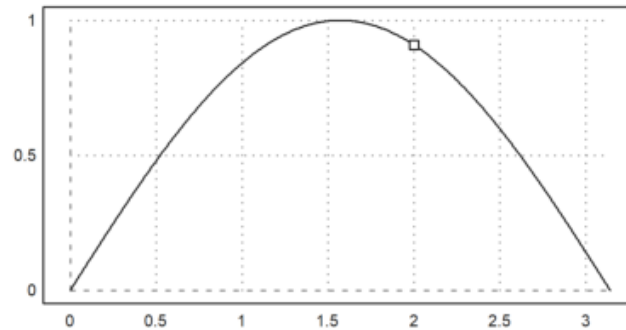


```
>aspect(2); plot2d("cos(2x)",0,2pi); plot2d("sin(5x)",color=red,style="-.-",>add):
```

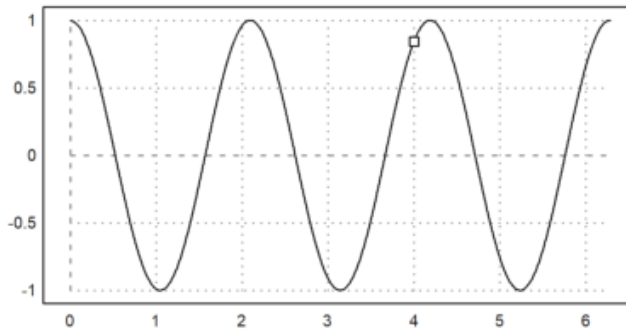


Salah satu kegunaan `>add` adalah untuk menambahkan titik pada kurva.

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

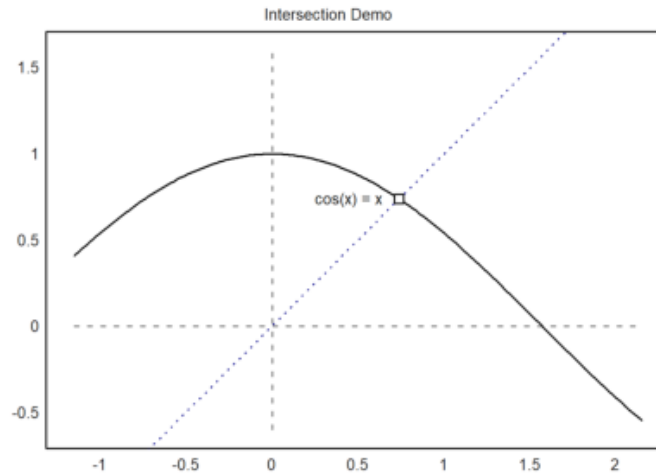


```
>plot2d("cos(3x)",0,2pi); plot2d(4,sin(1),>points,>add): // titik pada (4,sin(1))
```



Kami menambahkan titik perpotongan dengan label (pada posisi “cl” untuk kiri tengah), dan menyisipkan hasilnya ke dalam buku catatan. Kami juga menambahkan judul ke 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 ini, 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. Plot ini dilakukan dalam perintah multi-baris berikut dengan tiga pemanggilan `plot2d()`. Perintah kedua dan ketiga memiliki set flag `>add`, yang membuat plot menggunakan rentang sebelumnya. Kami menambahkan sebuah kotak label yang menjelaskan fungsi-fungsi tersebut.

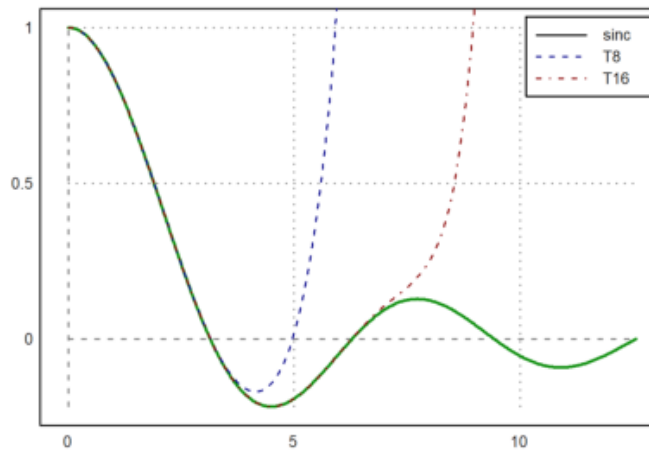
```
>$taylor(sin(x)/x,x,0,4)
```

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

```

>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]):

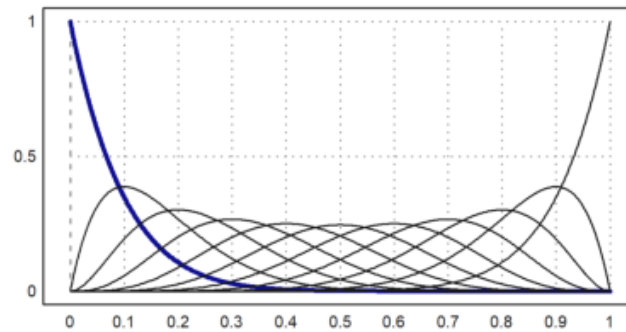
```



Pada contoh berikut, kami menghasilkan Polinomial Bernstein.

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

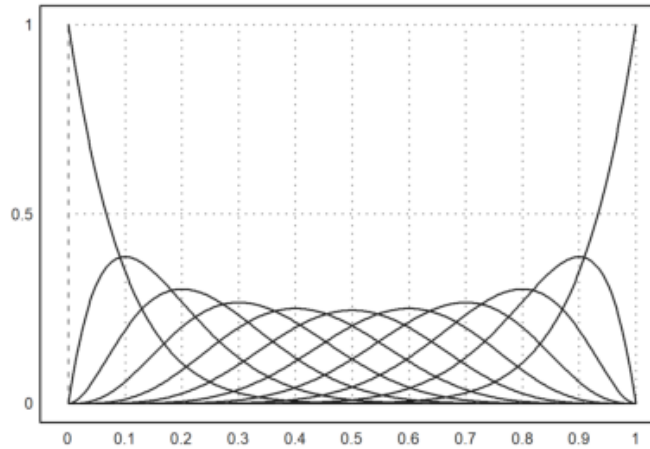
```
>plot2d("(1-x)^10",0,1,color=blue,thickness=3); // plot fungsi pertama
>for i=1 to 10; plot2d("bin(10,i)*x^i*(1-x)^(10-i)",>add); end;
>insimg;
```



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

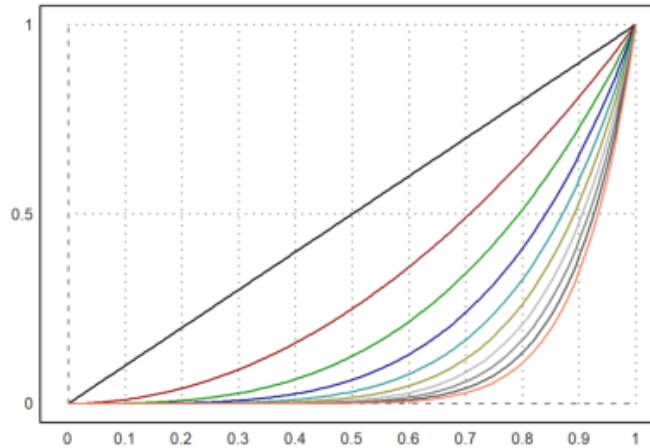
Kita membuat sebuah matriks nilai dengan satu Polinomial Bernstein di setiap baris. Untuk ini, kita cukup menggunakan vektor kolom i . Lihatlah pengantar tentang bahasa matriks untuk mempelajari lebih lanjut.

```
>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):
```



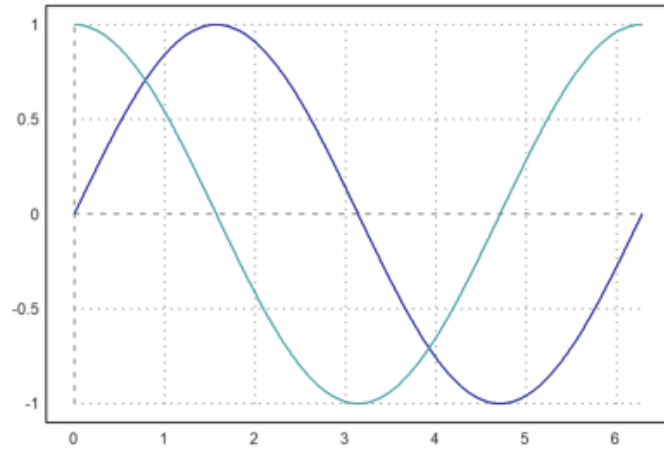
Perhatikan bahwa parameter warna dapat berupa vektor. Kemudian setiap warna digunakan untuk setiap baris matriks.

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

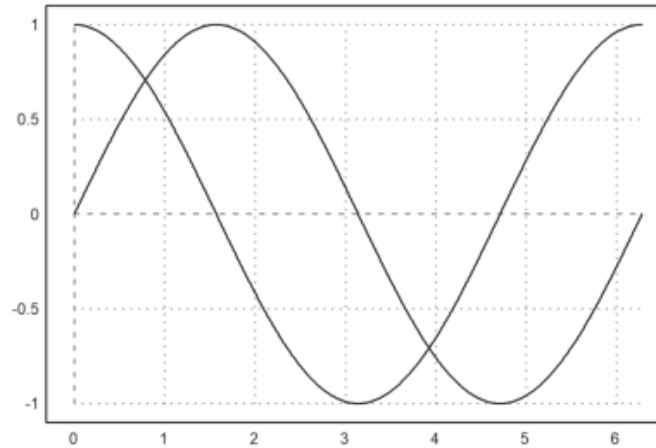



Metode lain adalah menggunakan vektor ekspresi (string). Anda kemudian dapat menggunakan larik warna, larik gaya, dan larik ketebalan dengan panjang yang sama.

```
>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) // membuat list
```

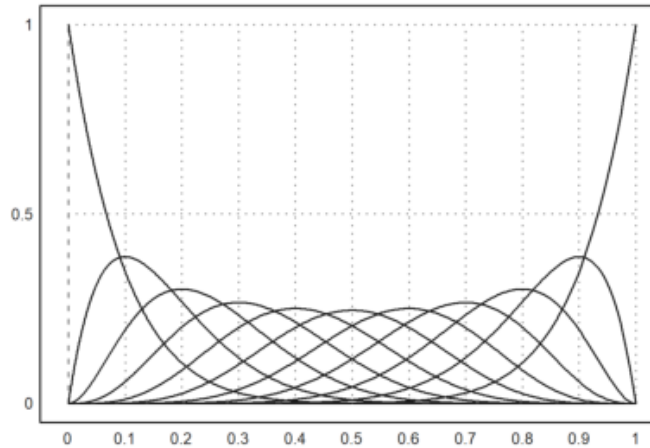
```

      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) // mendapatkan vektor string dari vektor simbolik
```

```
(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
```

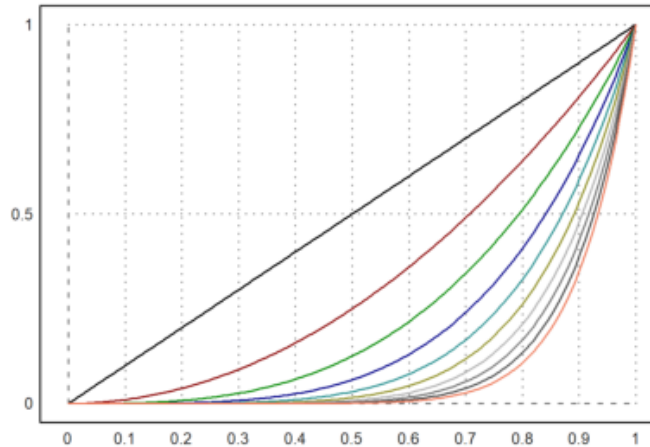


Alternatif lain adalah dengan menggunakan bahasa matriks Euler.

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):
```

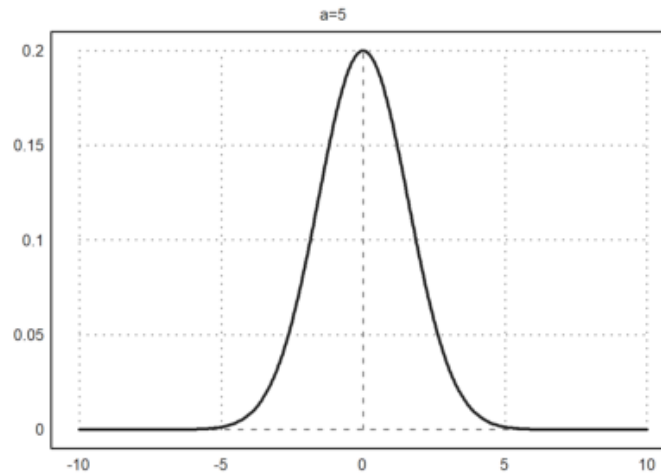


Ekspresi dan fungsi satu baris dapat melihat variabel global.

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

Berhati-hatilah untuk meletakkan semua parameter yang diberikan di akhir perintah plot2d. Pada contoh ini kita mengoper $a=5$ ke fungsi f , yang kita plot dari -10 ke 10.

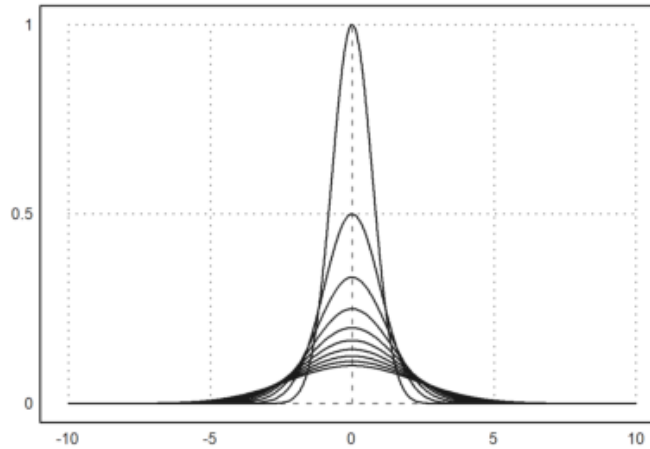
```
>function f(x,a) := 1/a*exp(-x^2/a); ...
>plot2d("f",-10,10;5,thickness=2,title="a=5"):
```



Atau, gunakan koleksi dengan nama fungsi dan semua parameter tambahan. Daftar khusus ini disebut koleksi panggilan, dan itu adalah cara yang lebih disukai untuk mengoper argumen ke fungsi yang dengan sendirinya dioper sebagai argumen ke fungsi lain.

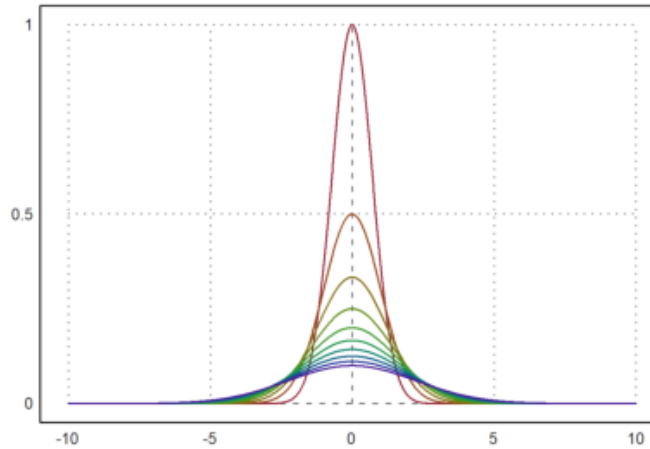
Pada contoh berikut, kita menggunakan perulangan untuk memplot beberapa fungsi (lihat tutorial tentang pemrograman perulangan).

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



Kita dapat mencapai hasil yang sama dengan cara berikut menggunakan bahasa matriks EMT. Setiap baris dari matriks $f(x,a)$ adalah satu fungsi. Selain itu, kita dapat 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)):
```

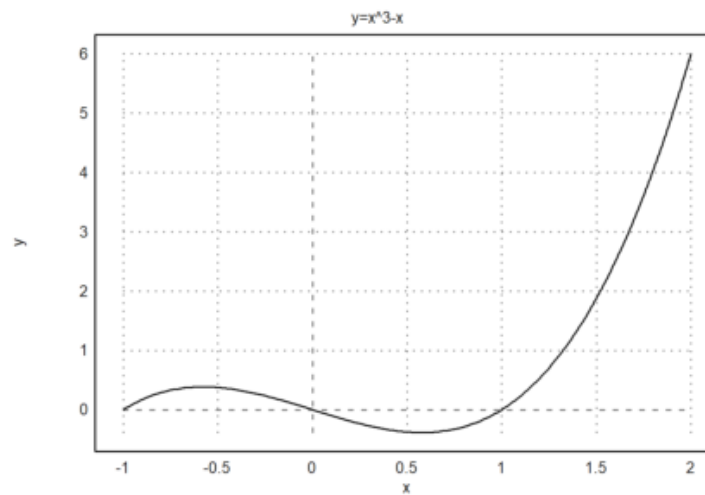
Label Teks

Dekorasi sederhana dapat berupa

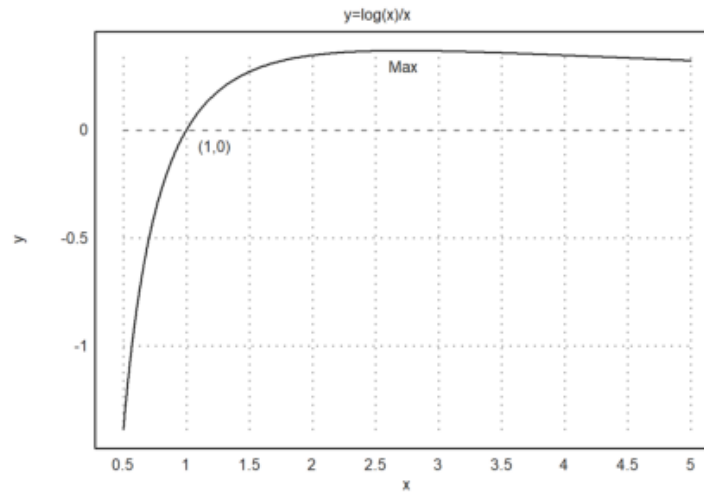
- sebuah judul dengan `title="..."`
- label x dan y dengan `xl="..."`, `yl="..."`
- label teks lain dengan `label("...",x,y)`

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

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

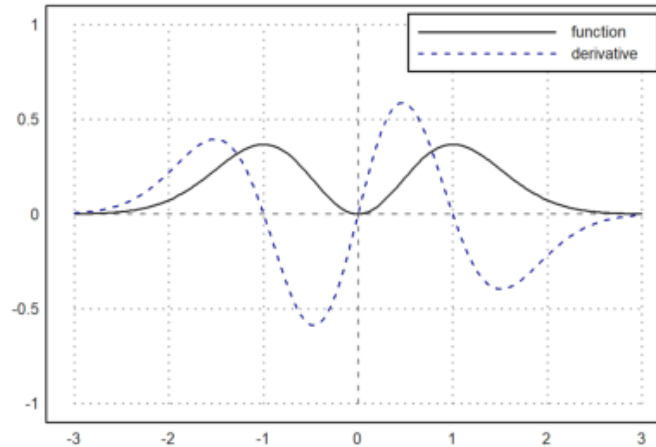


```
>expr := "log(x)/x"; ...  
> plot2d(expr,0.5,5,title="y="+expr,xl="x",yl="y"); ...  
> label("(1,0)",1,0); label("Max",E,expr(E),pos="lc"):
```



Ada juga fungsi `labelbox()`, yang dapat menampilkan fungsi dan teks. Fungsi ini membutuhkan 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):
```

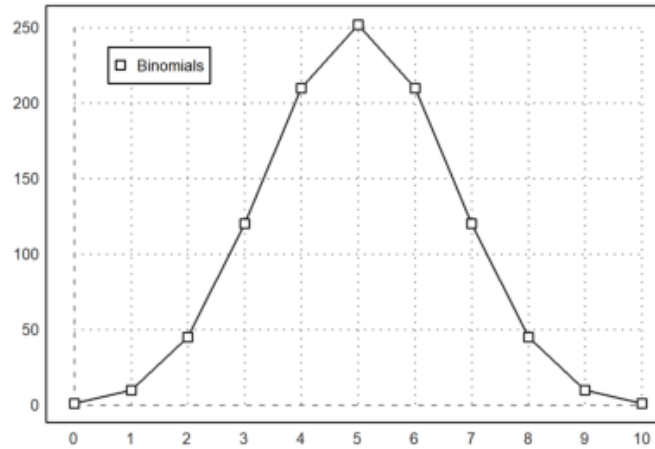


Kotak tersebut berlabuh di kanan atas secara default, tetapi >kiri menambatkannya di kiri atas. Anda dapat memindahkannya ke tempat mana pun yang Anda suka. Posisi jangkar adalah sudut kanan atas kotak, dan angkanya adalah pecahan dari ukuran jendela grafik. Lebarinya otomatis.

Untuk plot titik, kotak label juga dapat digunakan. Tambahkan parameter >titik, atau vektor bendera, satu untuk setiap label.

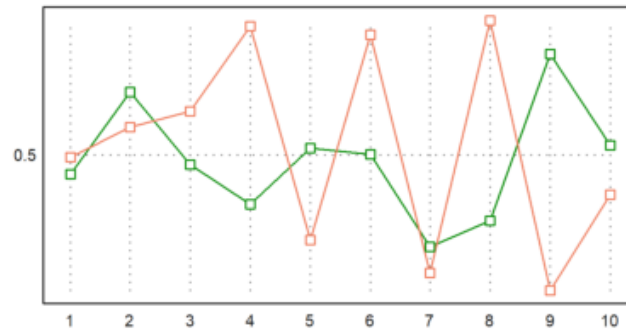
Pada contoh berikut, hanya ada satu fungsi. Jadi kita dapat menggunakan string sebagai pengganti vektor string. Kita mengatur warna teks menjadi hitam untuk contoh ini.

```
>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 pada `plot2d()` warna dapat diatur untuk setiap baris plot. Terdapat lebih banyak plot khusus untuk keperluan statistik (lihat tutorial tentang statistik).

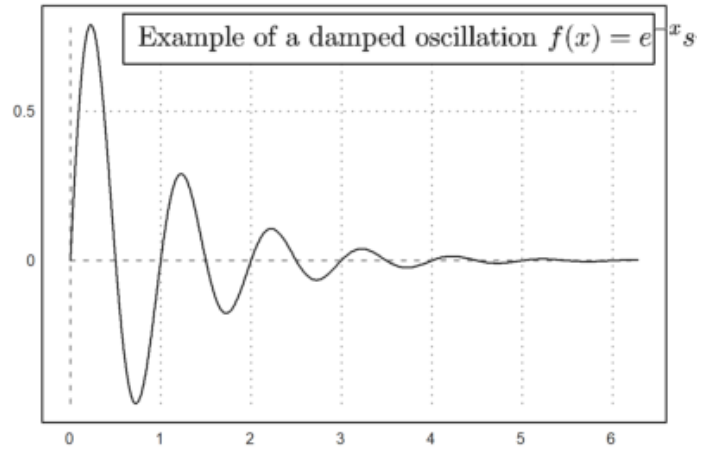
```
>statplot(1:10,random(2,10),color=[green,orange]):
```



Fitur yang serupa adalah fungsi `textbox()`.

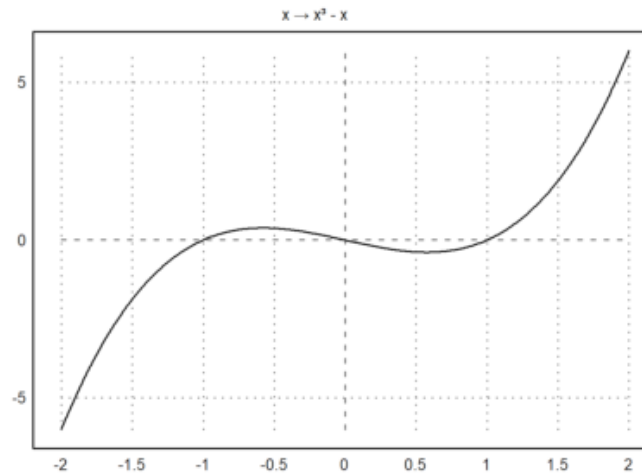
Lebar nya secara default adalah lebar maksimal baris teks. Tetapi bisa juga diatur oleh pengguna.

```
>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):
```



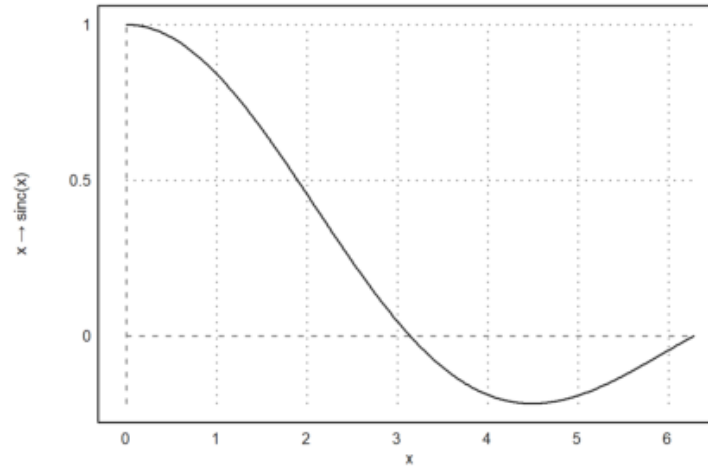
Label teks, judul, kotak label, dan teks lainnya dapat berisi string Unicode (lihat sintaksis EMT untuk informasi lebih lanjut tentang string Unicode).

```
>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=u"x &rarr; sinc(x)",>vertical):
```

LaTeX

Anda juga dapat memplot formula LaTeX jika Anda telah menginstal sistem LaTeX. Saya merekomendasikan MiKTeX. Jalur ke binari “latex” dan “dvi2png” harus berada di jalur sistem, atau Anda harus mengatur LaTeX di menu opsi.

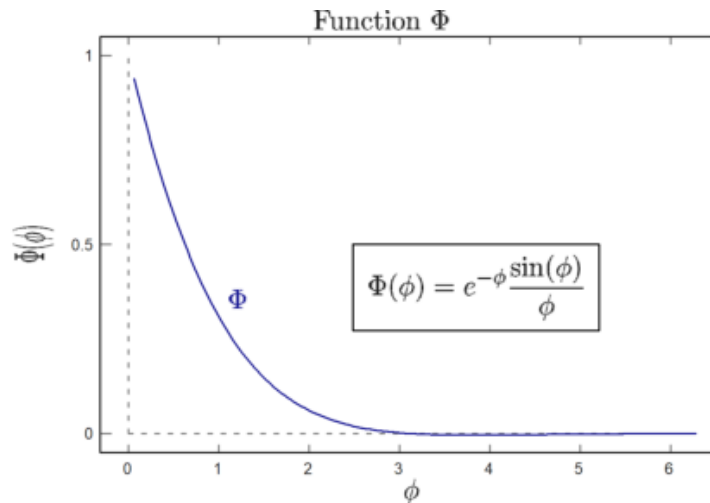
Perhatikan, bahwa penguraian LaTeX berjalan lambat. Jika Anda ingin menggunakan LaTeX dalam plot animasi, Anda harus memanggil `latex()` sebelum perulangan satu kali dan menggunakan hasilnya (gambar dalam matriks RGB).

Pada plot berikut ini, kita menggunakan LaTeX untuk label x dan y , sebuah 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):

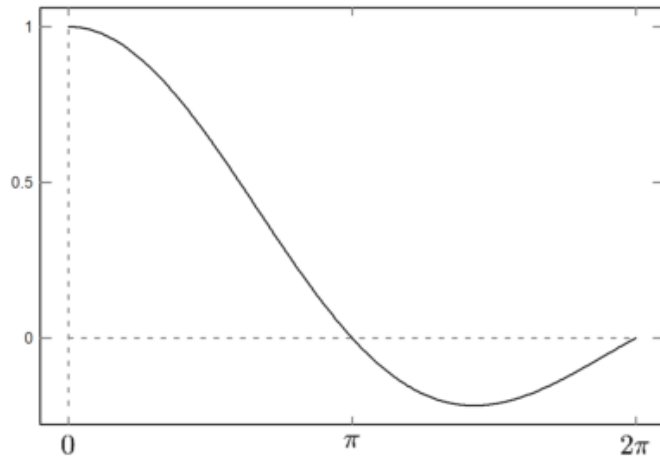
```



Seringkali, kita menginginkan spasi dan label teks yang tidak sesuai pada sumbu x. Kita dapat menggunakan `xaxis()` dan `yaxis()` seperti yang akan kita tunjukkan nanti.

Cara termudah adalah dengan membuat plot kosong dengan sebuah frame menggunakan `grid=4`, dan kemudian menambahkan grid dengan `ygrid()` dan `xgrid()`. Pada contoh berikut, kita menggunakan tiga buah 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.

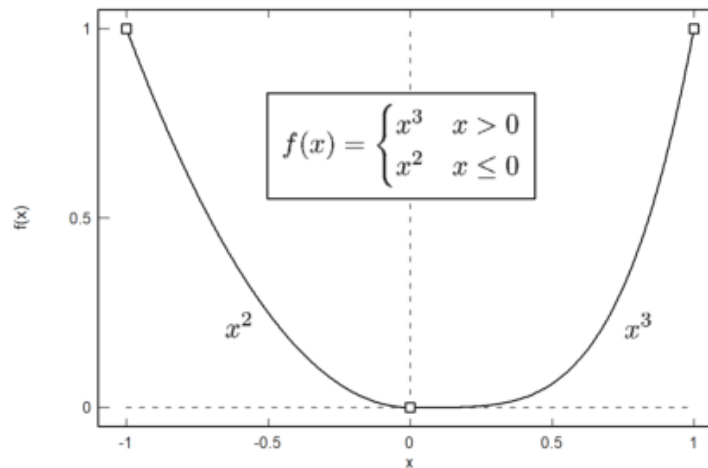
```
>function map f(x) ...
```

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

Parameter “map” membantu menggunakan fungsi untuk vektor. Untuk plot, hal ini tidak diperlukan. Tetapi untuk mendemonstrasikan bahwa vektorisasi berguna, kami menambahkan beberapa titik kunci pada plot pada $x=-1$, $x=0$ 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 Anda telah instal 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):
```



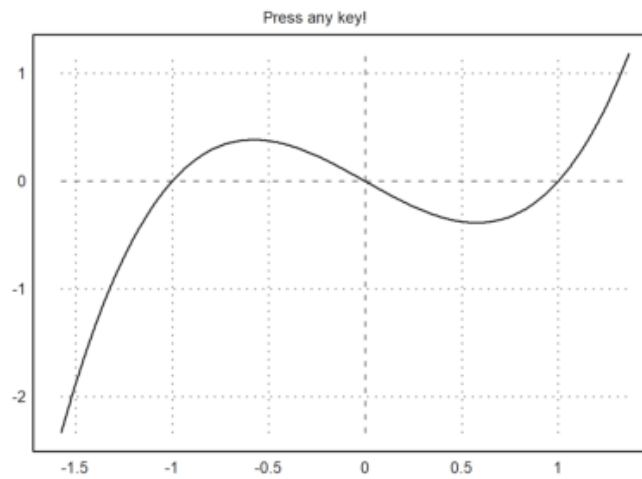
Ketika memplot fungsi atau ekspresi, parameter `>user` memungkinkan pengguna untuk memperbesar dan menggeser plot dengan tombol kursor atau mouse. Pengguna dapat

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

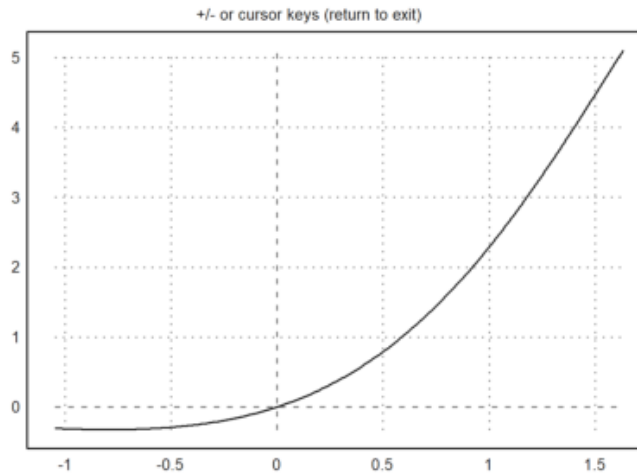
Tombol spasi akan mengatur ulang plot ke jendela plot awal.

Ketika memplot data, bendera `>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)":
```



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

Fungsi bawaan `mousedrag()` menunggu peristiwa mouse atau keyboard. Fungsi ini melaporkan mouse ke bawah, mouse bergerak atau mouse ke atas, dan penekanan tombol. Fungsi `dragpoints()` memanfaatkan hal ini, dan mengizinkan pengguna untuk menyeret titik manapun di dalam plot.

Kita membutuhkan fungsi plot terlebih dahulu. Sebagai contoh, kita melakukan interpolasi pada 5 titik dengan sebuah 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
```

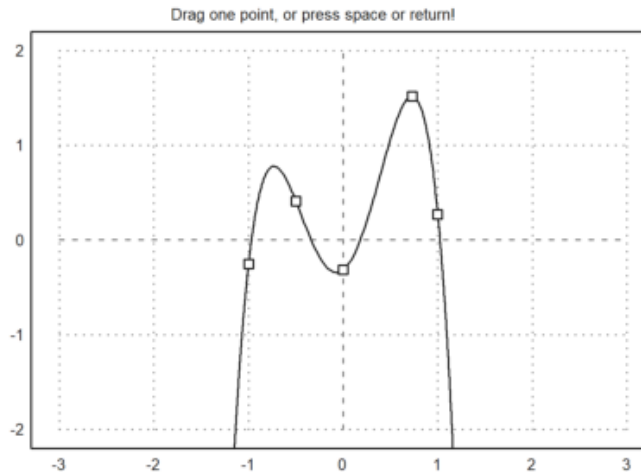
```

    plot2d(xp[select],yp[select],color=red,>points,>add);
endif;
title("Drag one point, or press space or return!");
endfunction

```

Perhatikan parameter titik koma pada plot2d (d dan xp), yang diteruskan ke evaluasi fungsi interp(). Tanpa ini, kita harus menulis fungsi plotinterp() terlebih dahulu, untuk mengakses 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 tergantung pada vektor parameter, dan memungkinkan pengguna menyesuaikan parameter ini.

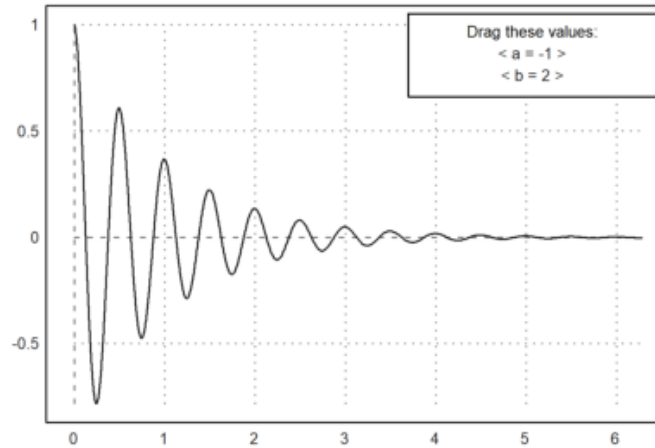
Pertama, kita memerlukan fungsi plot.

```
>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, dan secara opsional, sebuah garis judul.

Terdapat slider interaktif, yang dapat mengatur nilai oleh pengguna. Fungsi dragvalues() menyediakan ini.

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



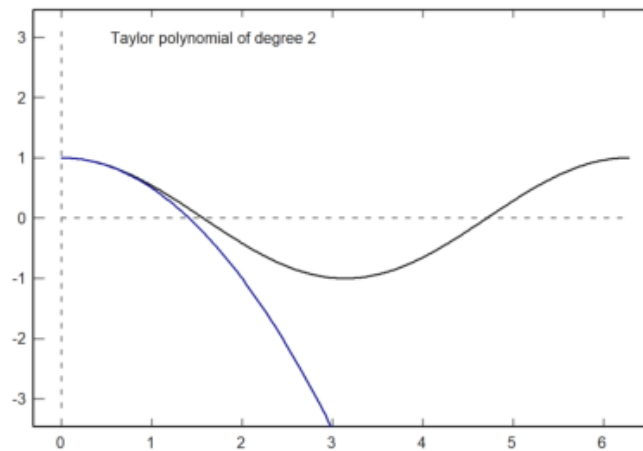
Anda dapat membatasi nilai yang diseret menjadi bilangan bulat. Sebagai contoh, kita menulis fungsi plot, yang memplot polinomial Taylor dengan derajat n ke fungsi kosinus.

```
>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 sampai 20 dalam 20 stop. Hasil dari `dragvalues()` digunakan untuk memplot sketsa dengan n ini, dan untuk menyisipkan 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 peragaan sederhana dari fungsi ini. Pengguna dapat menggambar di atas jendela plot, meninggalkan jejak titik.

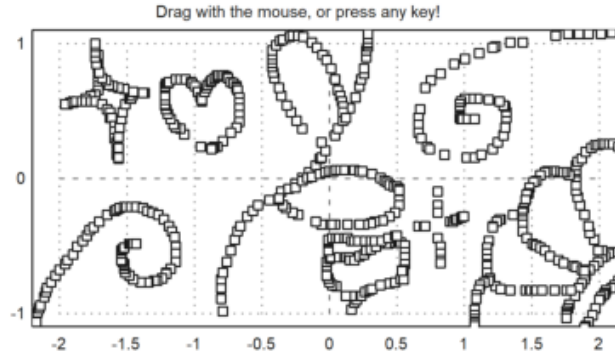
```
>function dragtest ...
```

```

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

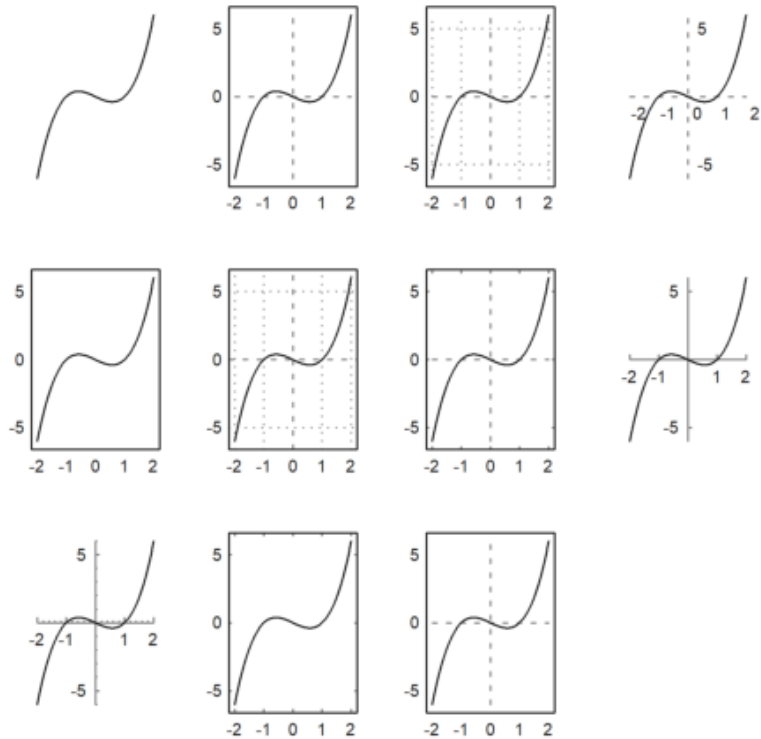
```

>dragtest: // lihat hasilnya dan cobalah lakukan!



Secara default, EMT menghitung tanda sumbu otomatis dan menambahkan label pada setiap tanda. Hal ini dapat diubah dengan parameter grid. Gaya default sumbu dan label dapat dimodifikasi. 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):
```



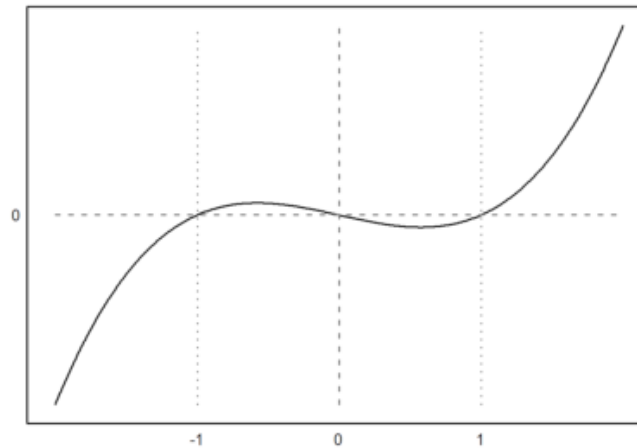
Parameter `<frame` mematikan bingkai, dan `framecolor=blue` menetapkan bingkai ke warna biru.

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

```

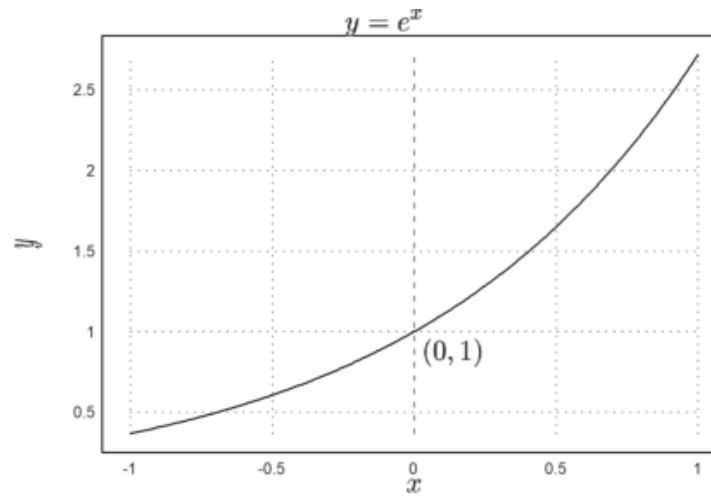


Untuk judul plot dan label sumbu, lihat contoh berikut.

```

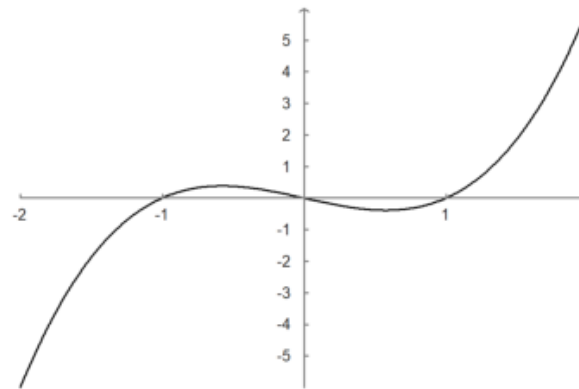
>plot2d("exp(x)",-1,1);
>textcolor(black); // set the text color to black
>title(latex("y=e^x")); // title above the plot
>xlabel(latex("x")); // "x" for x-axis
>ylabel(latex("y"),>vertical); // vertical "y" for y-axis
>label(latex("(0,1)"),0,1,color=blue): // label a point

```



Sumbu dapat digambar secara terpisah dengan `xaxis()` dan `yaxis()`.

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

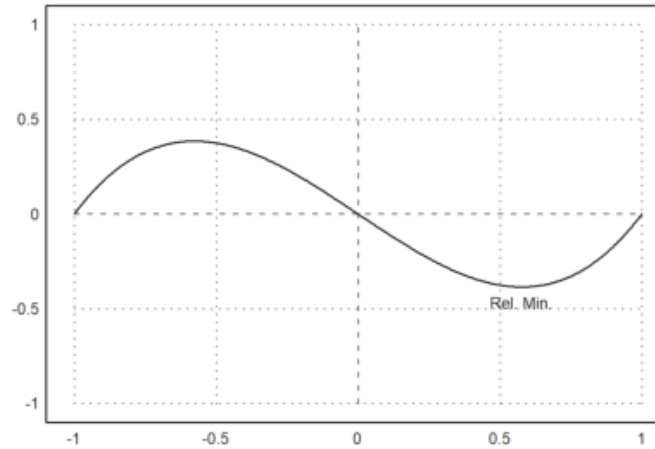



Teks pada plot dapat diatur dengan `label()`. Dalam contoh berikut, "lc" berarti tengah bawah. Ini mengatur posisi label relatif terhadap koordinat plot.

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

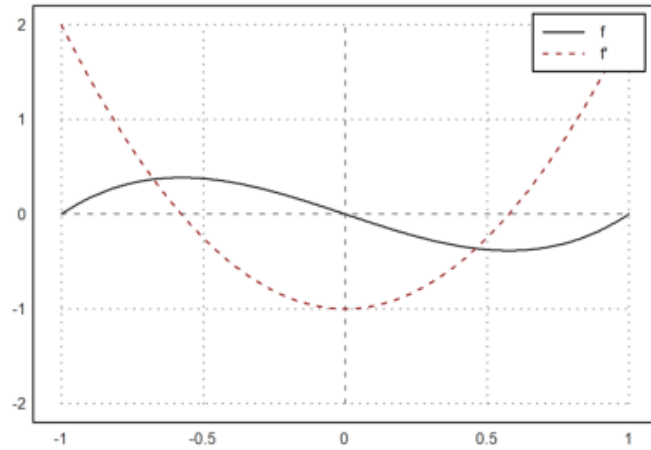
$$x^3 - x$$

```
>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
```

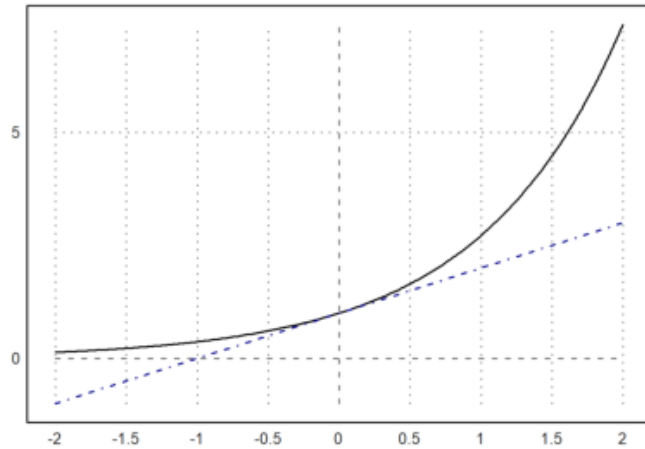


Ada juga kotak teks.

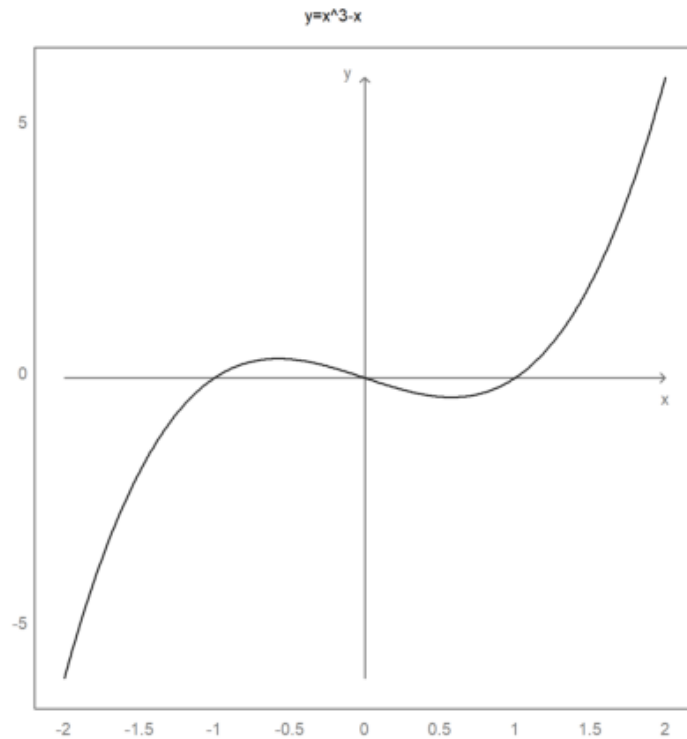
```
>plot2d(&f(x),-1,1,-2,2); // function  
>plot2d(&diff(f(x),x),>add,style="--",color=red); // derivative  
>labelbox(["f","f'"],["-","--"],[black,red]): // label box
```



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



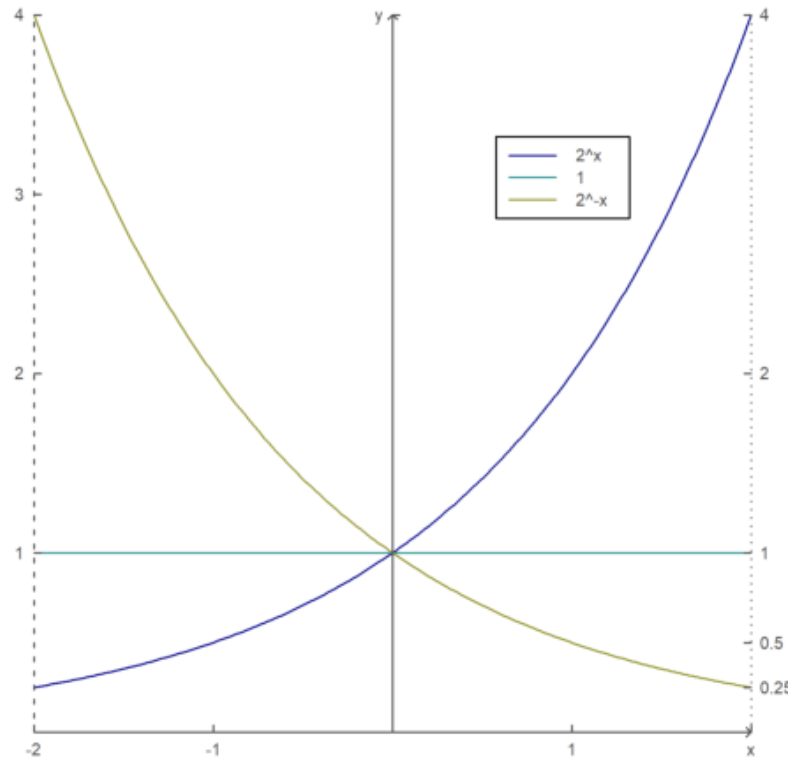
```
> gridstyle("->", color=gray, textcolor=gray, framecolor=gray); ...  
> plot2d("x^3-x", grid=1); ...  
> setttitle("y=x^3-x", color=black); ...  
> label("x", 2, 0, pos="bc", color=gray); ...  
> label("y", 0, 6, pos="cl", color=gray); ...  
> reset():
```



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

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

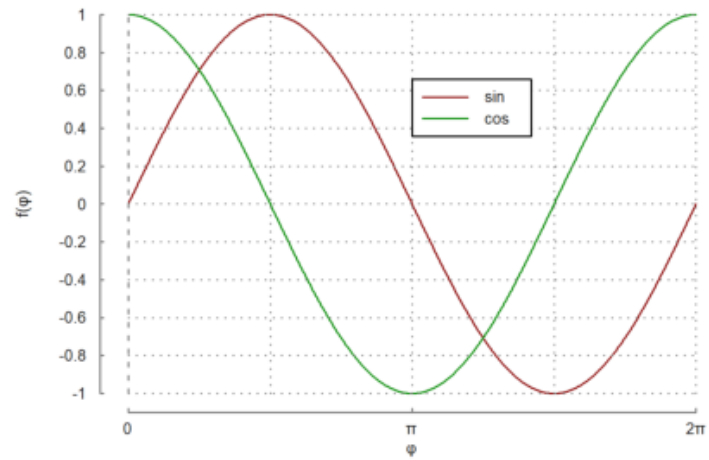
```
>fullwindow; ...  
> gridstyle(color=darkgray,textcolor=darkgray); ...  
> plot2d(["2^x","1","2^(-x)"],a=-2,b=2,c=0,d=4,<grid,color=4:6,<frame); ...  
> xaxis(0,-2:1,style="->"); xaxis(0,2,"x",<axis); ...  
> yaxis(0,4,"y",style="->"); ...  
> yaxis(-2,1:4,>left); ...  
> yaxis(2,2^(-2:2),style=".",<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 berada 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"&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); ...  
> xlabel(u"&phi;"); ylabel(u"f(&phi;)"):
```



Memplot 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 diatur untuk mempertahankan rasio aspek persegi.

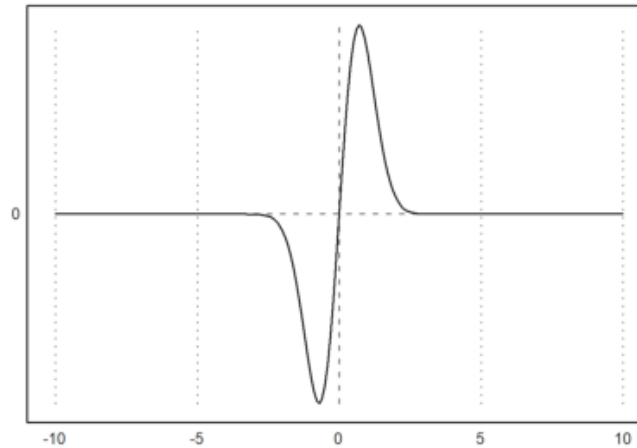
Memplot ekspresi hanyalah singkatan untuk plot data. Untuk plot data, Anda memerlukan satu atau beberapa baris nilai x , dan satu atau beberapa baris nilai y . Dari rentang dan nilai x , fungsi `plot2d` akan menghitung data untuk diplot, secara default dengan evaluasi adaptif dari fungsi tersebut. Untuk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

Namun 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):
```



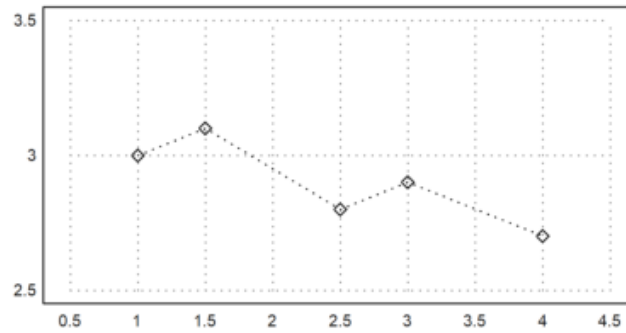
Data juga dapat diplot sebagai titik. Gunakan `poin=true` untuk ini. Plot ini bekerja seperti poligon, namun hanya menggambar sudut-sudutnya saja.

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

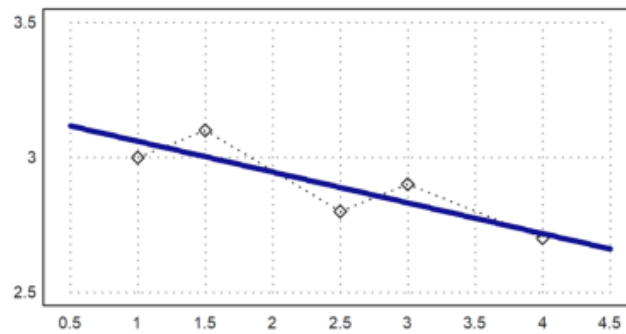
Untuk memplot kumpulan titik, gunakan `>titik`. Jika warna adalah sebuah vektor warna, setiap titik mendapatkan warna yang berbeda. Untuk sebuah matriks koordinat dan vektor kolom, warna berlaku pada baris-baris matriks.

Parameter `>addpoints` menambahkan titik-titik pada 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="."); // garis
>plot2d(xdata,ydata,>points,>add,style="<>"): // menambahkan point
```



```
>p=polyfit(xdata,ydata,1); // get regression line  
>plot2d("polyval(p,x)",>add,color=blue,thickness=4): // add plot of line
```



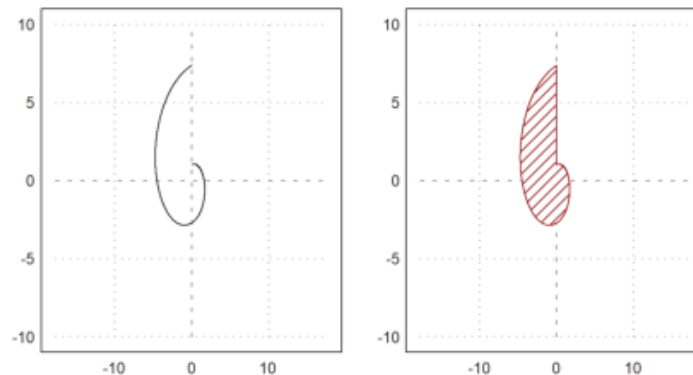
Menggambar Daerah Yang Dibatasi Kurva

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.

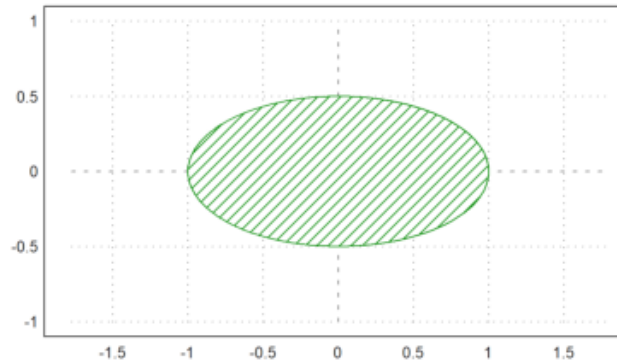
Warna isian ditentukan oleh argumen “fillcolor”, dan pada pilihan <outline mencegah menggambar batas untuk semua gaya kecuali gaya default.

```
>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):
```

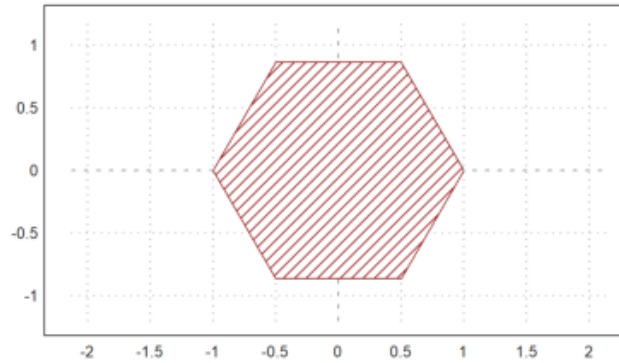


Pada contoh berikut ini, kami memplot elips terisi dan dua segi enam terisi menggunakan kurva tertutup dengan 6 titik dengan gaya isian yang berbeda.

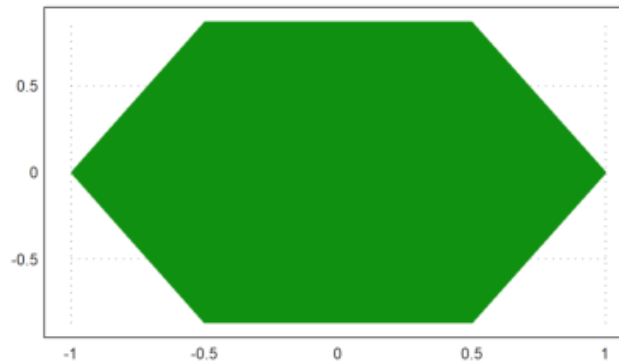
```
>x=linspace(0,2pi,1000); plot2d(sin(x),cos(x)*0.5,r=1,>filled,style="/"):
```



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

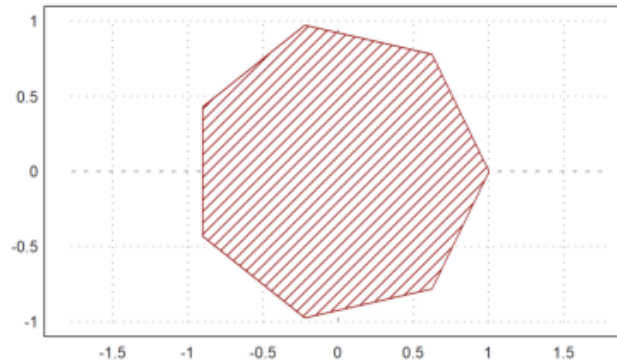


```
>t=linspace(0,2pi,6); plot2d(cos(t),sin(t),>filled,style="#"):
```



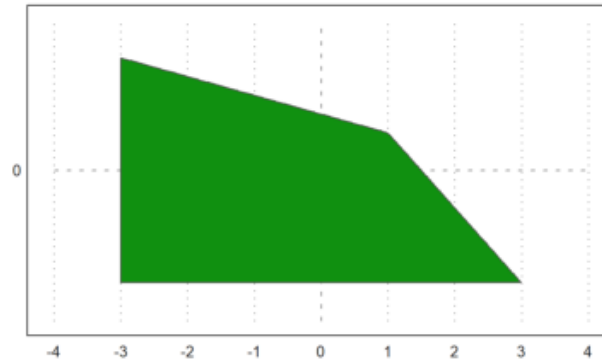
Contoh lain adalah septagon, yang kita buat dengan 7 titik pada lingkaran satuan.

```
>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 bahwa bahasa ini memungkinkan untuk menghasilkan tabel fungsi dengan mudah.

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

Kita sekarang memiliki vektor nilai x dan y. `plot2d()` dapat memplot nilai-nilai ini sebagai sebuah kurva yang menghubungkan titik-titik. Plot dapat diisi. Dalam kasus ini ini memberikan hasil yang bagus karena aturan penggulungan, yang digunakan untuk pengisian.

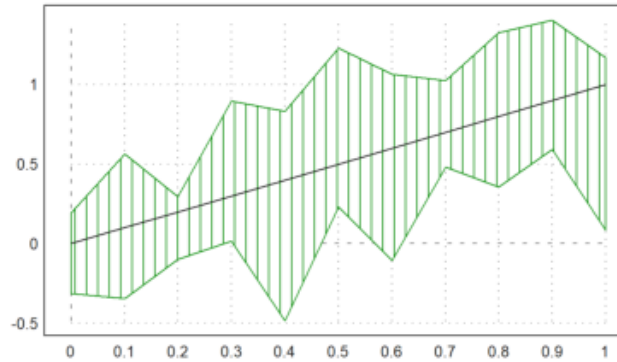
```
>plot2d(x,y,<grid,<frame,>filled):
```




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

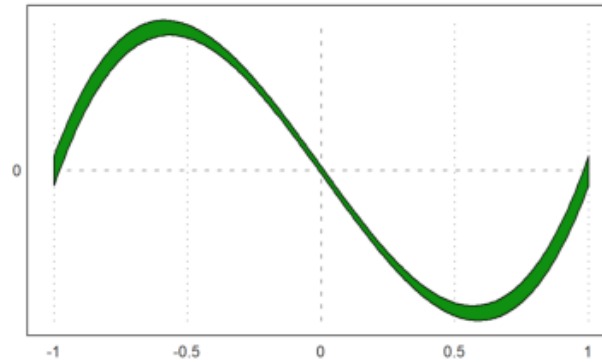
Hal ini dapat berguna untuk memplot kesalahan perhitungan. Tapi itu bisa juga dapat digunakan untuk memplot kesalahan statistik.

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



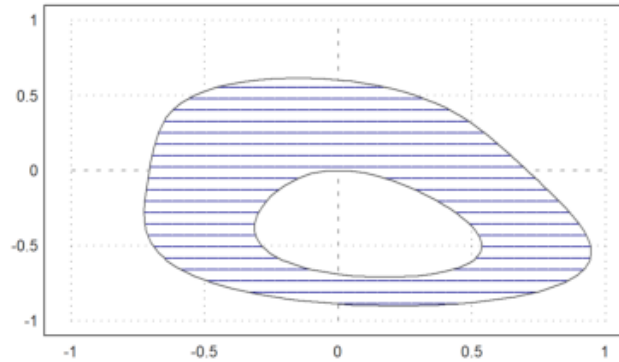
Jika x adalah vektor yang diurutkan, dan y adalah vektor interval, maka `plot2d` akan memplot rentang interval yang terisi pada bidang, gaya isian sama dengan gaya poligon.

```
>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 $2 \times n$. Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

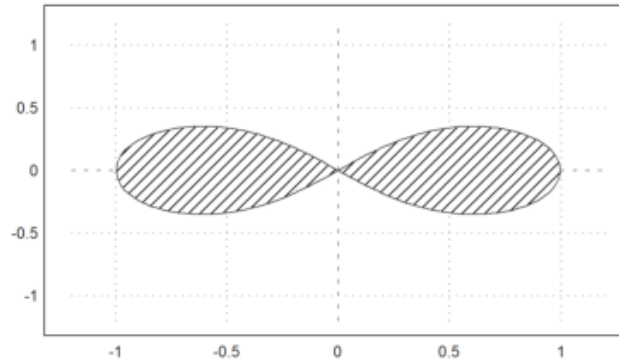
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=[0;1],style="-",color=blue): // 0 <= f(x,y) <= 1
```



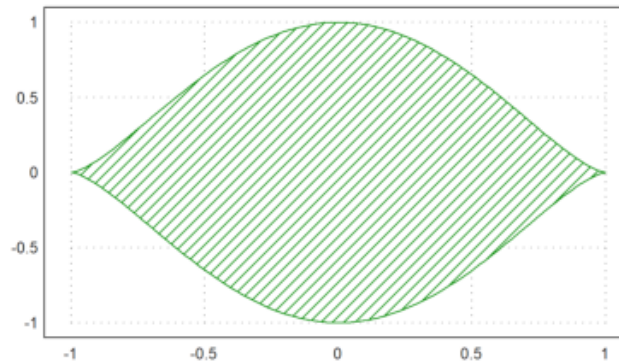
Kita juga dapat mengisi rentang nilai seperti

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

```
>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

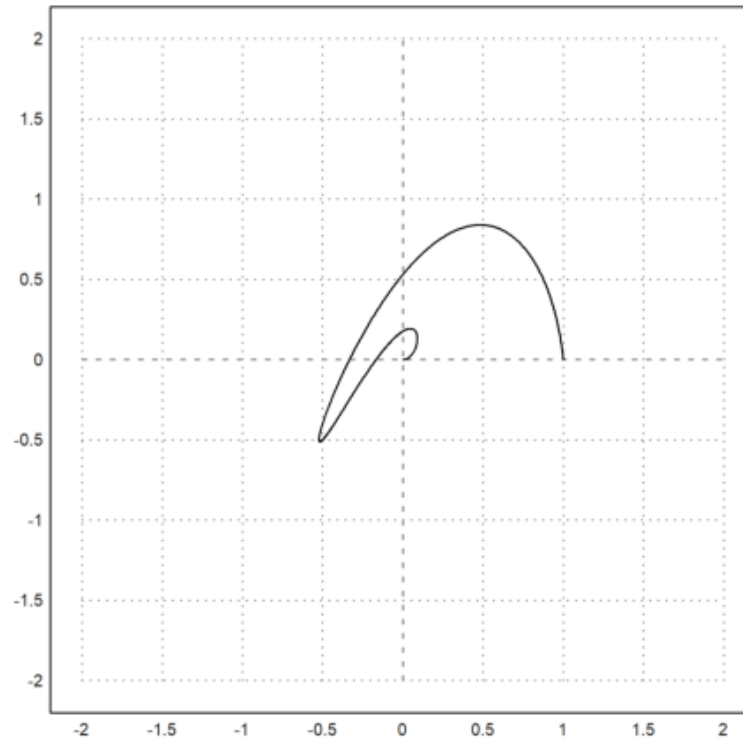
Nilai x tidak perlu diurutkan. (x,y) hanya menggambarkan sebuah kurva. Jika x diurutkan, kurva tersebut adalah grafik fungsi.

Pada contoh berikut, kita memplot spiral

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

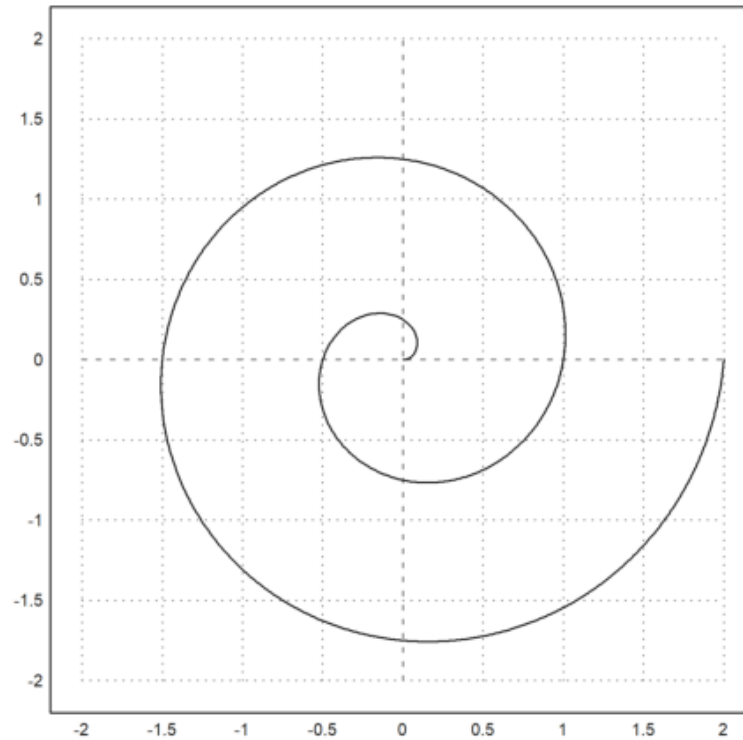
Kita mungkin perlu menggunakan sangat banyak titik untuk tampilan yang halus atau fungsi `adaptive()` untuk mengevaluasi ekspresi (lihat fungsi `adaptive()` untuk lebih jelasnya).

```
>t=linspace(0,1,1000); ...  
>plot2d(t*cos(2*pi*t),t*sin(3*pi*t),r=2):
```

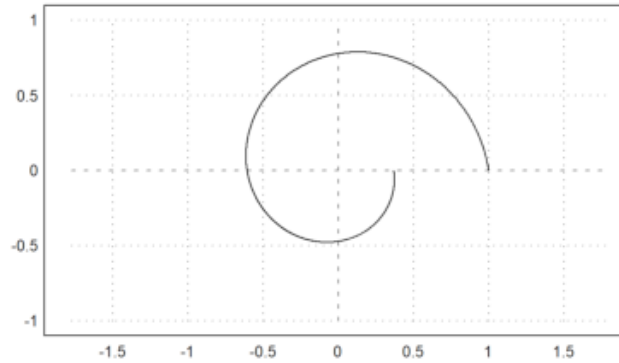


Sebagai alternatif, Anda dapat menggunakan dua ekspresi untuk kurva. Berikut ini adalah plot kurva yang sama seperti di atas.

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



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

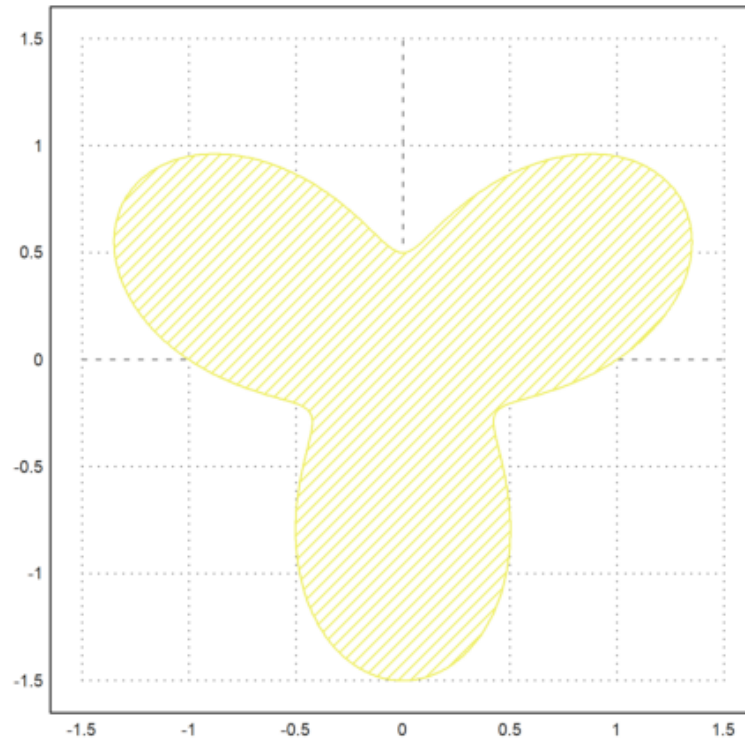
Dalam contoh berikut, kami memplot kurva

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

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=yellow,style="/",r=1.5):
```



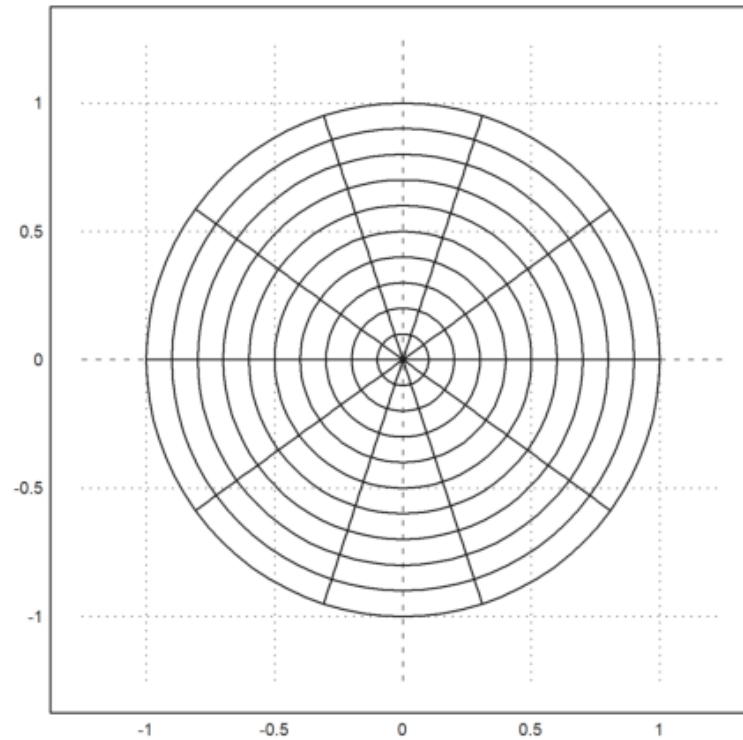
Menggambar Grafik Bilangan Kompleks

Sebuah deretan bilangan kompleks juga dapat diplot. Kemudian titik-titik kisi akan dihubungkan. Jika sejumlah garis kisi ditentukan (atau vektor 1x2 garis kisi) pada argumen `cgrid`, hanya garis-garis kisi tersebut yang akan terlihat.

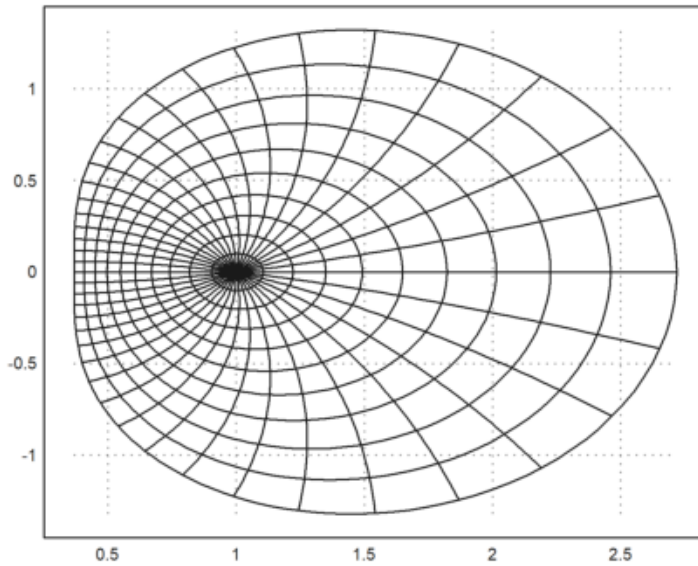
Matriks bilangan kompleks akan secara otomatis diplot sebagai sebuah grid pada bidang kompleks.

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

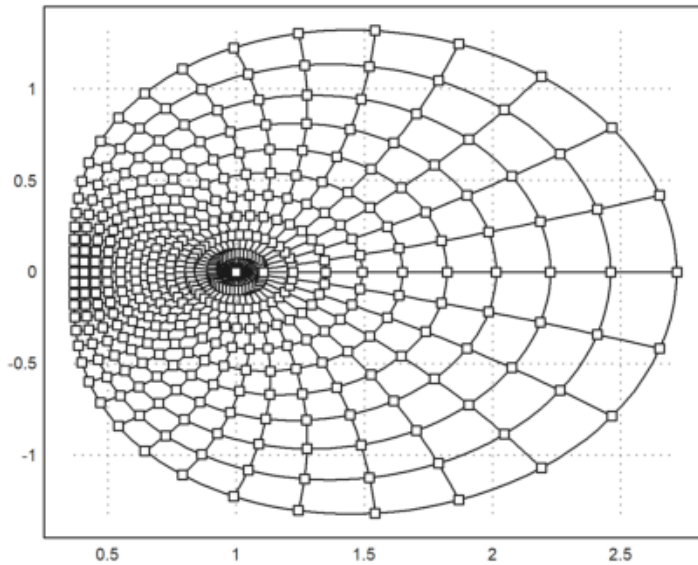
```
>aspect(); r=linspace(0,1,50); a=linspace(0,2pi,80)'; z=r*exp(I*a);...  
>plot2d(z,a=-1.25,b=1.25,c=-1.25,d=1.25,cgrid=10):
```



```
>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):
```

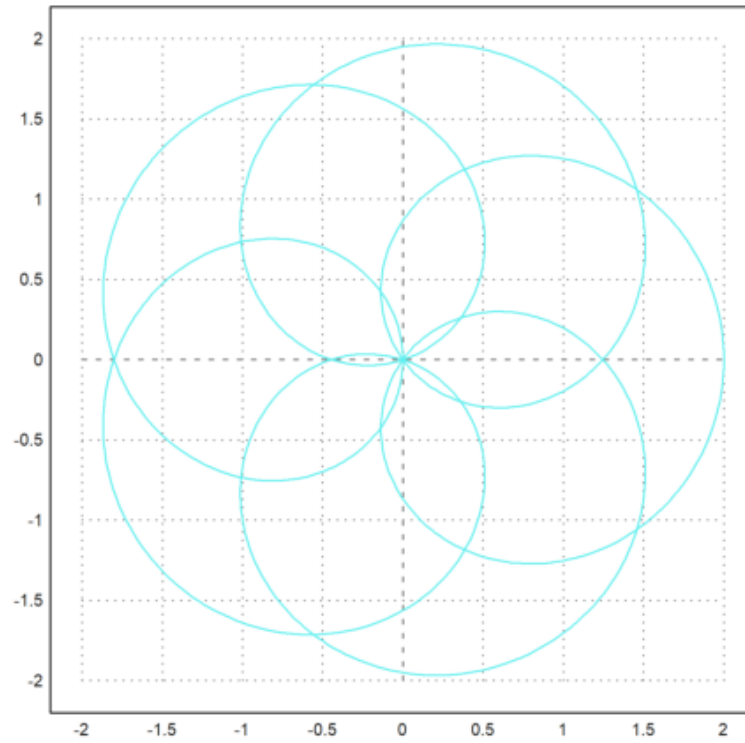


Vektor bilangan kompleks secara otomatis diplot sebagai kurva pada bidang kompleks dengan bagian nyata dan bagian imajiner.

Pada contoh, kami memplot lingkaran satuan dengan

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

```
>t=linspace(0,2pi,1000); ...
>plot2d(exp(I*t)+exp(6*I*t),r=2,color=lightblue):
```



Plot Statistik

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

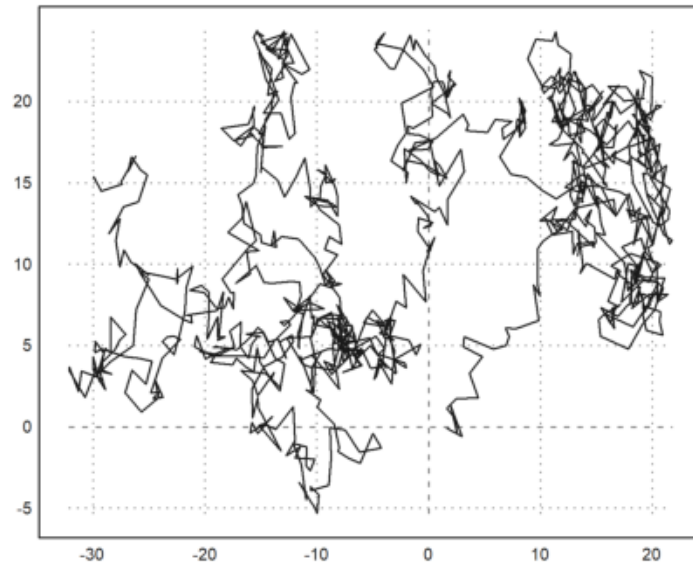
Jumlah kumulatif dari nilai berdistribusi normal 0-1 menghasilkan jalan acak.

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

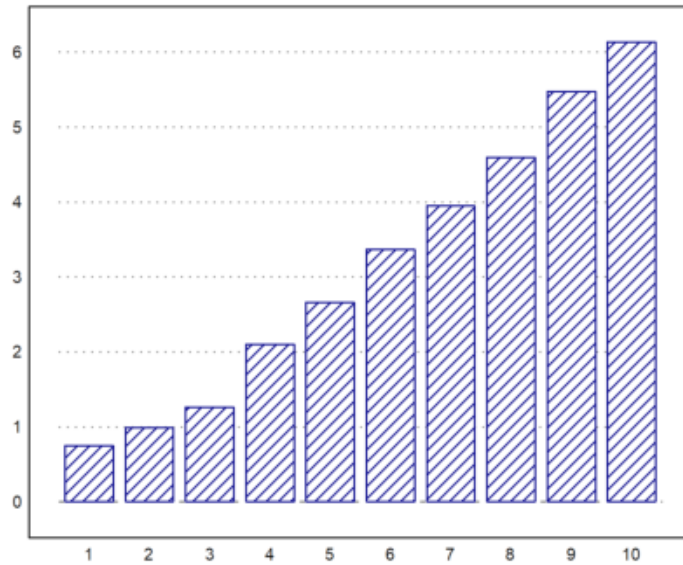


Menggunakan dua baris menunjukkan jalan dalam dua dimensi.

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

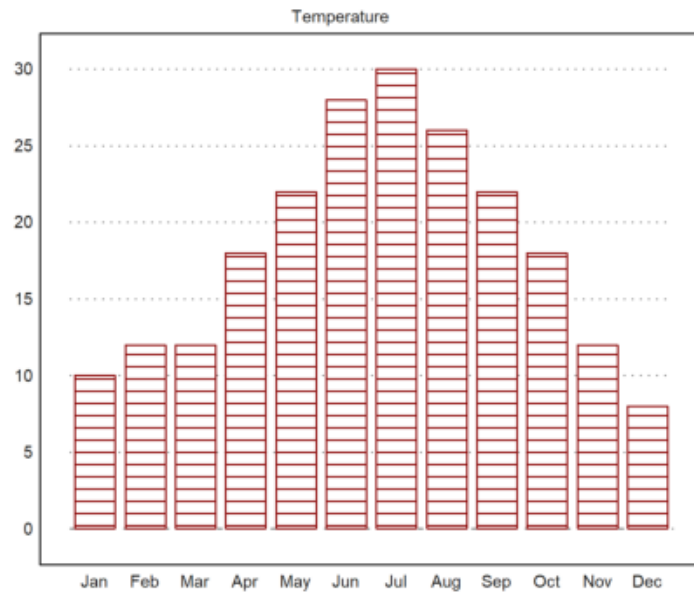


```
>columnplot(cumsum(random(10)),style="/",color=blue):
```

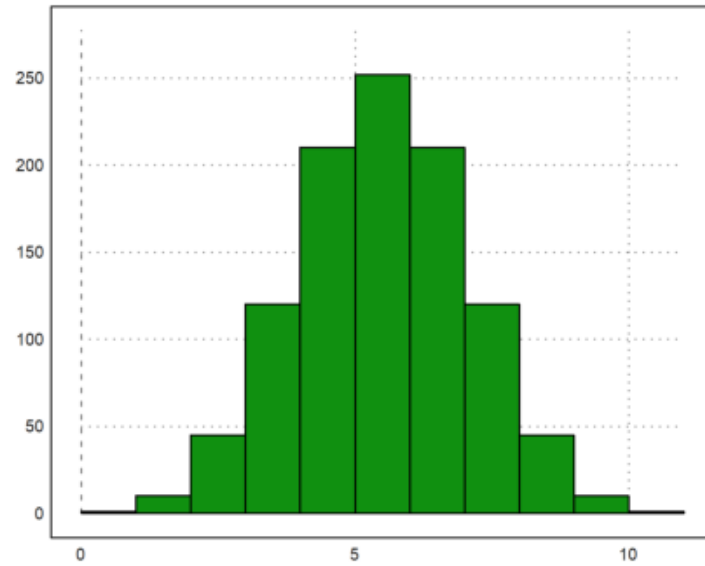


Ia juga dapat menampilkan string sebagai label.

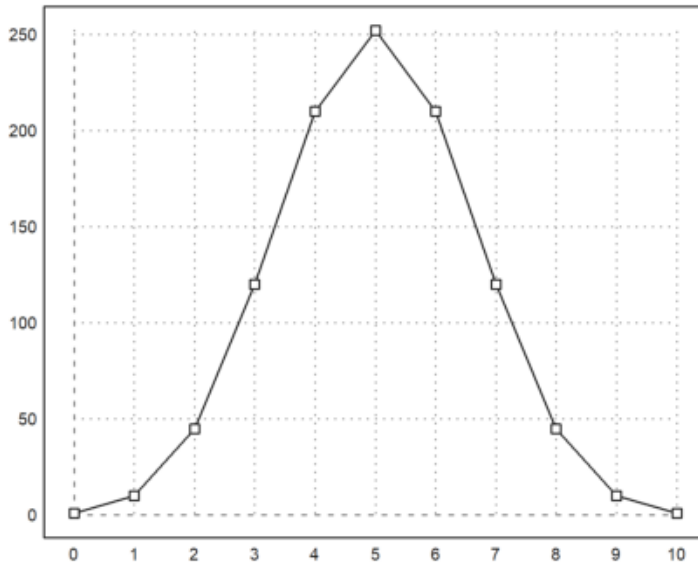
```
>months=["Jan","Feb","Mar","Apr","May","Jun", ...  
> "Jul","Aug","Sep","Oct","Nov","Dec"];  
>values=[10,12,12,18,22,28,30,26,22,18,12,8];  
>columnplot(values,lab=months,color=red,style="-");  
>title("Temperature"):
```



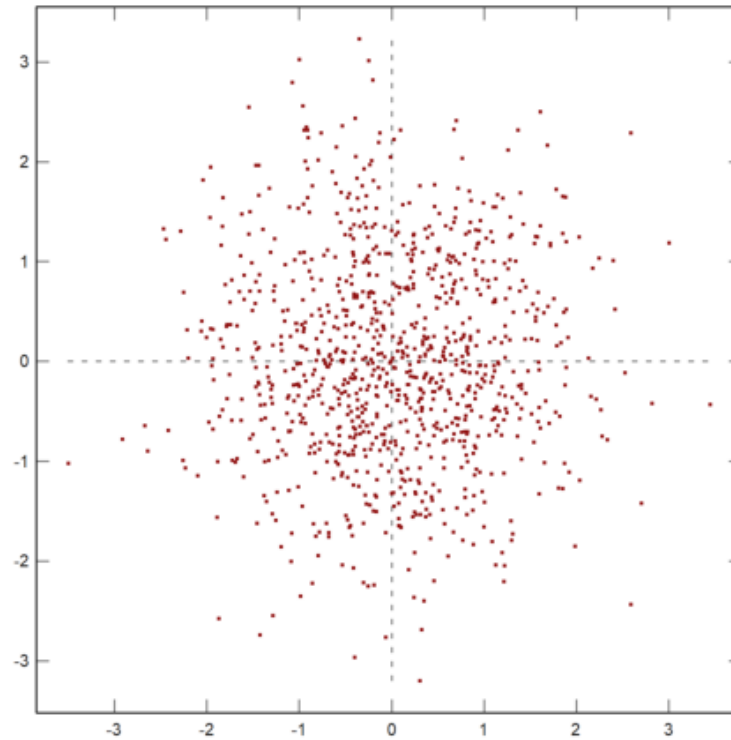
```
>k=0:10;  
>plot2d(k,bin(10,k),>bar):
```



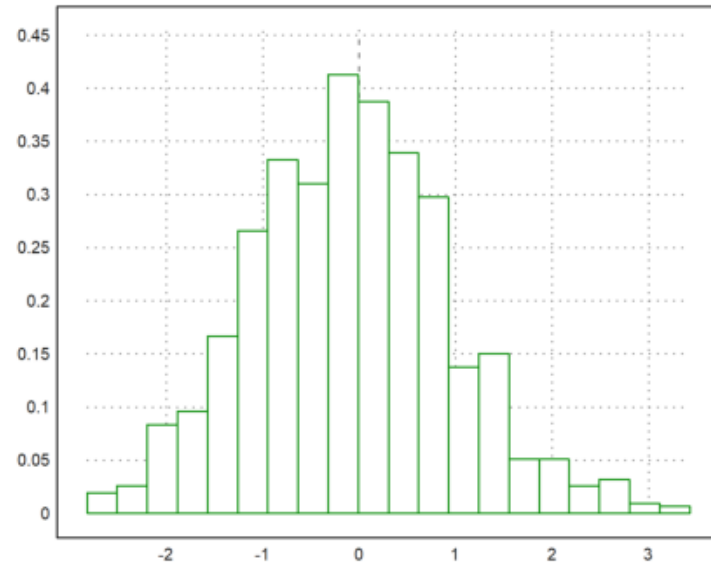
```
>plot2d(k,bin(10,k)); plot2d(k,bin(10,k),>points,>add):
```



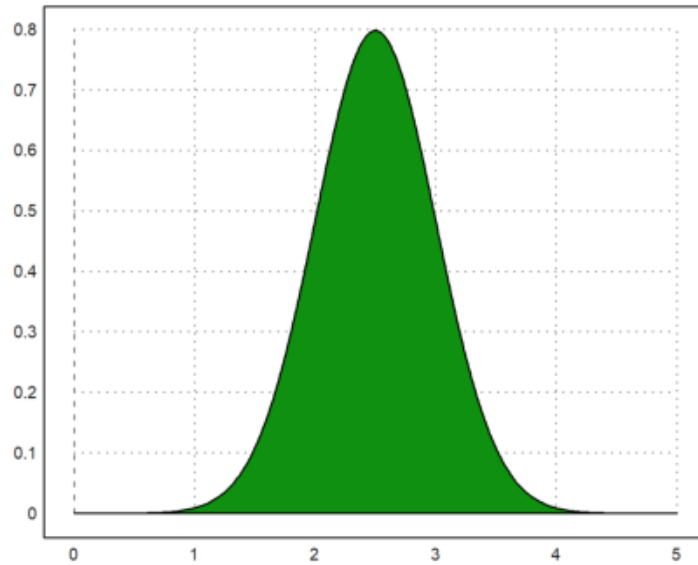
```
>plot2d(normal(1000),normal(1000),>points,grid=6,style="..",color=red):
```



```
>plot2d(normal(1,1000),>distribution,style="0"):
```

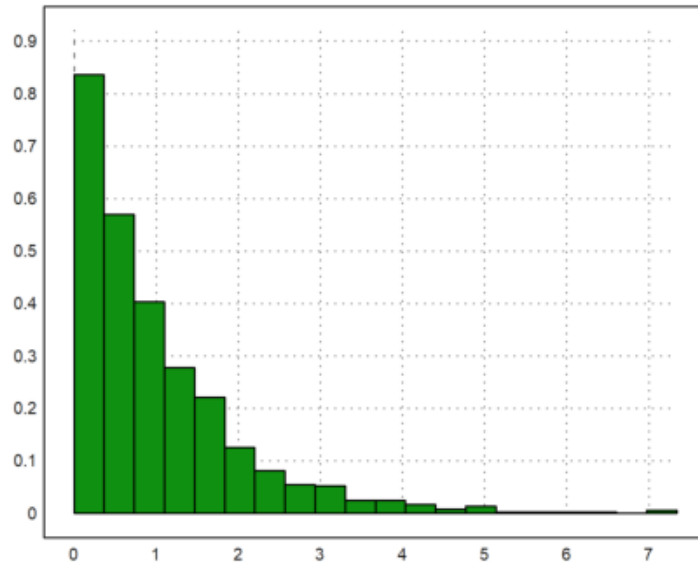


```
>plot2d("qnormal",0,5;2.5,0.5,>filled):
```



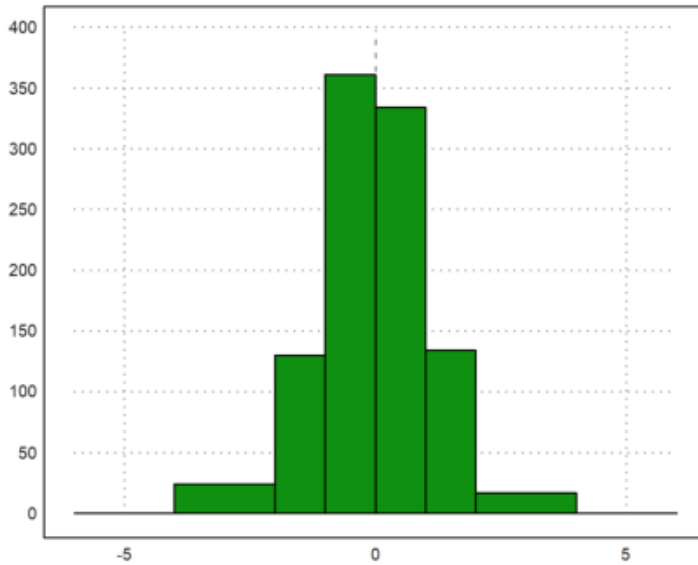
Untuk memplot distribusi statistik eksperimental, Anda dapat menggunakan `distribution=n` dengan `plot2d`.

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

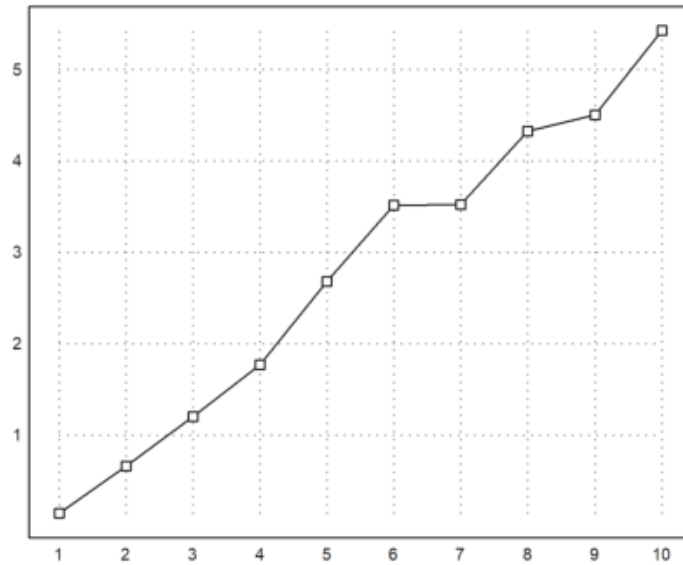
Atau Anda dapat menghitung distribusi dari data dan memplot hasilnya dengan `>bar` di `plot3d`, atau dengan `plot kolom`.

```
>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):
```

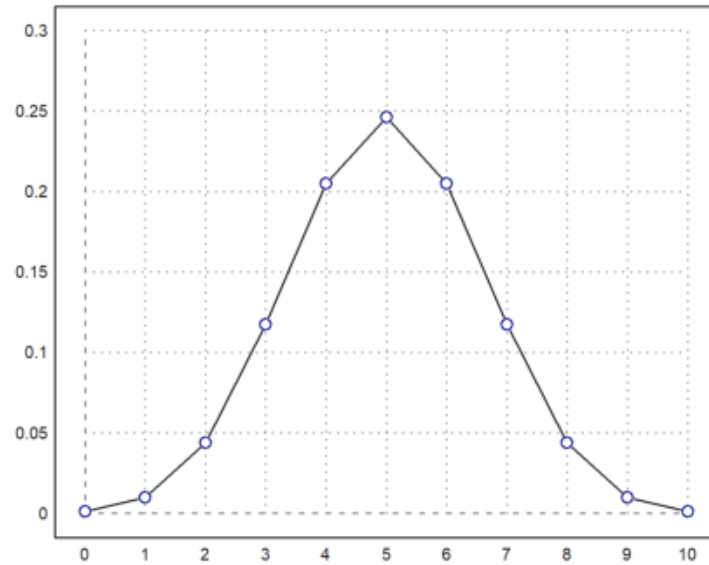


Fungsi `statplot()` mengatur gaya dengan string sederhana.

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



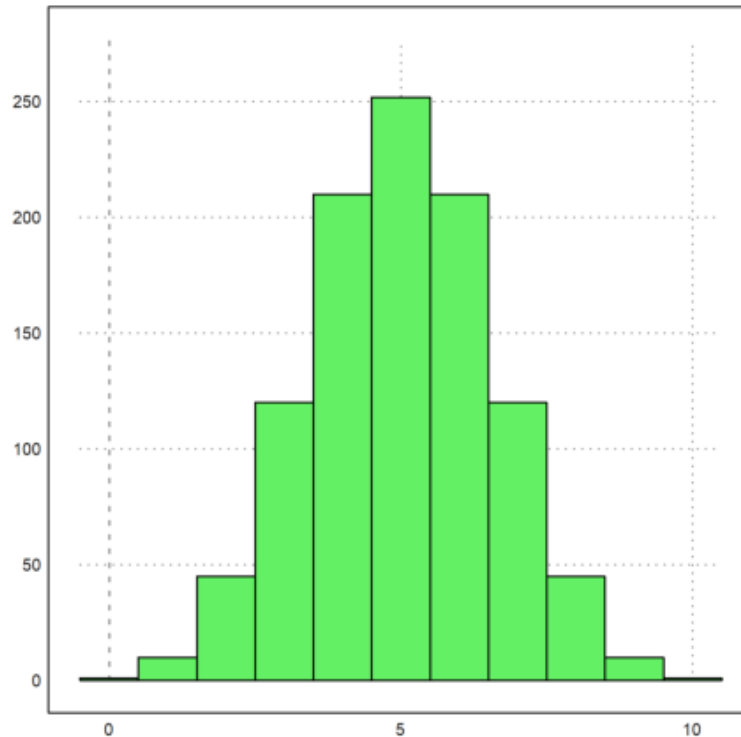
```
>n=10; i=0:n; ...  
>plot2d(i,bin(n,i)/2^n,a=0,b=10,c=0,d=0.3); ...  
>plot2d(i,bin(n,i)/2^n,points=true,style="ow",add=true,color=blue):
```



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

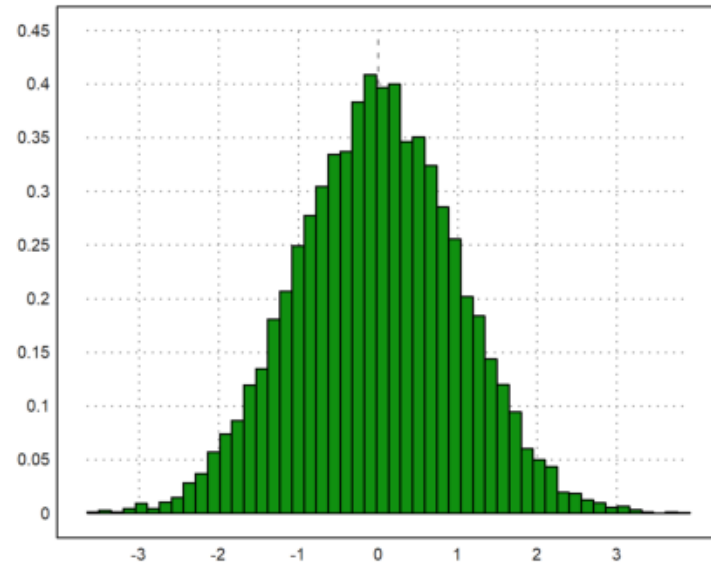
Gaya isian dapat digunakan seperti di atas.

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

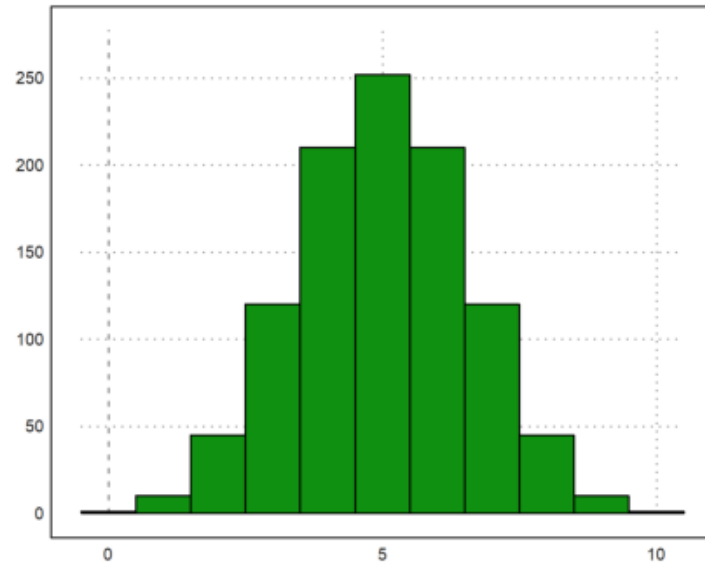


Data untuk plot batang (batang = 1) dan histogram (histogram = 1) dapat diberikan secara eksplisit dalam xv dan yv, atau dapat dihitung dari distribusi empiris dalam xv dengan `>distribusi` (atau `distribusi = n`). Histogram dari nilai xv akan dihitung secara otomatis dengan `>histogram`. Jika `>even`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

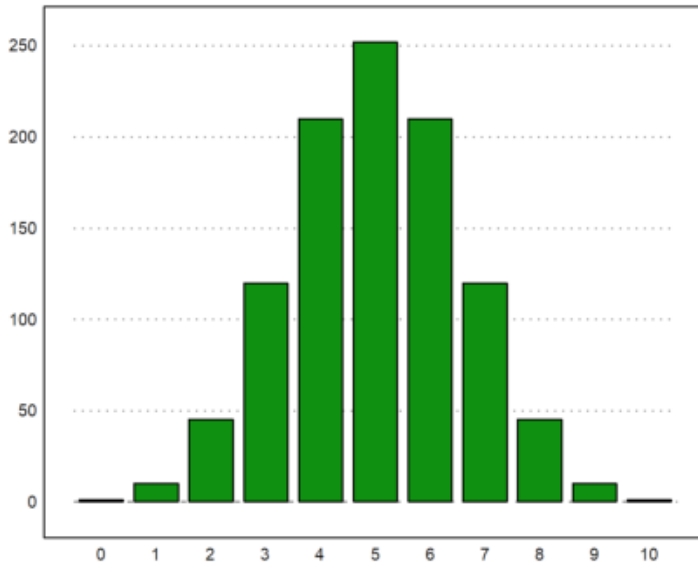
```
>plot2d(normal(10000),distribution=50):
```



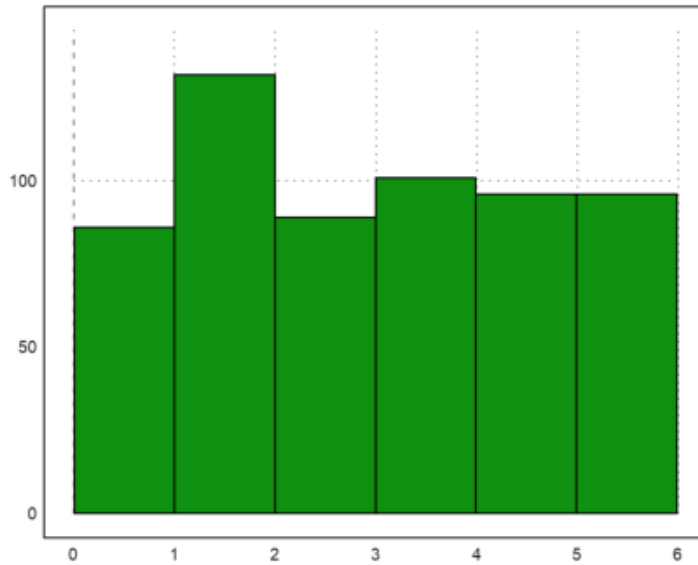
```
>k=0:10; m=bin(10,k); x=(0:11)-0.5; plot2d(x,m,>bar):
```



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

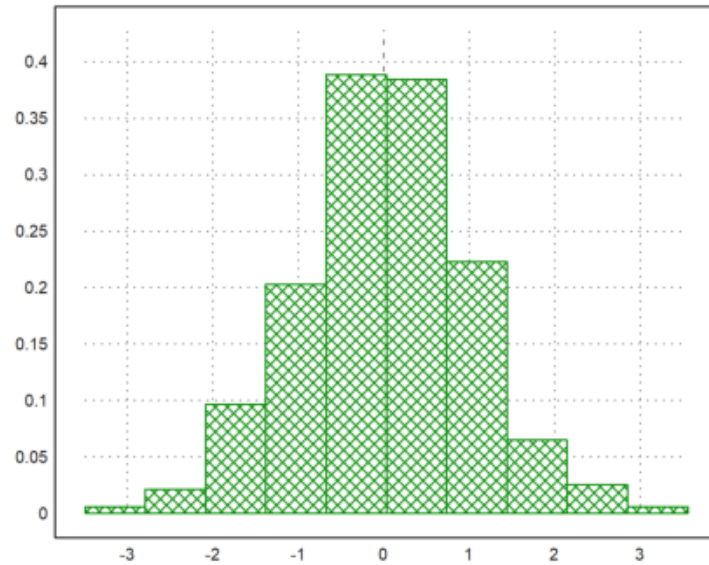


```
>plot2d(random(600)*6,histogram=6):
```

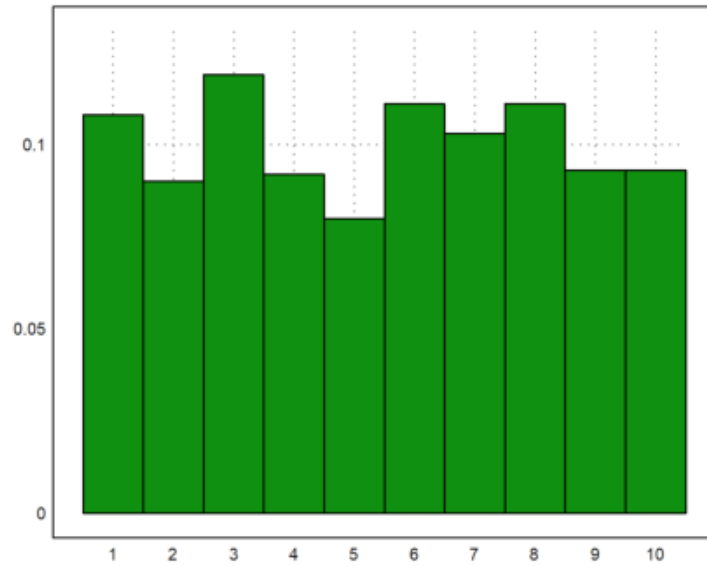
Untuk distribusi, ada parameter `distribution=n`, yang menghitung nilai secara otomatis dan mencetak distribusi relatif dengan `n` sub-interval.

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



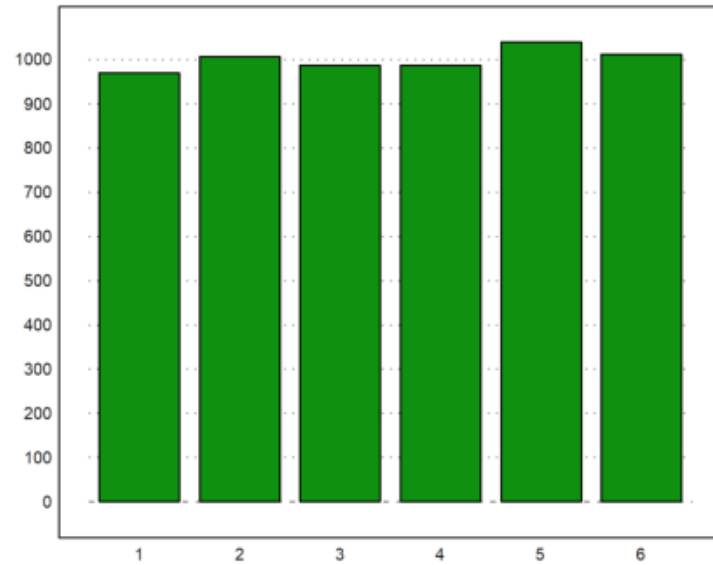
Dengan parameter `even=true`, ini akan menggunakan interval integer.

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

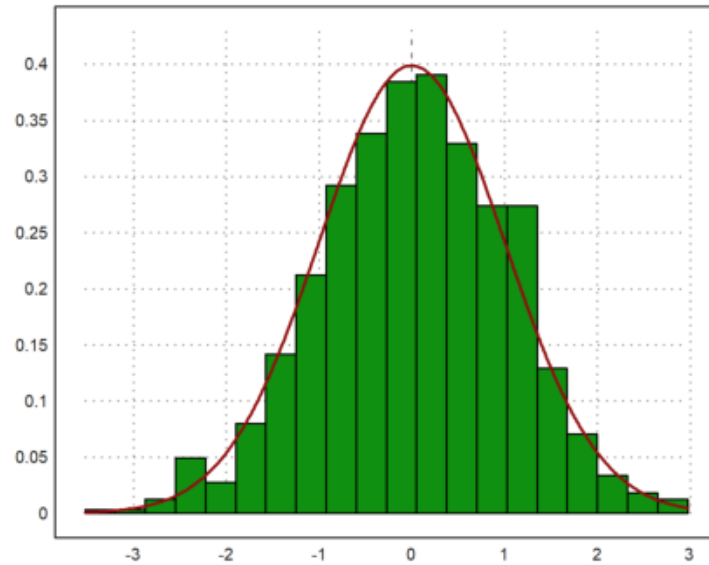


Perhatikan bahwa ada banyak plot statistik yang mungkin berguna. Lihatlah tutorial tentang statistik.

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

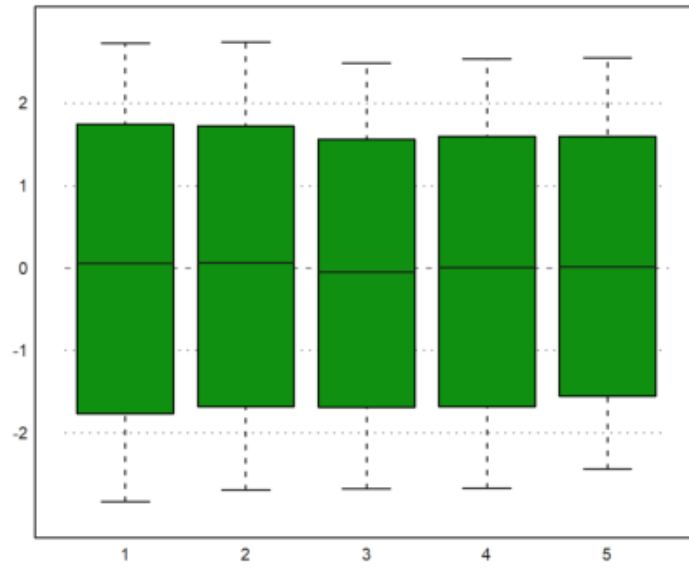


```
>plot2d(normal(1,1000),>distribution); ...  
> plot2d("qnormal(x)",color=red,thickness=2,>add):
```



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

```
>M=normal(5,1000); boxplot(quantiles(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 n garis level, yang akan menyebar di antara minimum dan maksimum fungsi secara merata. Warna yang lebih gelap atau lebih terang dapat ditambahkan dengan $>\text{hue}$ untuk mengindikasikan nilai fungsi. Untuk fungsi implisit, xv haruslah sebuah fungsi atau ekspresi dari parameter x dan y , atau, sebagai alternatif, xv dapat berupa matriks nilai.

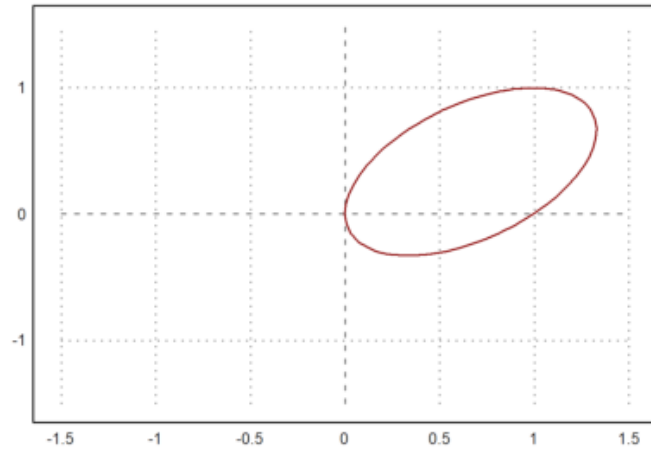
Euler dapat menandai garis level

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

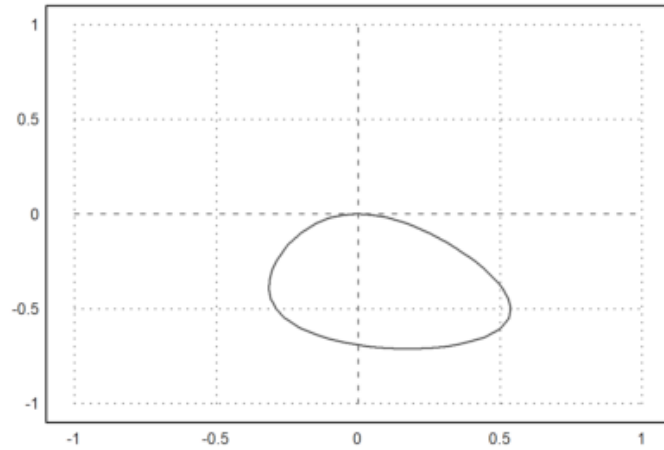
dari fungsi apa pun.

Untuk menggambar himpunan $f(x,y) = c$ untuk satu atau lebih konstanta c , Anda bisa menggunakan `plot2d()` dengan plot implisitnya pada bidang. Parameter untuk c adalah $\text{level} = c$, di mana c dapat berupa vektor garis level. Sebagai tambahan, sebuah skema warna dapat digambar pada latar belakang untuk mengindikasikan nilai fungsi untuk setiap titik pada plot. Parameter “ n ” menentukan kehalusan plot.

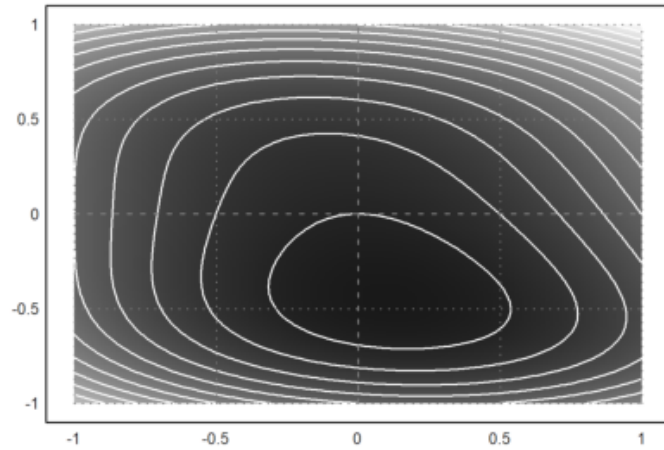
```
>aspect(1.5);  
>plot2d("x^2+y^2-x*y-x",r=1.5,level=0,contourcolor=red):
```



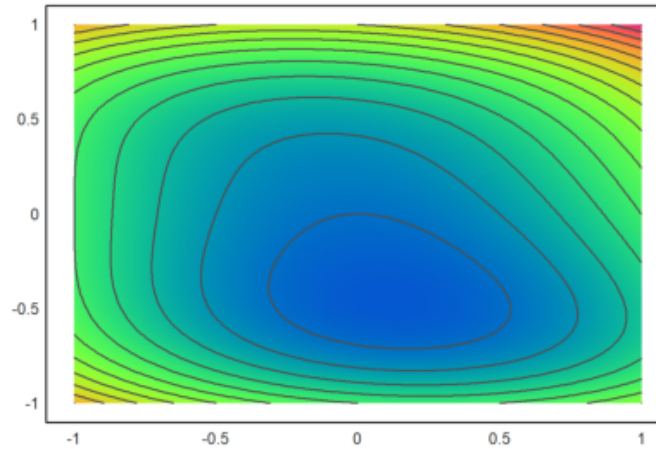
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=0): // Solutions of f(x,y)=0
```

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

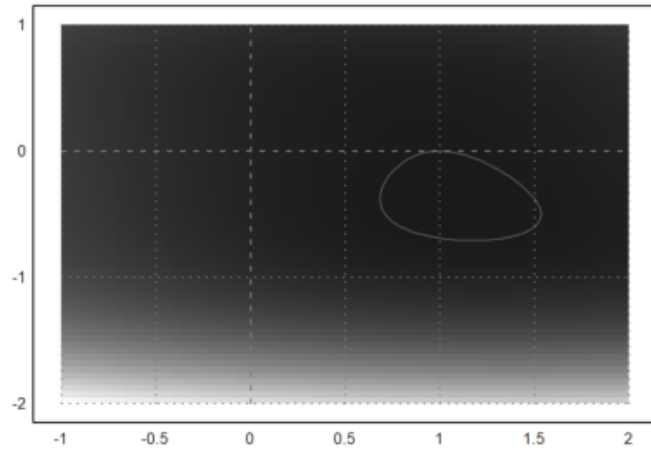


```
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=200,grid=4): // nicer
```

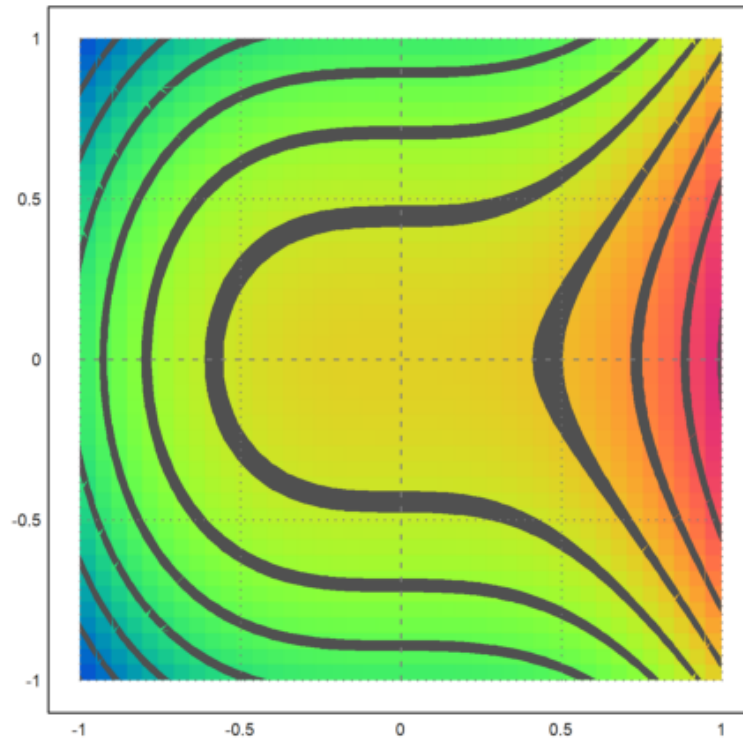


Hal ini juga berlaku untuk plot data. Tetapi Anda harus menentukan rentang untuk label sumbu.

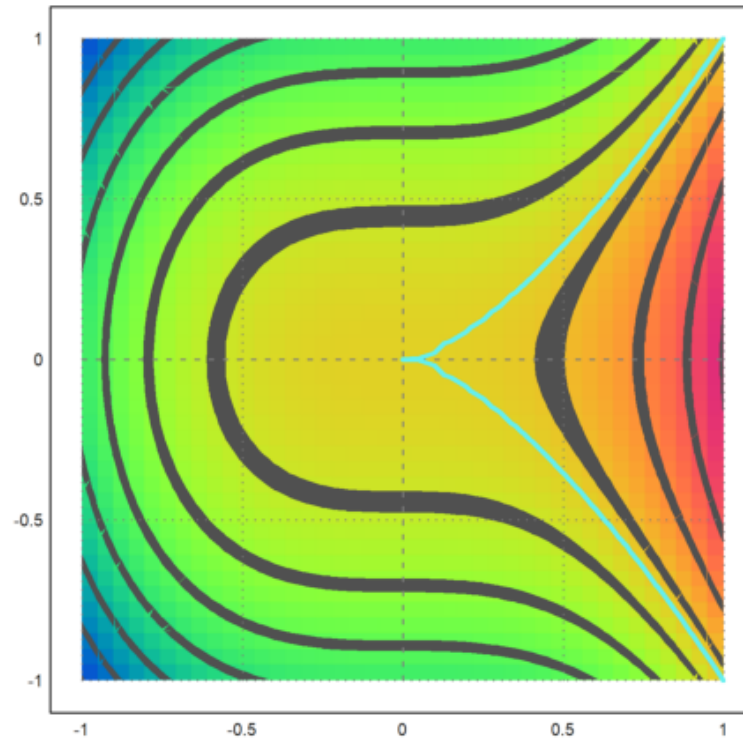
```
>x=-2:0.05:1; y=x'; z=expr(x,y);  
>plot2d(z,level=0,a=-1,b=2,c=-2,d=1,>hue):
```



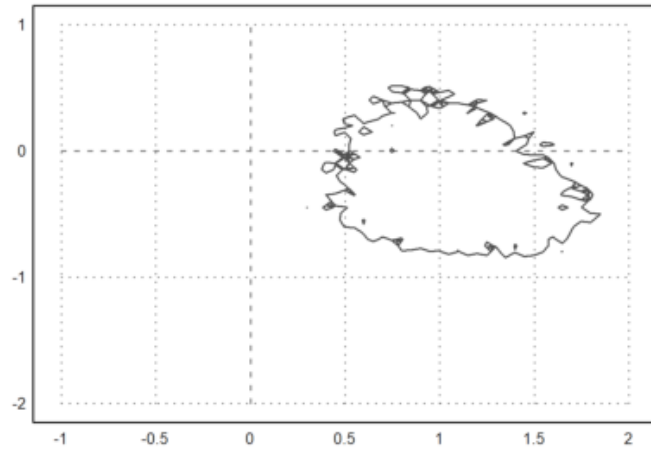
```
>plot2d("x^3-y^2",>contour,>hue,>spectral):
```



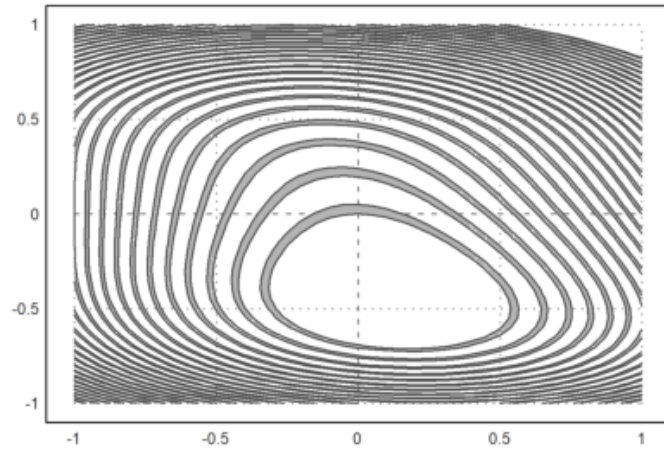
```
>plot2d("x^3-y^2",level=0,contourwidth=3,>add,contourcolor=lightblue):
```



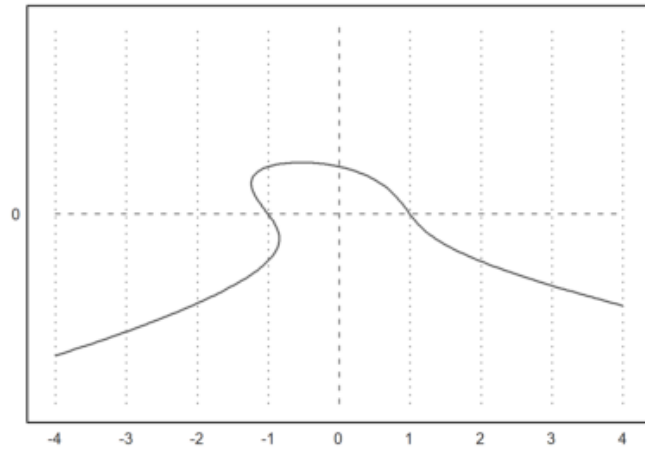
```
>z=z+normal(size(z))*0.2;  
>plot2d(z,level=0.5,a=-1,b=2,c=-2,d=1):
```



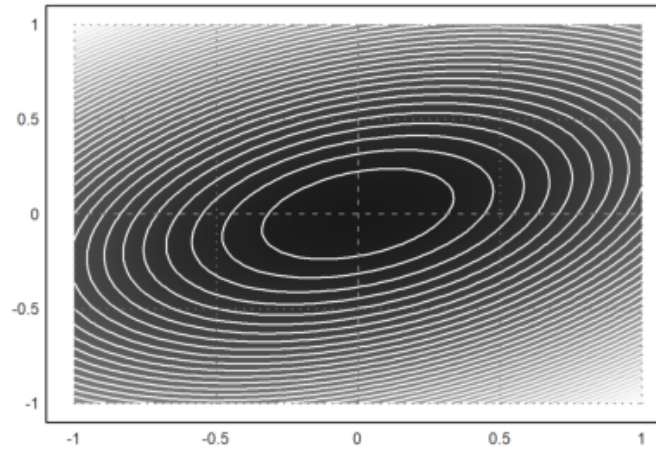
```
>plot2d(expr,level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
```



```
>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):
```



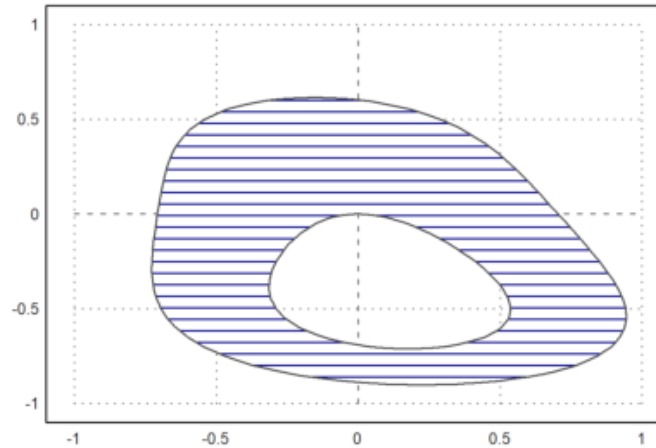
Dimungkinkan juga untuk mengisi set

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

dengan rentang level.

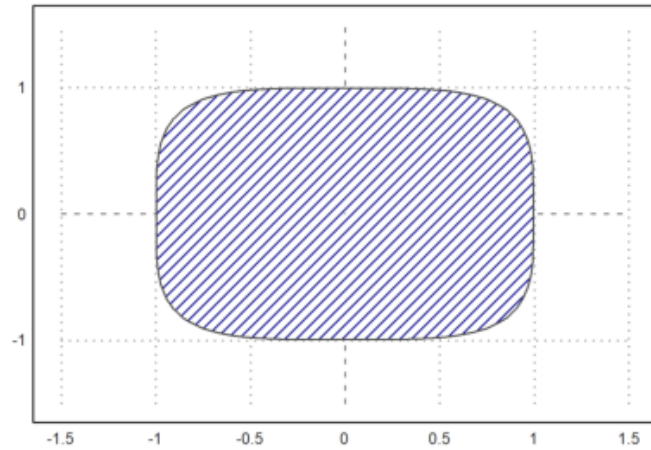
Dimungkinkan untuk mengisi wilayah nilai untuk fungsi tertentu. Untuk ini, level harus berupa matriks 2xn. Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

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

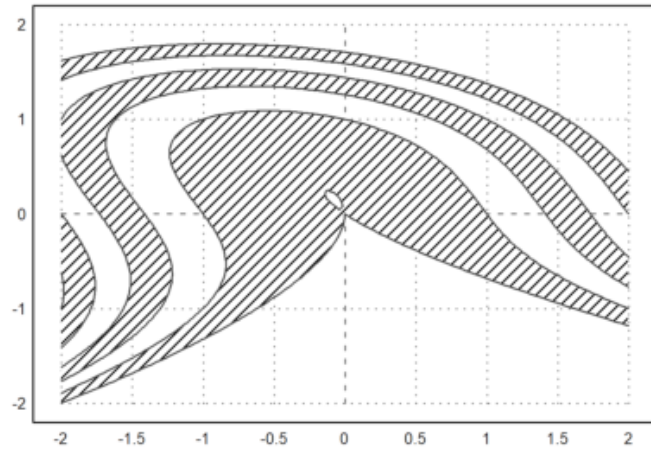


Plot implisit juga dapat menunjukkan rentang level. Maka level harus berupa matriks 2xn interval level, di mana baris pertama berisi awal dan baris kedua adalah akhir dari setiap interval. Sebagai alternatif, vektor baris sederhana dapat digunakan untuk level, dan parameter dl memperluas nilai level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

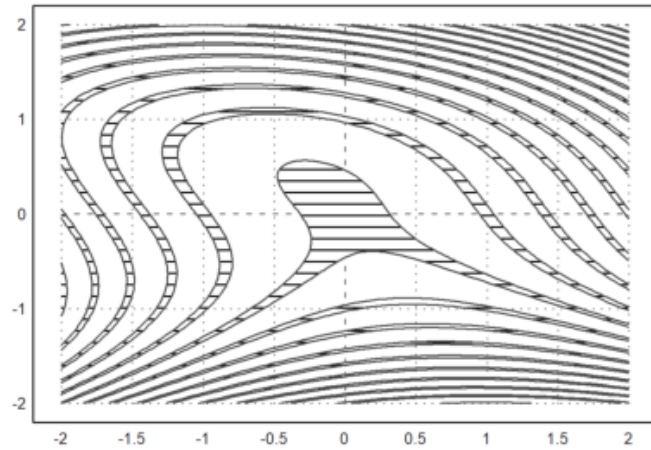
```
>plot2d("x^4+y^4",r=1.5,level=[0;1],color=blue,style="/"):
```



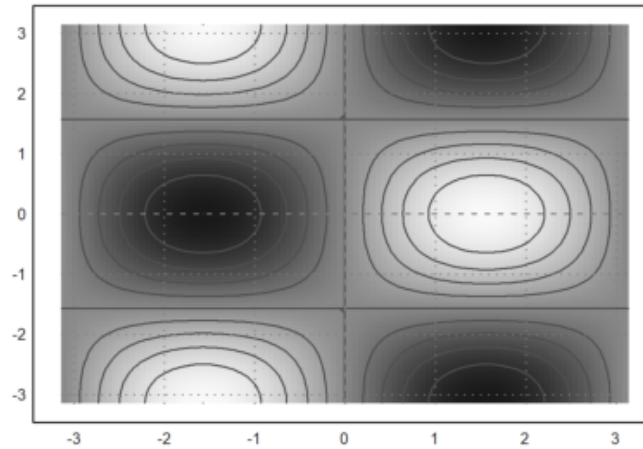
```
>plot2d("x^2+y^3+x*y",level=[0,2,4;1,3,5],style="/",r=2,n=100):
```



```
>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):
```

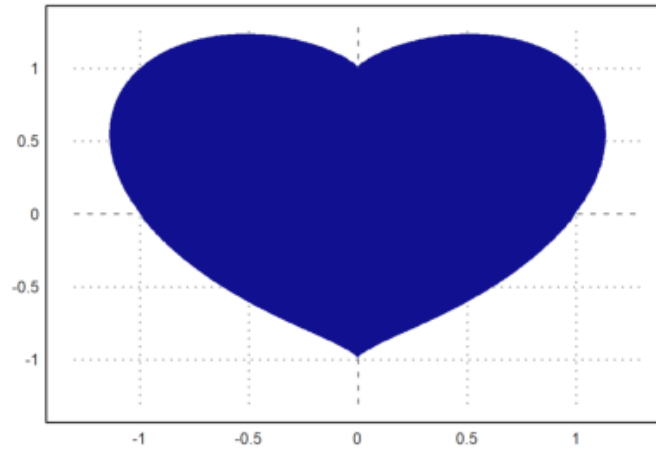


Anda juga dapat menandai suatu wilayah

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

Hal ini dilakukan dengan menambahkan level dengan dua baris.

```
>reset;
>aspect(1.5); plot2d("(x^2+y^2-1)^3-x^2*y^3",r=1.3, ...
> style="#",color=blue,<outline, ...
> level=[-2;0],n=100):
```

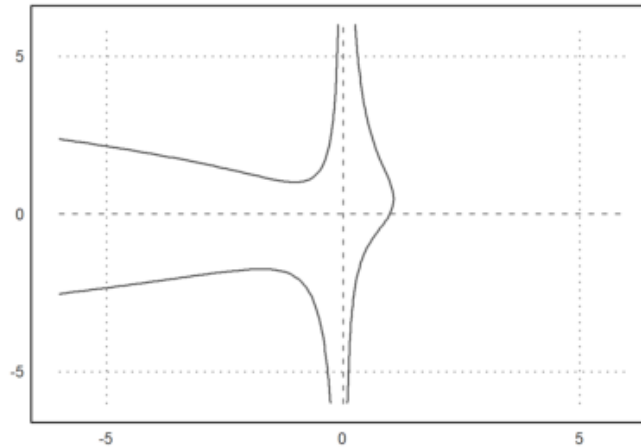


>

Dimungkinkan untuk menentukan level tertentu. Sebagai contoh, kita dapat memplot solusi dari 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);
  end
end

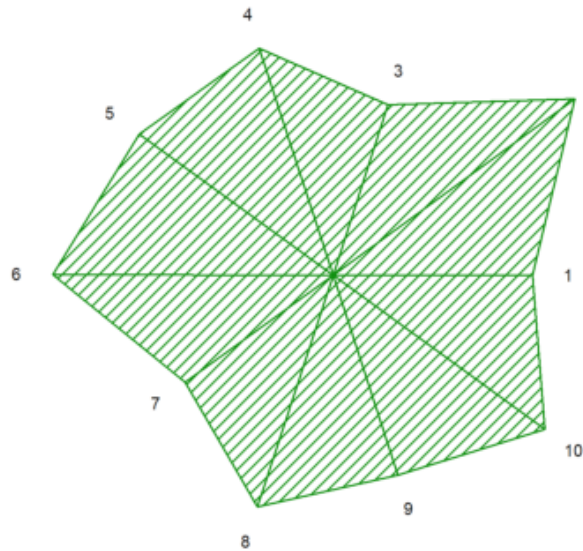
```

```
        endif;  
    end;  
    barcolor(cl); barstyle(st);  
    holding(h);  
    window(w);  
endfunction
```

Tidak ada kisi-kisi atau kutu sumbu di sini. Selain itu, kami menggunakan jendela penuh untuk plot.

Kami memanggil reset sebelum kami menguji plot ini untuk mengembalikan default grafis. Hal ini tidak perlu dilakukan, jika Anda yakin bahwa plot Anda berfungsi.

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



Terkadang, Anda mungkin ingin memplot sesuatu yang tidak dapat dilakukan oleh plot2d, tetapi hampir. Pada fungsi berikut ini, kita akan 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

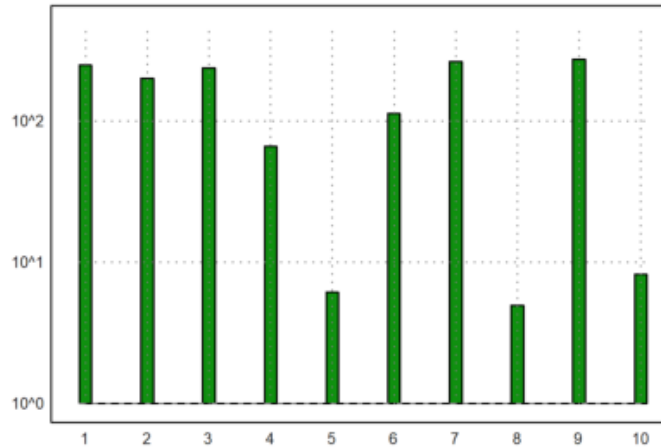
```

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 menghidupkan kurva 2D dengan 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 pada 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(1);  
endfunction
```

Tekan tombol apa saja untuk menghentikan animasi ini.

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

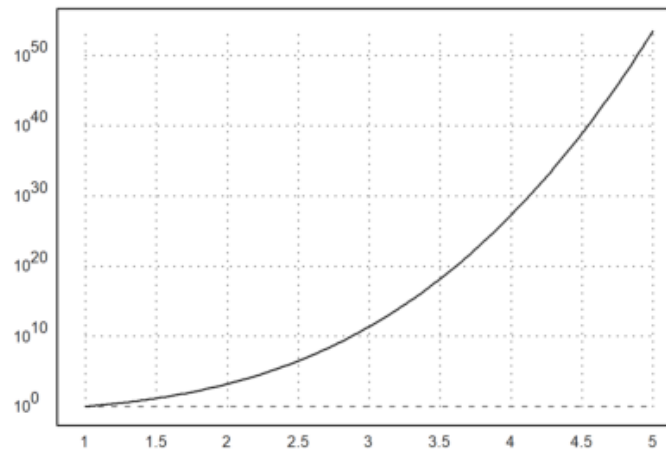
Plot Logaritmik

EMT menggunakan parameter “logplot” untuk skala logaritmik.

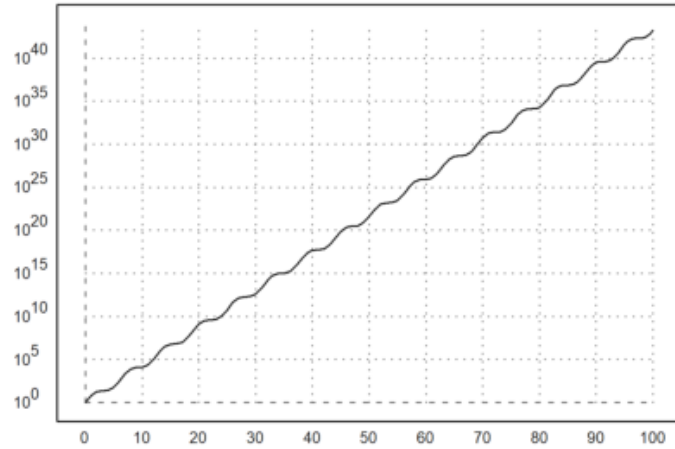
Plot logaritmik dapat diplot menggunakan skala logaritmik dalam y dengan `logplot = 1`, atau menggunakan skala logaritmik dalam x dan y dengan `logplot = 2`, atau dalam x dengan `logplot = 3`.

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

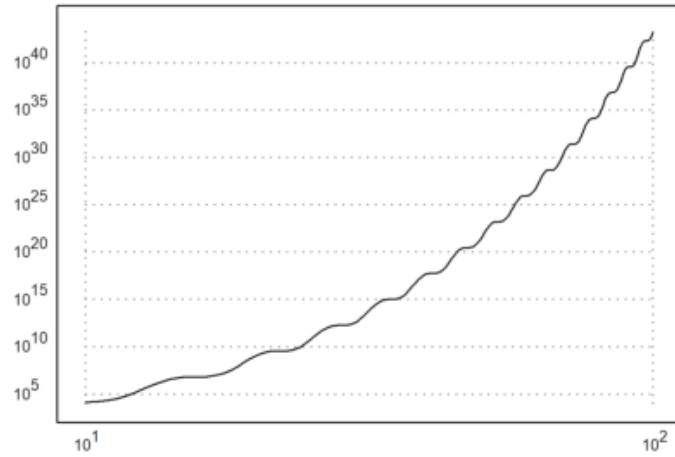
```
>plot2d("exp(x^3-x)*x^2",1,5,logplot=1):
```



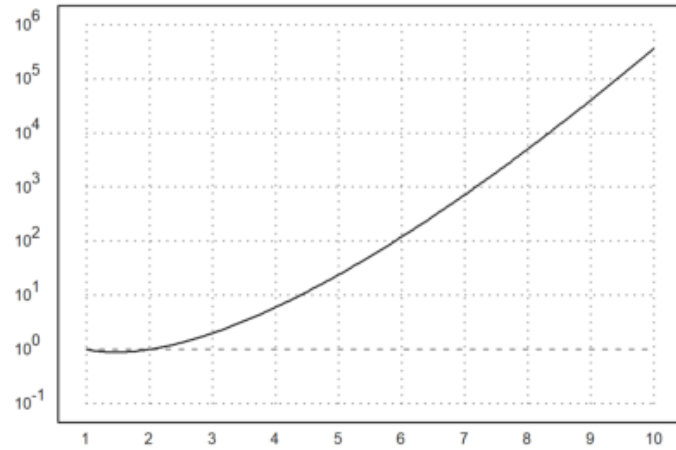
```
>plot2d("exp(x+sin(x))",0,100,logplot=1):
```



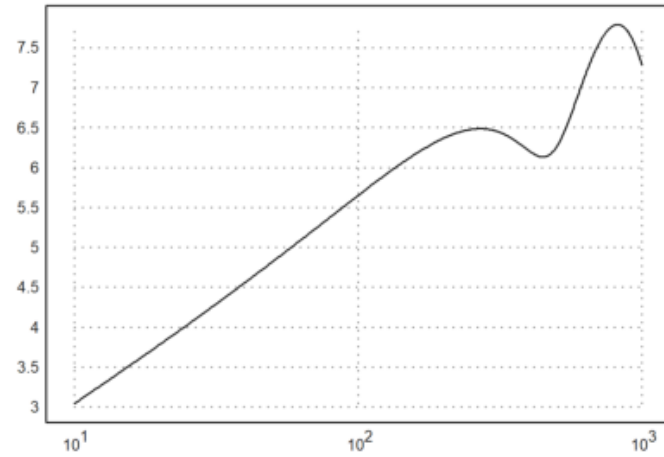
```
>plot2d("exp(x+sin(x))",10,100,logplot=2):
```

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

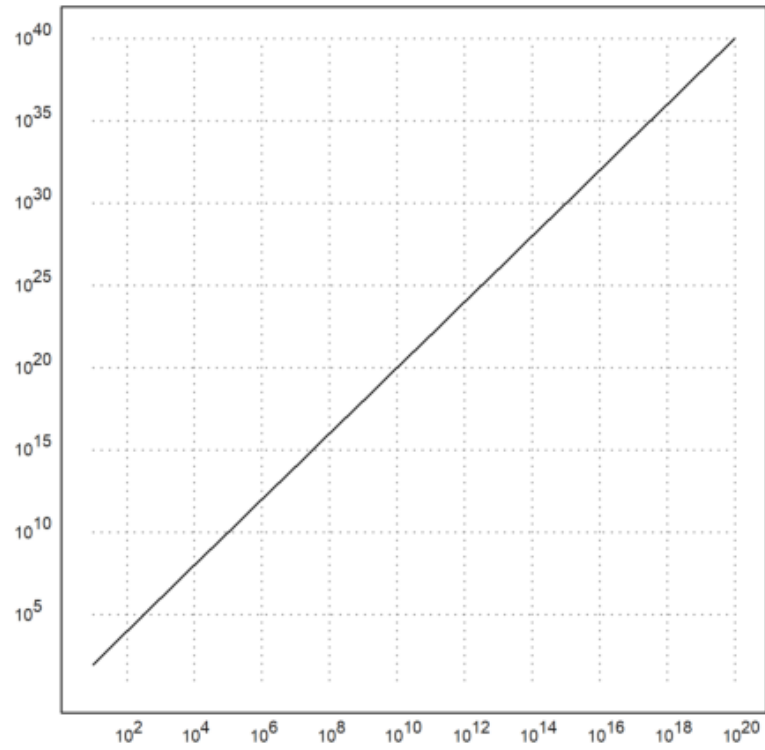


```
>plot2d("log(x*(2+sin(x/100)))",10,1000,logplot=3):
```

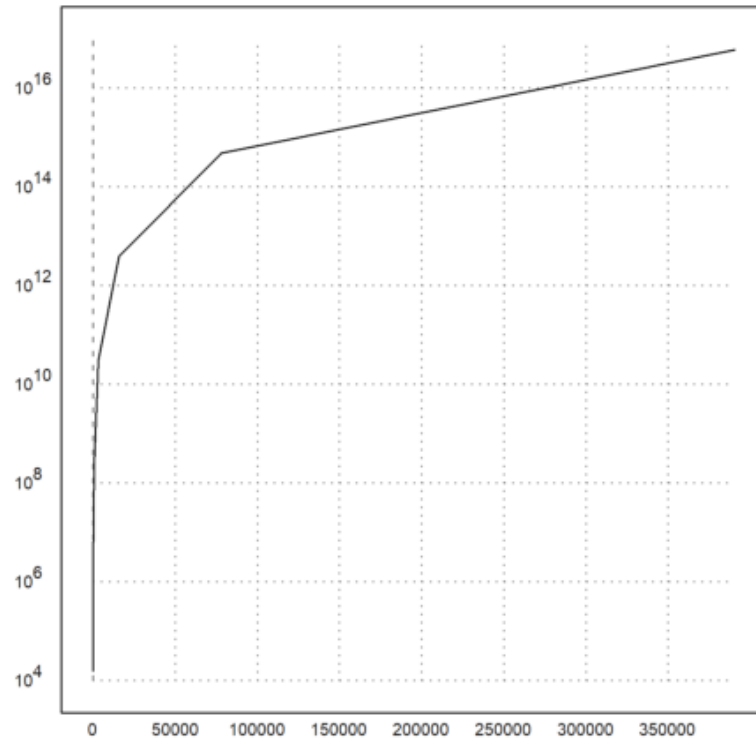


Hal ini juga berlaku untuk plot data.

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



```
>reset; x=5^(2:8); y=x^3-2*x;  
>plot2d(x,y,logplot=4):
```



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
"#", "#0", "0", "/", "\", "\/",
"+", "|", "-", "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`.