

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

Terdapat beberapa fungsi dasar dalam membuat grafik. Ada koordinat layar yang selalu berada di rentang 0 hingga 1024 di setiap sumbu, terlepas layar berbentuk persegi maupun tidak. Terdapat pula koordinat grafik yang dapat diatur dengan `setplot()`. Pemetaan antar koordinat bergantung pada jendela plot yang sedang aktif. Misalnya fungsi bawaan `shrinkwindow()` menyisakan ruang untuk label sumbu dan judul plot.

Pada contoh, kita hanya menggambar beberapa garis acak dalam berbagai warna. Untuk instruksi yang lebih detail pada fungsi ini, dapat mempelajari 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
```



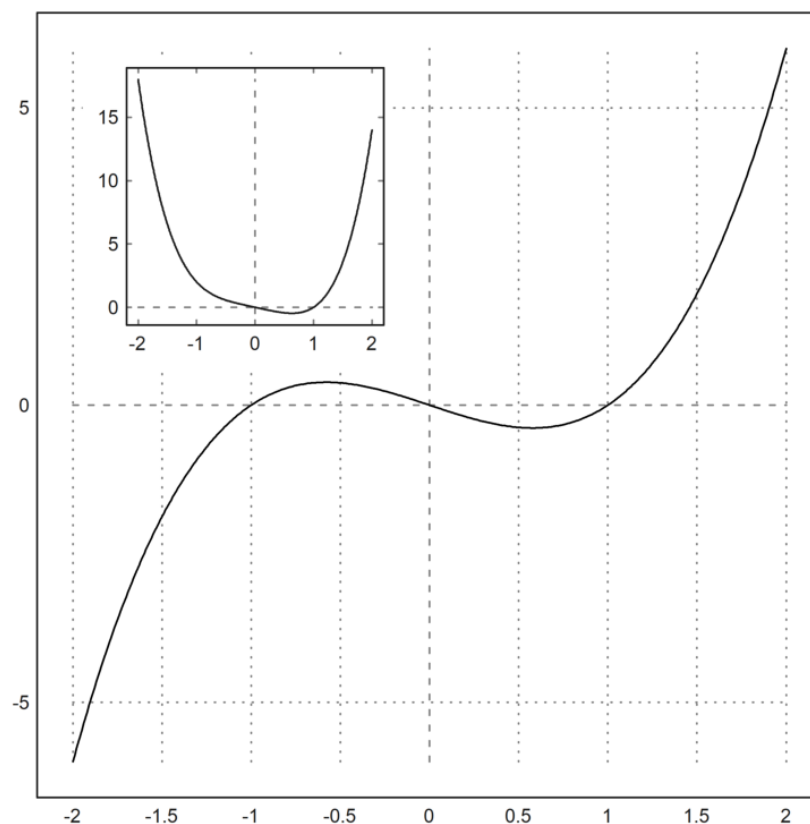
```
>reset;
```

Penting untuk meng-hold grafik karena plot() akan membersihkan jendela plot. Untuk menghapus semua grafik/plot dapat menggunakan reset()

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

Sebagai contoh lain, kita dapat menggambar sebuah grafik kecil (inset) di dalam grafik utama. Ini dilakukan dengan mendefinisikan jendela plot yang lebih kecil di dalam jendela yang lebih besar. Perlu diperhatikan bahwa jendela kecil ini tidak secara otomatis menyediakan ruang untuk label sumbu di luar area plot, sehingga kita perlu menambahkan margin secara manual jika diperlukan. Selain itu, kita harus menyimpan dan mengembalikan pengaturan jendela penuh, serta menggunakan perintah hold agar grafik utama tetap ditampilkan saat kita menambahkan grafik 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);
```

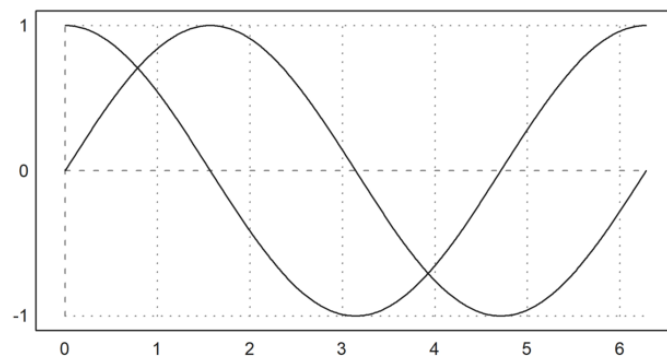
Plot dengan beberapa gambar dapat dibuat dengan cara yang sama. Ada fungsi utilitas figure() untuk itu.

Plot Aspect

Plot bawaan menggunakan jendela berbentuk persegi. Anda bisa mengubahnya dengan fungsi aspect(). Jangan lupa untuk mereset kembali aspek tersebut nanti. Anda juga bisa mengubah bawaan ini lewat menu dengan 'Set Aspect' ke rasio tertentu atau ke ukuran jendela grafik saat ini.

Akan tetapi, Anda juga bisa mengubahnya hanya untuk satu plot saja. Untuk itu, ukuran area plot saat ini diubah, dan jendela disesuaikan supaya label punya cukup ruang.

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



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

Fungsi reset() mengembalikan pengaturan plot ke kondisi awal, termasuk rasio aspeknya.

2D Plots in Euler

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

Dimungkinkan juga untuk melakukan plot di Maxima menggunakan Gnuplot atau di Python menggunakan Matplotlib. Euler dapat membuat plot 2D dari:

- ekspresi,
- fungsi, variabel, atau kurva parametrik,
- vektor nilai x-y,
- sebaran titik pada bidang,
- kurva implisit dengan level atau daerah level,
- fungsi kompleks.

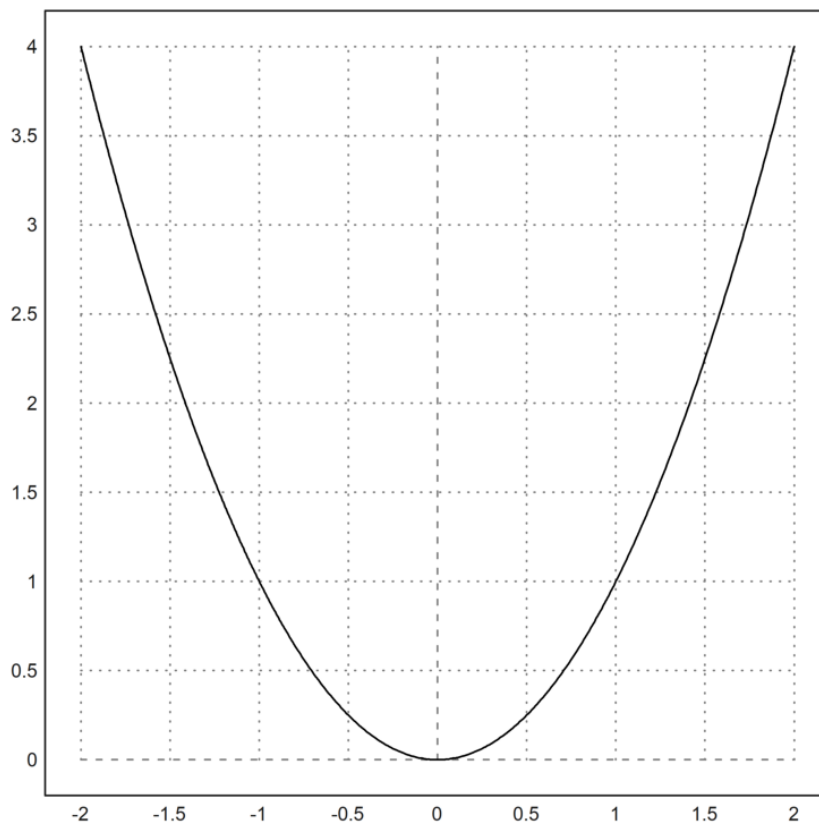
Gaya plot mencakup berbagai gaya untuk garis dan titik, diagram batang, serta plot berarsir.

Plot Ekspresi atau Variabel

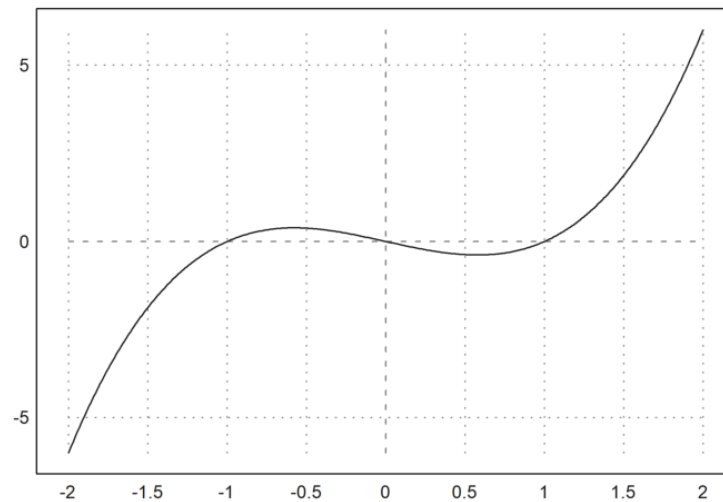
Sebuah ekspresi tunggal dalam variabel x (misalnya $4x^2$) atau nama sebuah fungsi (misalnya f) akan menghasilkan grafik dari fungsi tersebut. Berikut contoh paling dasar, yang menggunakan rentang bawaan (default range) dan menetapkan rentang y yang sesuai agar grafik fungsi dapat ditampilkan dengan baik.

Catatan: Jika sebuah baris perintah diakhiri dengan titik dua `:`, maka grafik akan dimasukkan ke dalam jendela teks. Jika tidak, tekan tombol TAB untuk menampilkan grafik apabila jendela plot tertutup.

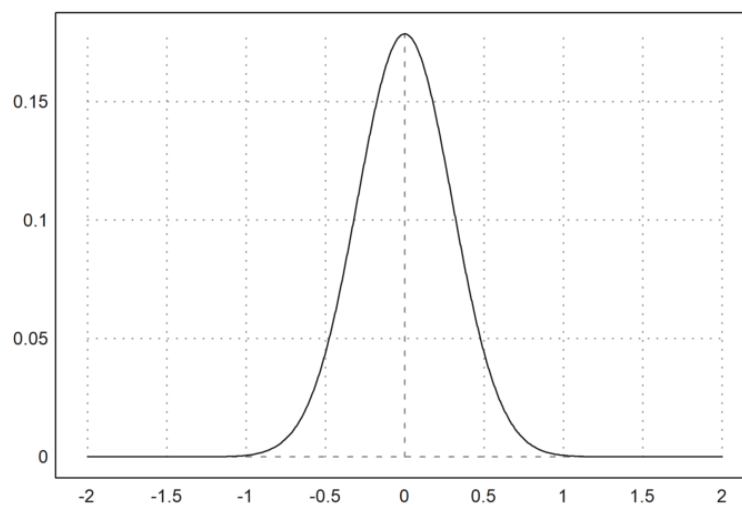
```
>plot2d("x^2"):
```



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



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

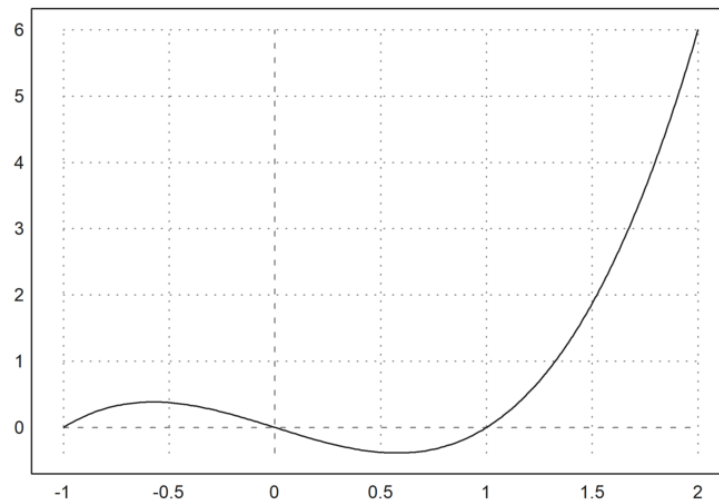


Dari beberapa contoh sebelumnya Anda dapat melihat bahwa aslinya gambar plot menggunakan sumbu X dengan rentang nilai dari -2 sampai dengan 2. Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan nilai-nilai batas X (dan Y) di belakang ekspresi yang digambar.

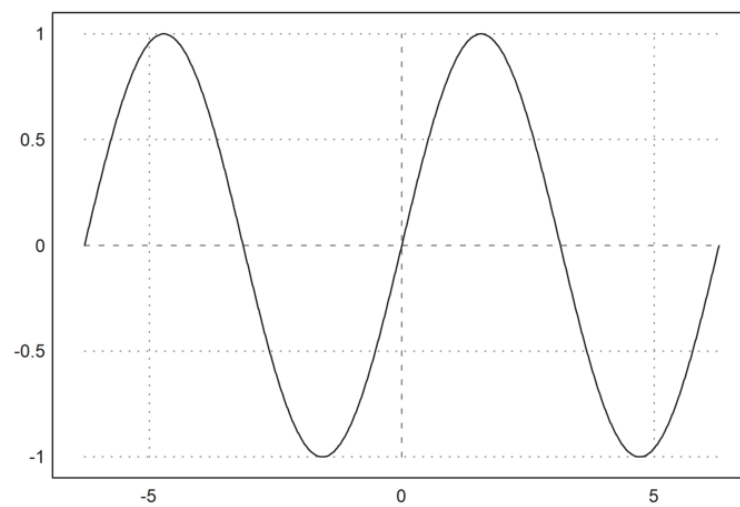
Rentang plot ditentukan melalui parameter yang telah ditetapkan sebagai berikut:

- a, b: rentang sumbu x [default, -2,2]
- c, d: rentang sumbu y [default: menyesuaikan nilai]
- r: jari-jari alternatif yang digunakan untuk menentukan rentang plot berdasarkan pusat plot.
- cx, cy: koordinat pusat plot [default 0,0]

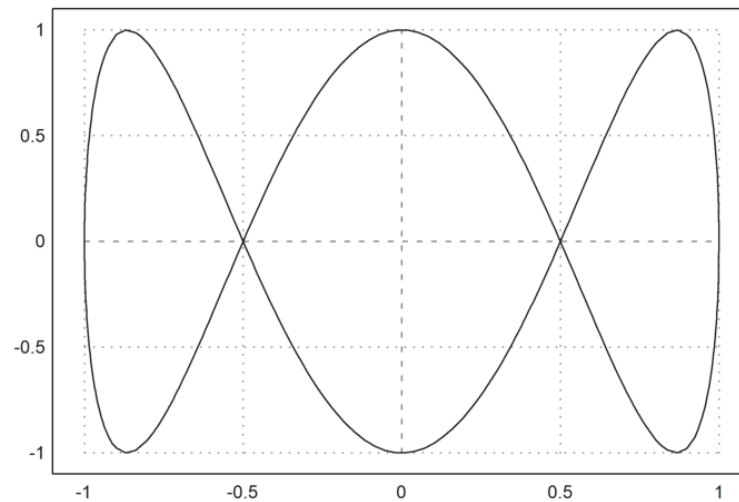
```
>plot2d("x^3-x",-1,2):
```



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



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



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

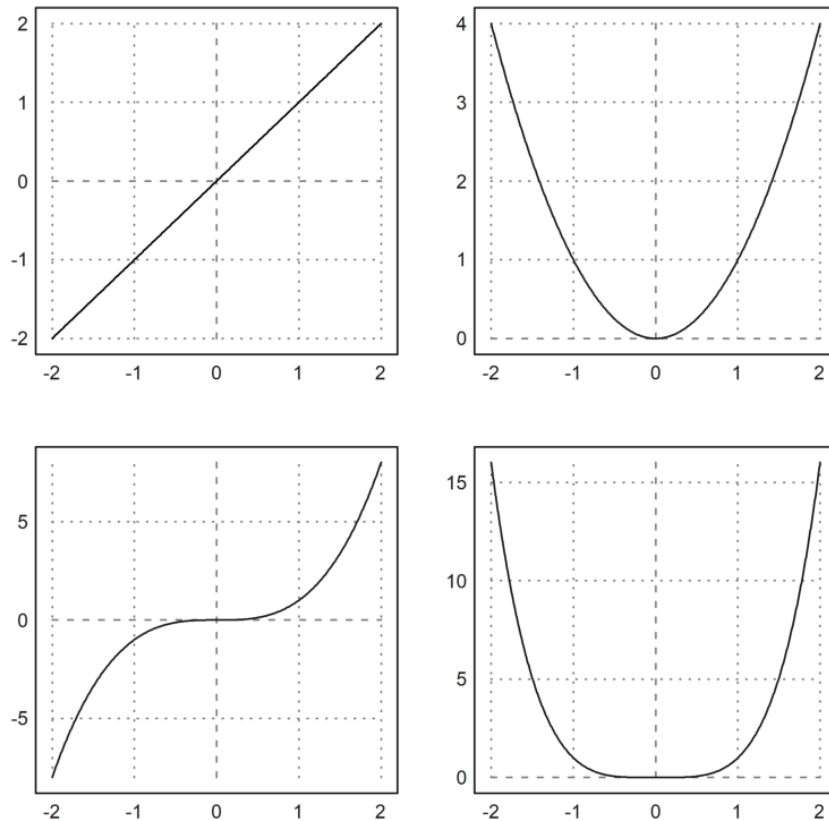
Melalui opsi yang tersedia, tampilan plot dapat diatur agar:

- muncul pada jendela terpisah yang dapat diubah ukurannya, atau
- ditampilkan langsung pada jendela notebook.

Selain itu, gaya plot yang lebih beragam dapat diperoleh dengan menggunakan perintah plot yang lebih spesifik. Apabila plot tertutup atau tidak terlihat, tekan tombol Tabulator untuk menampilkannya kembali.

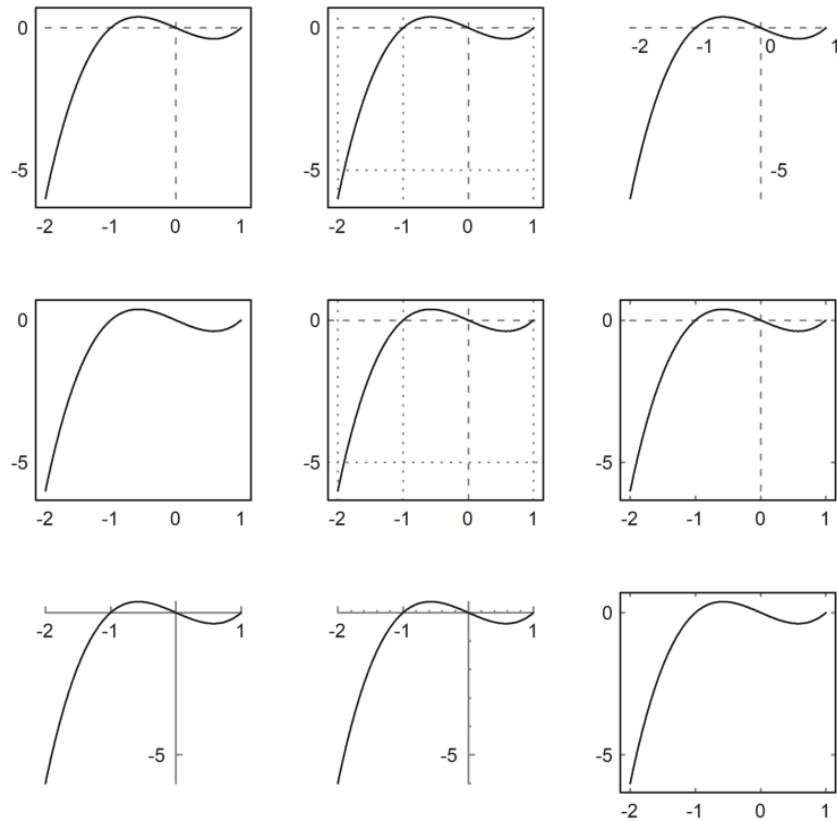
Untuk membagi jendela menjadi beberapa bagian plot, digunakan perintah `figure()`. Sebagai contoh, pemanggilan perintah untuk memplot fungsi x^1 hingga x^4 akan menghasilkan empat bagian dalam satu jendela. Perintah `figure(0)` digunakan untuk mengatur ulang jendela ke kondisi baku (default).

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



Dalam `plot2d()`, tersedia beragam gaya alternatif yang dapat digunakan dengan opsi `grid = x`. Untuk memberikan gambaran menyeluruh, berbagai gaya grid dapat ditampilkan sekaligus dalam satu gambar (lihat penggunaan perintah `figure()`). Gaya `grid = 0` tidak termasuk dalam tampilan. Pengaturan ini meniadakan grid maupun bingkai (frame) pada plot.

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

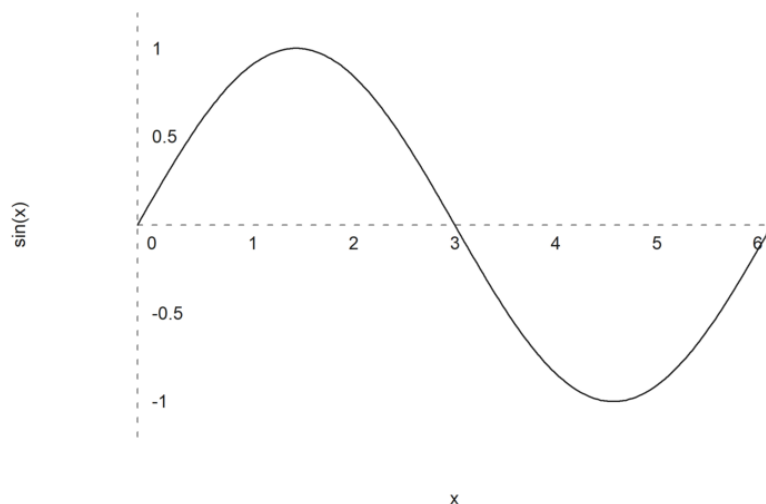



Apabila argumen pada `plot2d()` berupa sebuah ekspresi yang diikuti oleh empat bilangan, maka bilangan-bilangan tersebut akan merepresentasikan rentang sumbu-x dan sumbu-y pada plot.

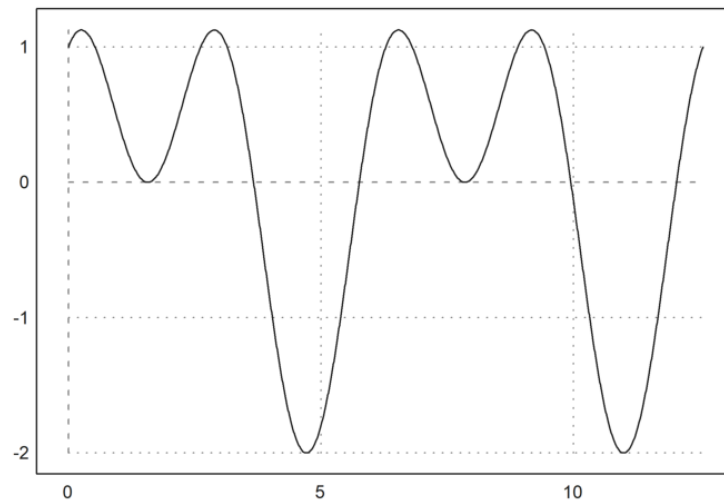
Sebagai alternatif, parameter rentang dapat pula ditetapkan melalui penugasan variabel, misalnya `a=...`, `b=...`, `c=...`, dan `d=...`.

Pada contoh berikut, dilakukan modifikasi dengan cara, mengubah gaya grid, menambahkan label pada plot, serta menggunakan label vertikal untuk sumbu-y.

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



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

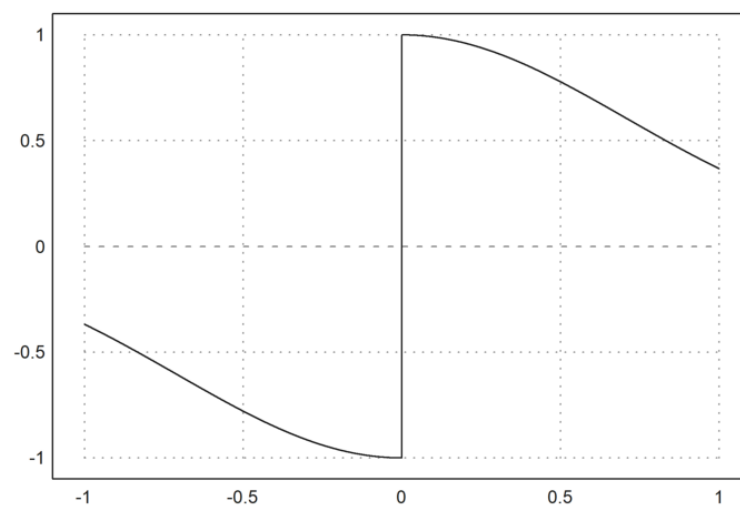


Gambar yang dihasilkan dengan menyisipkan plot ke dalam jendela teks secara otomatis disimpan pada direktori yang sama dengan notebook. Secara baku, gambar tersebut ditempatkan dalam sebuah subdirektori bernama "images". File gambar ini juga digunakan dalam proses ekspor ke format HTML.

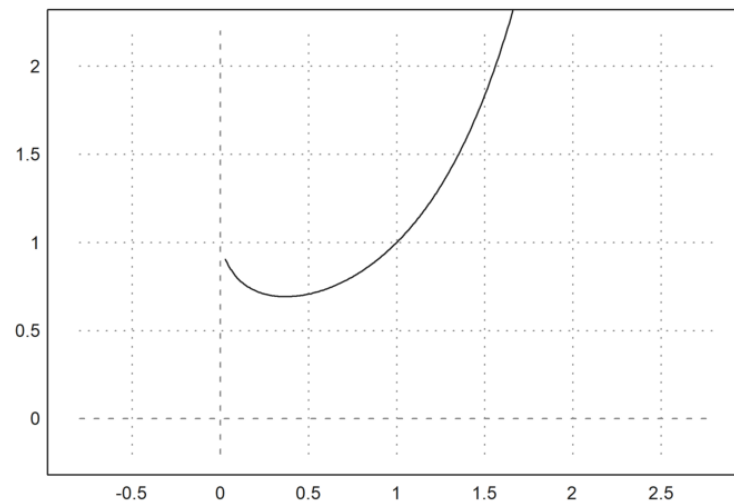
Setiap gambar dapat disalin ke clipboard dengan cara menandainya, kemudian menekan kombinasi tombol Ctrl + C. Selain itu, grafik yang sedang aktif juga dapat diekspor melalui fungsi yang tersedia pada menu File.

Ekspresi atau fungsi pada plot2d dievaluasi secara adaptif. Untuk meningkatkan kecepatan, mode evaluasi adaptif dapat dinonaktifkan dengan menggunakan opsi <adaptive, serta jumlah subinterval dapat ditentukan dengan parameter n=.... Namun, pengaturan ini biasanya hanya diperlukan pada kasus tertentu.

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

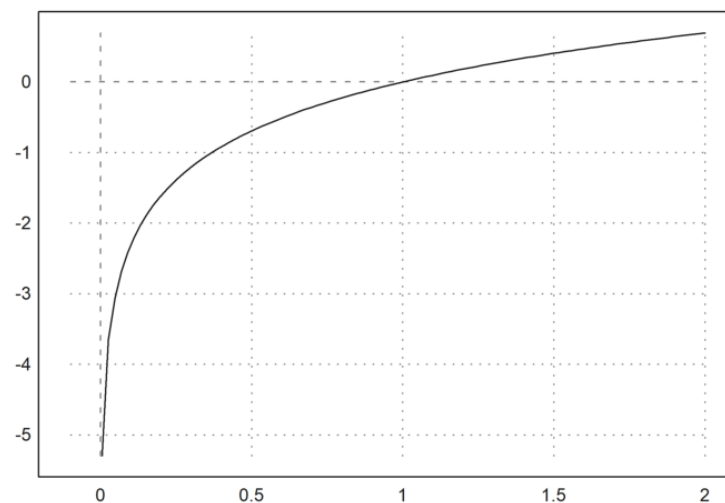


```
>plot2d("x^x",r=1.2,cx=1,cy=1):
```



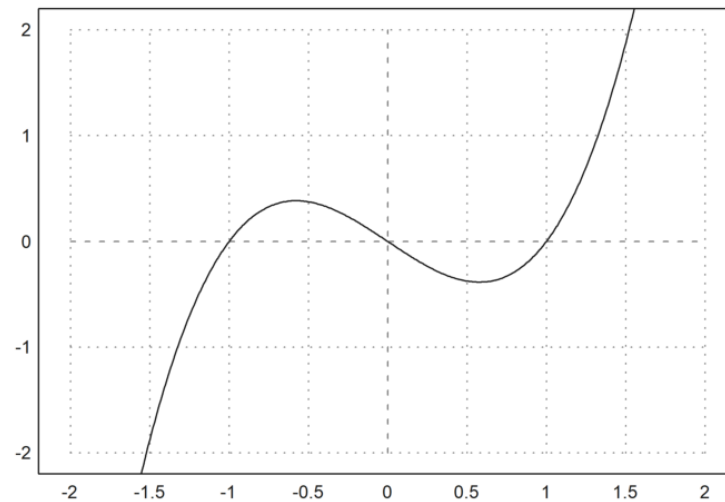
Perlu diperhatikan bahwa fungsi x^x tidak terdefinisi untuk $x=0$. Fungsi `plot2d` secara otomatis mengenali kesalahan ini, dan akan memulai proses plotting segera setelah fungsi berada dalam daerah definisinya. Mekanisme tersebut berlaku untuk semua fungsi yang menghasilkan nilai NAN (Not a Number) ketika dievaluasi di luar daerah definisinya.

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

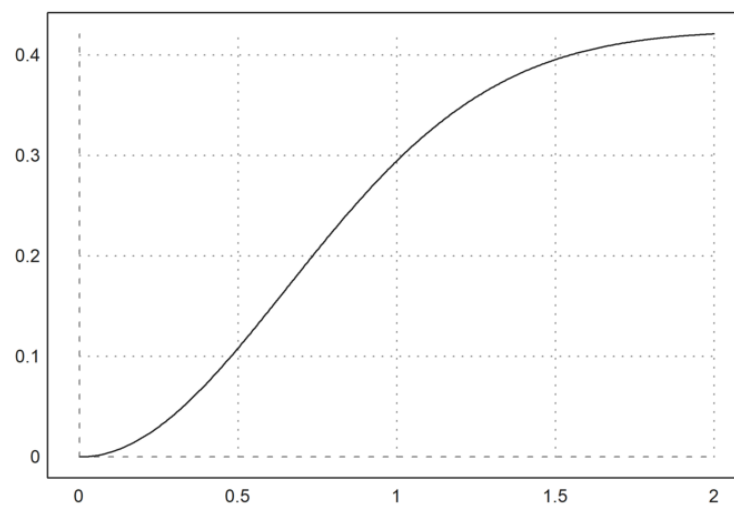


Parameter `square = true` (atau `>square`) digunakan untuk secara otomatis menyesuaikan rentang sumbu-y sehingga menghasilkan jendela plot berbentuk persegi. Perlu dicatat bahwa, secara bawaan (default), Euler sudah menggunakan ruang berbentuk persegi di dalam jendela plot.

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

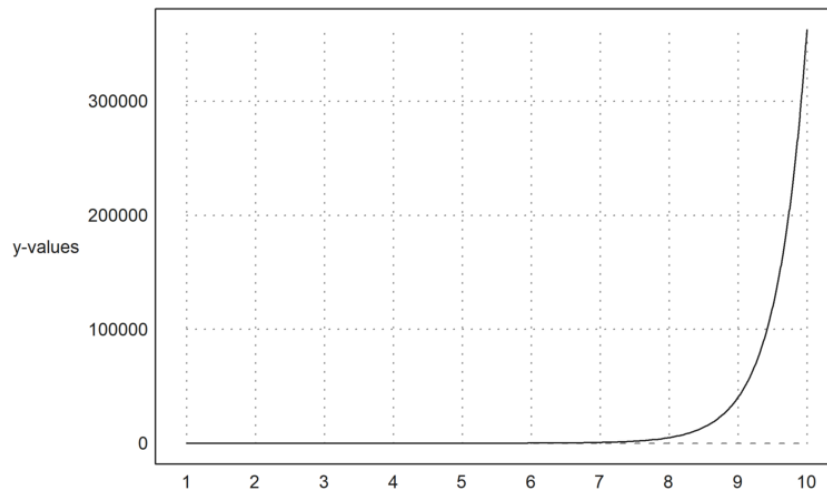


```
>plot2d(''integrate("sin(x)*exp(-x^2)",0,x)'',0,2): // plot integral
```



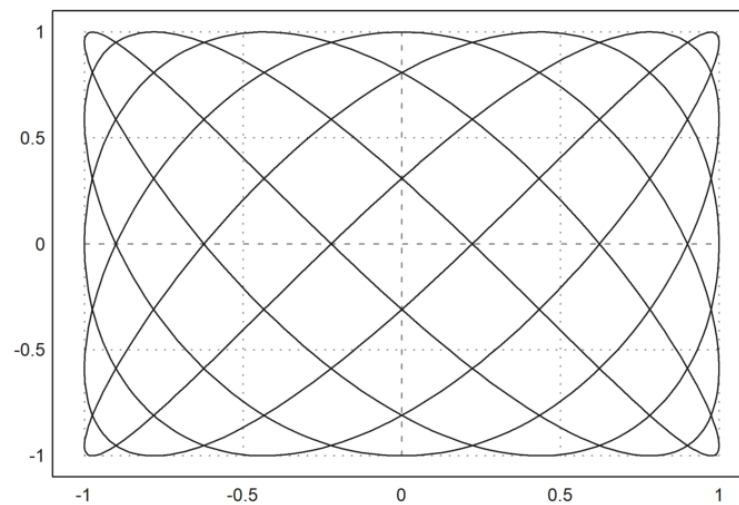
Apabila anda memerlukan lebih banyak ruang untuk label sumbu y, gunakan `shrinkwindow()` dengan `smaller` parameter, atau tetapkan nilai positif pada “`smaller`” di dalam `plot2d()`

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

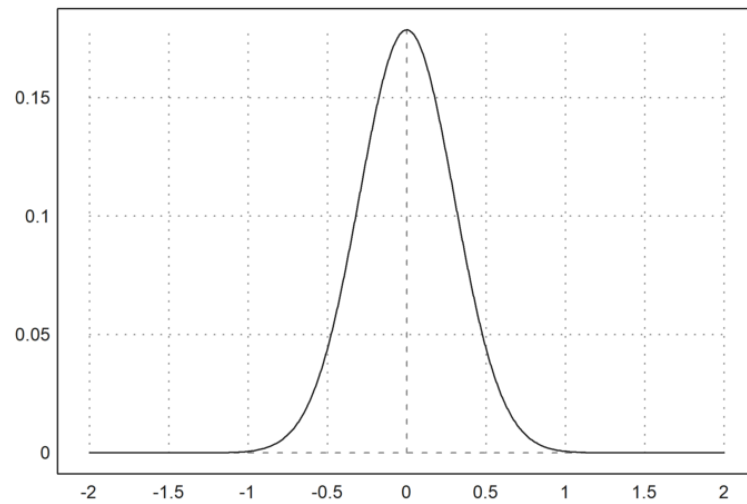


Ekspresi simbolik juga dapat digunakan karena tersimpan dalam bentuk string sederhana.

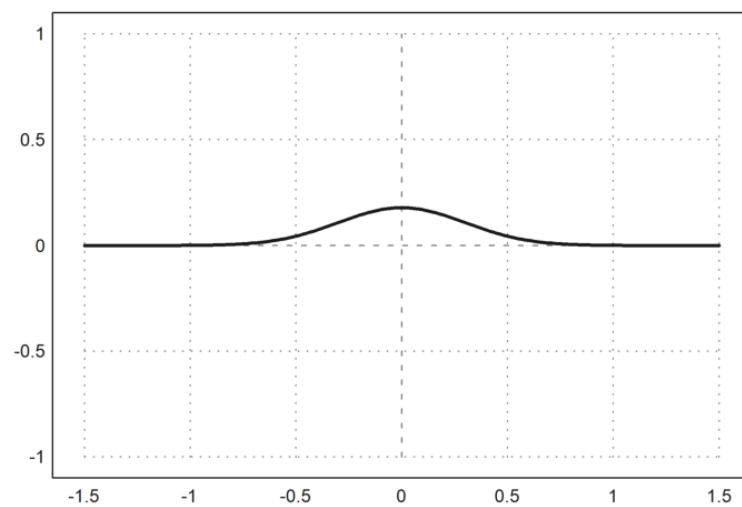
```
>x=linspace(0,2pi,1000); plot2d(sin(5x),cos(7x)):
```



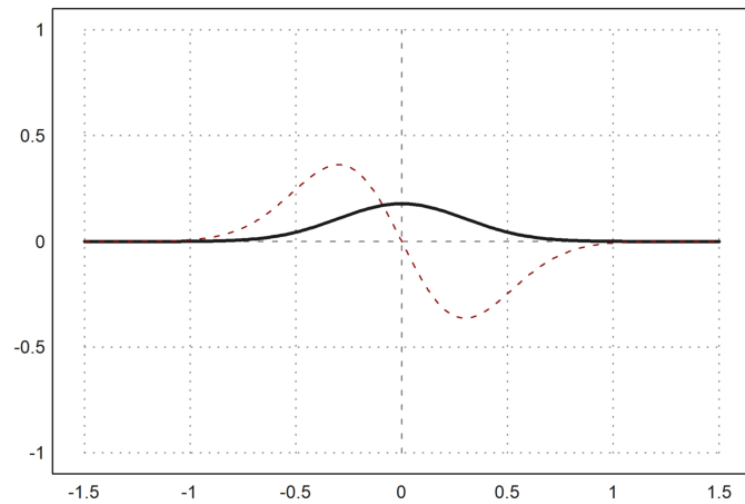
```
>a:=5.6; expr &= exp(-a*x^2)/a; // define expression
>plot2d(expr,-2,2): // plot from -2 to 2
```



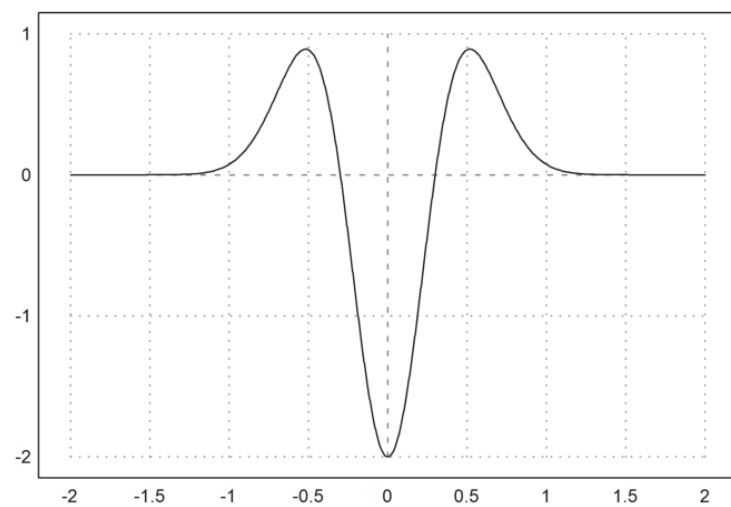
```
>plot2d(expr,r=1,thickness=2): // plot in a square around (0,0)
```



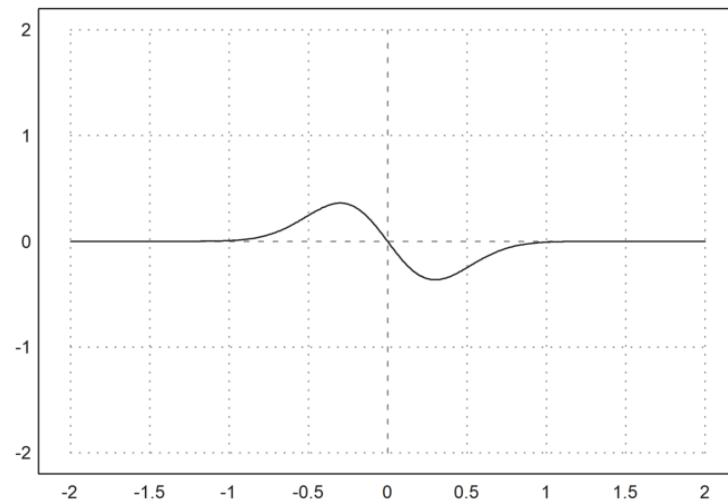
```
>plot2d(&diff(expr,x),>add,style="--",color=red): // add another plot
```



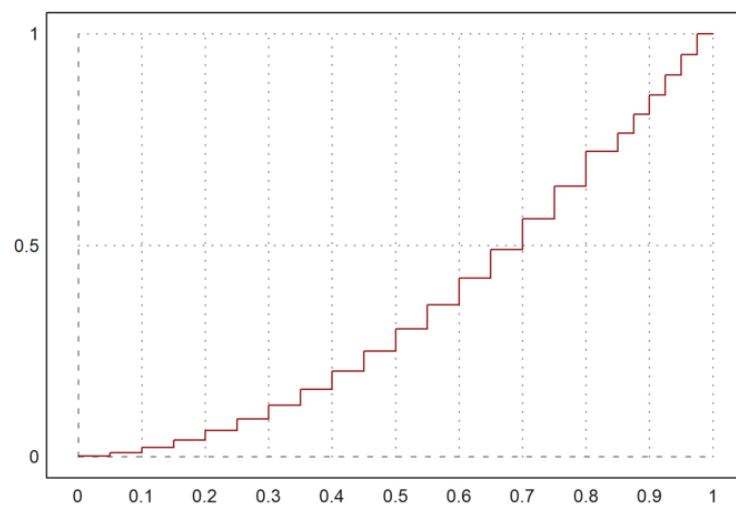
```
>plot2d(&diff(expr,x,2),a=-2,b=2,c=-2,d=1): // plot in rectangle
```



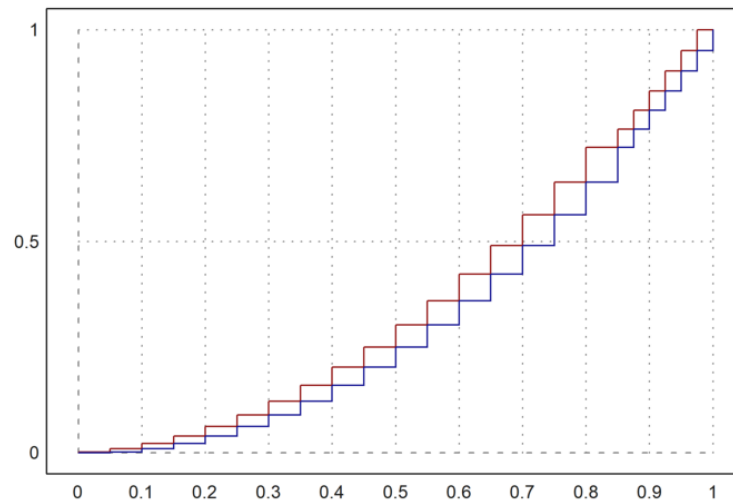
```
>plot2d(&diff(expr,x),a=-2,b=2,>square): // keep plot square
```



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



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

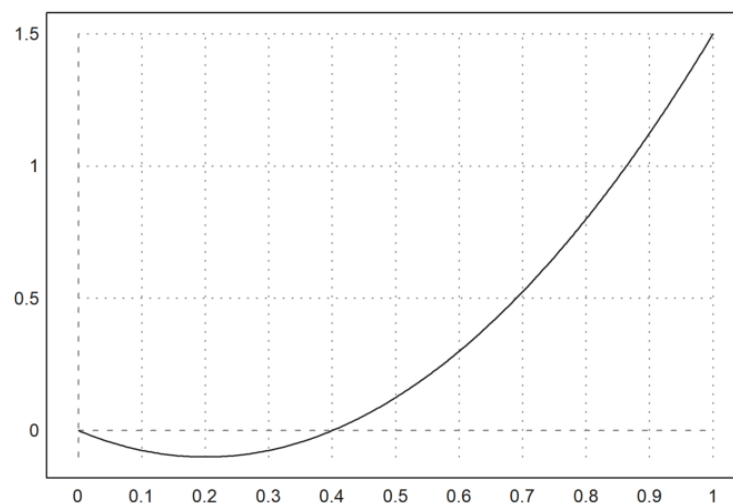



Fungsi dalam Satu Parameter

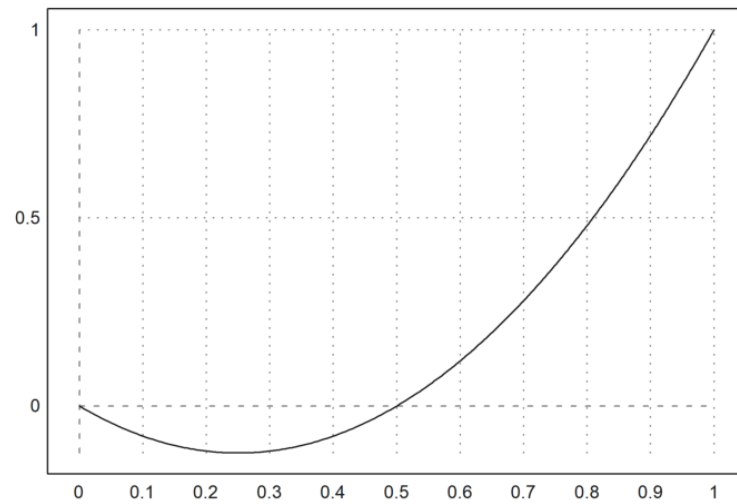
Fungsi plotting yang paling penting untuk grafik bidang datar adalah `plot2d()`. Fungsi ini diimplementasikan dalam bahasa Euler pada berkas “plot.e”, yang secara otomatis dimuat ketika program dijalankan.

Berikut ditampilkan beberapa contoh penggunaan dengan sebuah fungsi. Seperti biasa dalam EMT, fungsi-fungsi yang dapat bekerja pada fungsi lain maupun ekspresi memungkinkan penyertaan parameter tambahan (selain variabel ??). Parameter ini dapat diberikan melalui parameter dengan tanda titik koma (semicolon parameters) atau dengan menggunakan call collection.

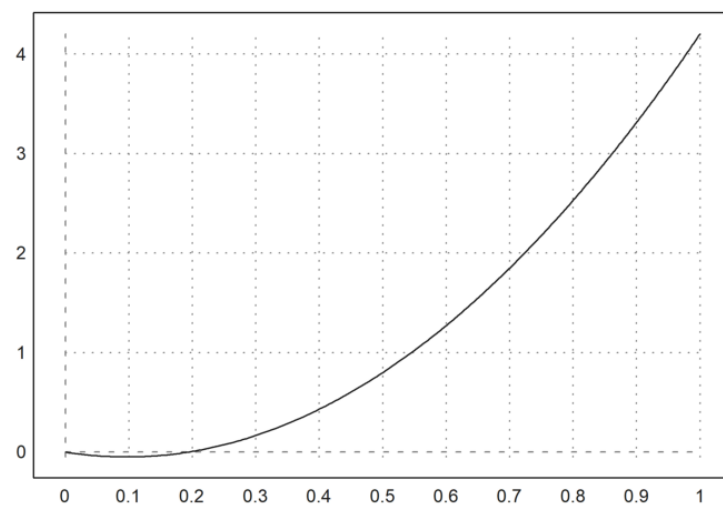
```
>function f(x,a) := x^2/a+a*x^2-x; // mendefinisikan fungsi
>a=0.5; plot2d("f",0,1;a): // a=0.5
```



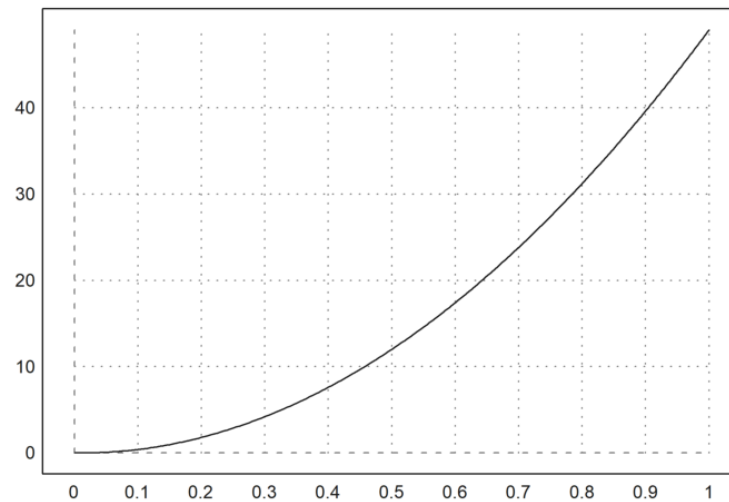
```
>plot2d("f",0,1;1): // a=1
```



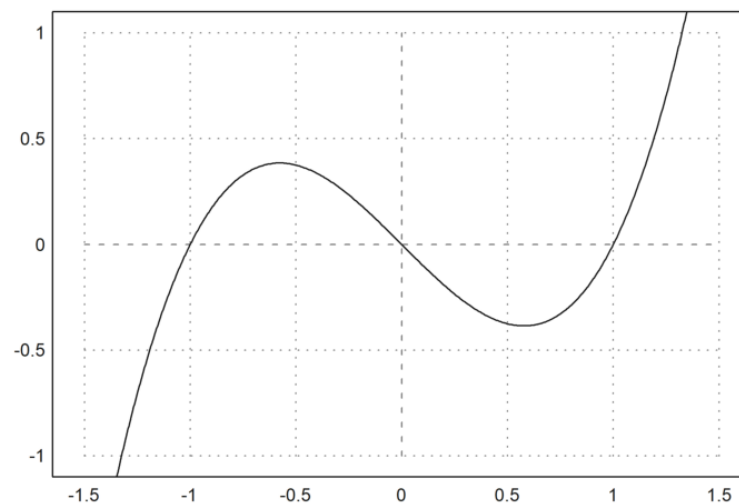
```
>plot2d({{"f",0.2}},0,1): // plot with a=0.2
```



```
>plot2d({{"f(x,b)",b=50}},0,1): // b=50
```



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



Ringkasan fungsi yang diterima dalam plot2d():

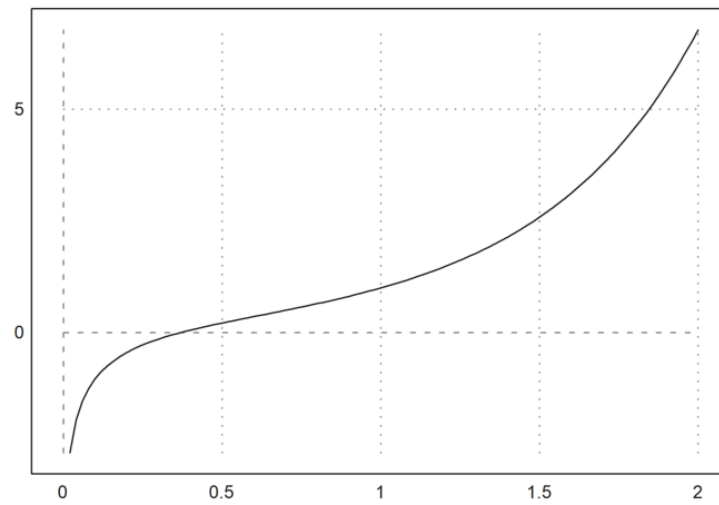
- ekspresi atau ekspresi simbolik dalam variabel ??.
- fungsi atau fungsi simbolik yang dipanggil dengan nama, misalnya "f".
- fungsi simbolik yang cukup dituliskan dengan namanya saja, misalnya f.

Selain itu, plot2d() juga mendukung fungsi simbolik. Untuk jenis ini, penggunaan nama fungsi saja sudah mencukupi tanpa tanda kutip.

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

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

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

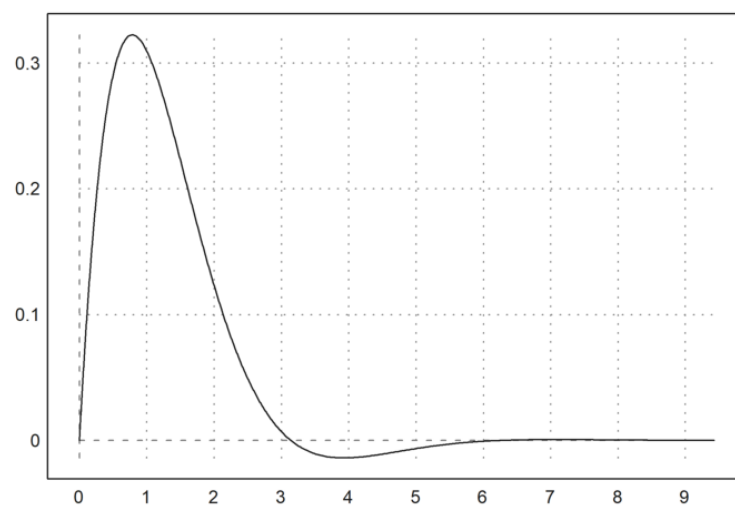


Tentu saja, untuk ekspresi maupun ekspresi simbolik, cukup dengan menyertakan nama variabel saja agar dapat dipetakan (diplot) menggunakan plot2d().

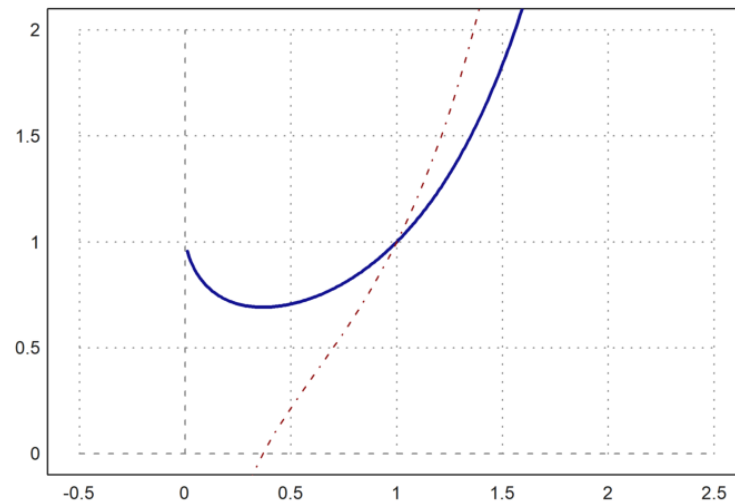
```
>expr &= sin(x)*exp(-x)
```

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

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



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



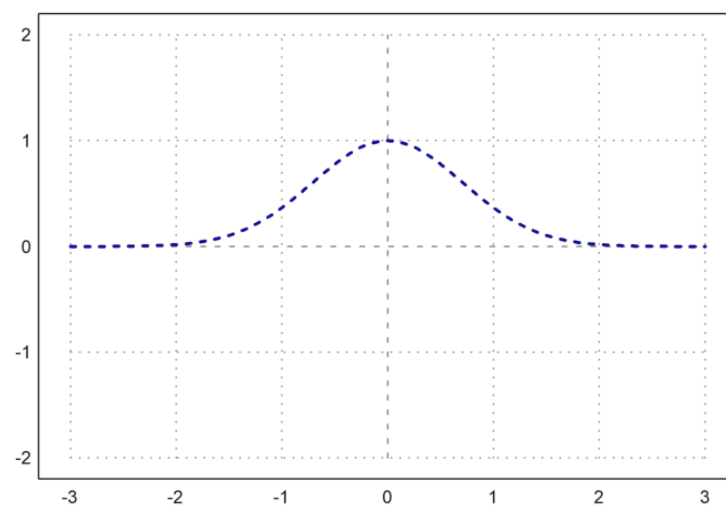
Untuk pengaturan gaya garis (line style), tersedia beberapa opsi sebagai berikut:

- pilih "-", "--", "-.-", ":", ":", "-.-", "-.-".
- warna: lihat bawah
- ketebalan: default adalah 1

Warna dapat default atau RGB

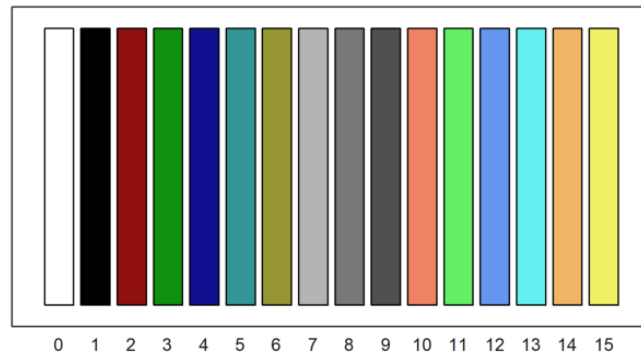
- Indeks warna default adalah 0..15
- Konstanta warna: white, black, red, green, blue, cyan, olive, lightgray, gray, darkgray, orange, lightgreen, turquoise, lightblue, lightorange, yellow.
- RGB: parameter bilangan real [0,1]

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



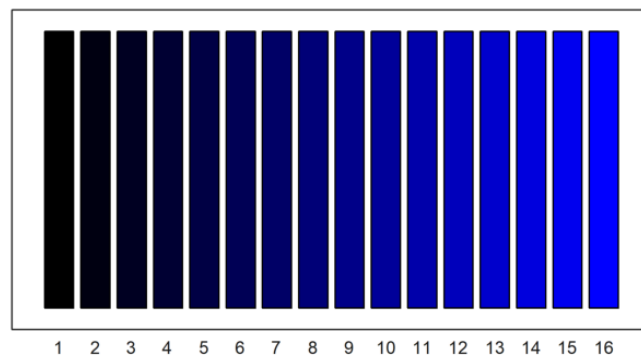
Berikut merupakan tampilan warna-warni baku (predefined colors) yang tersedia dalam EMT.

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



Anda dapat memilih warna lainnya.

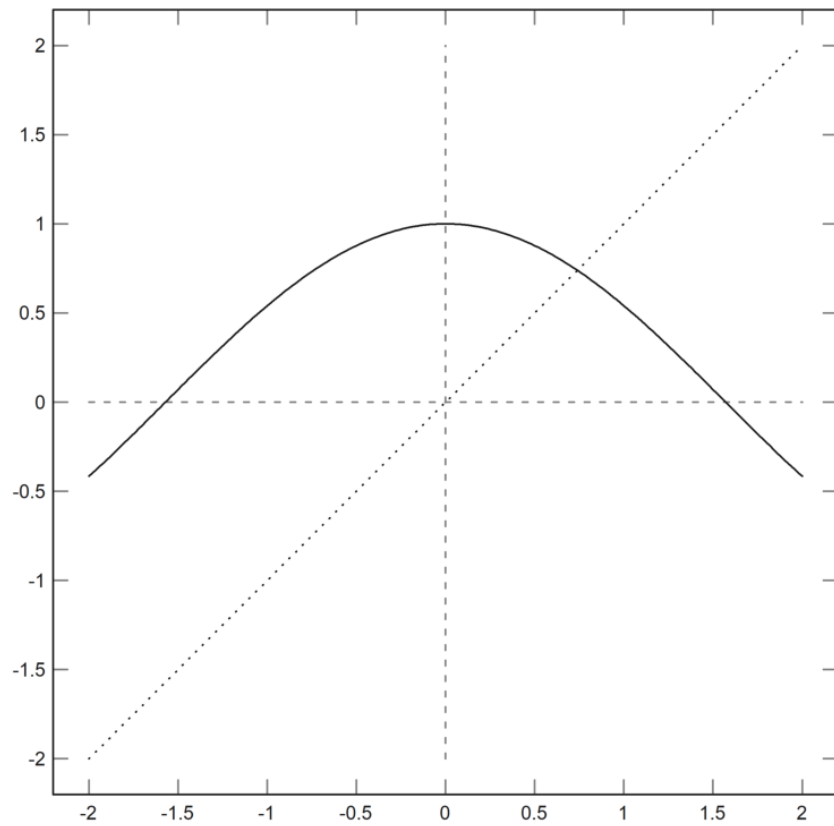
```
>columnsplot(ones(1,16),grid=0,color=rgb(0,0,linspace(0,1,15))):
```



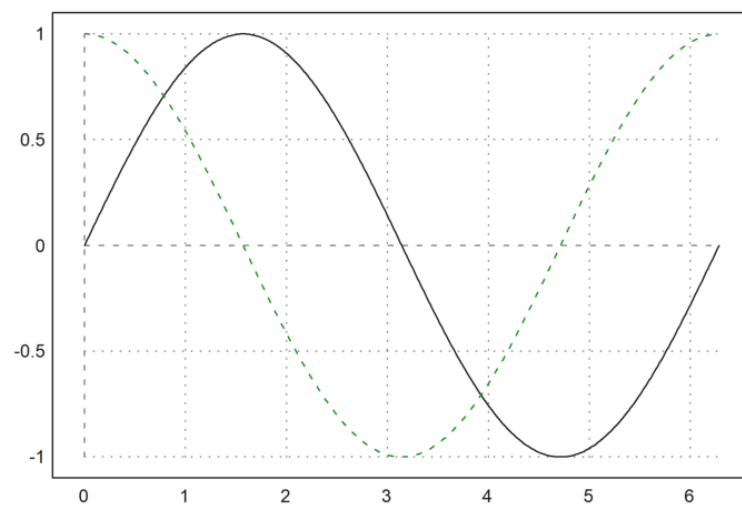
Menggambar Beberapa Kurva pada bidang koordinat yang sama

Plot beberapa fungsi sekaligus (multiple functions) dalam satu jendela dapat dilakukan dengan beberapa cara. Salah satu metode yang tersedia adalah menggunakan opsi `>add` pada setiap pemanggilan `plot2d`, kecuali pada pemanggilan pertama. Fitur ini telah digunakan pada contoh-contoh sebelumnya.

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

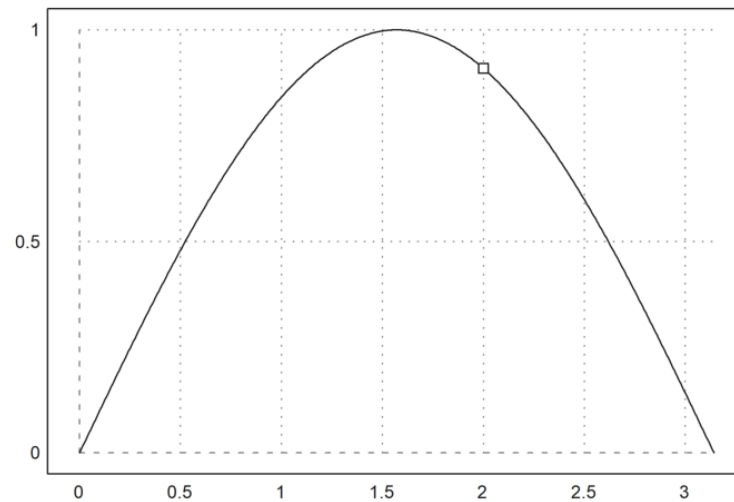


```
>aspect(1.5); plot2d("sin(x)",0,2pi); plot2d("cos(x)",color=green,style="--",>add):
```



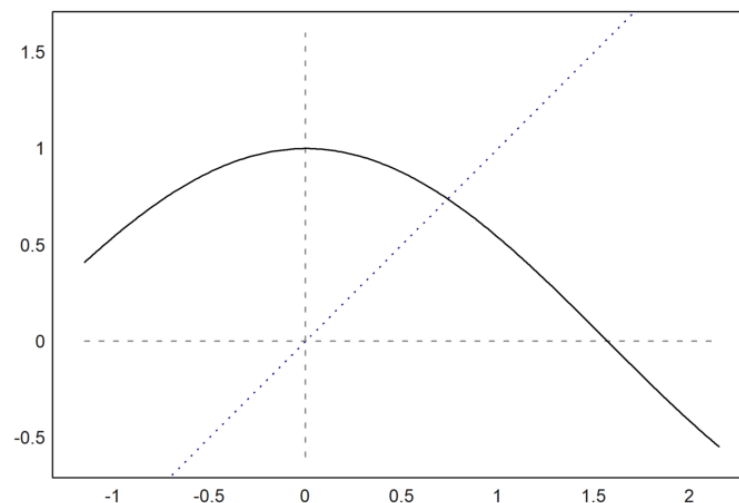
Salah satu kegunaan `>add` adalah untuk menambahkan titik pada kurva.

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

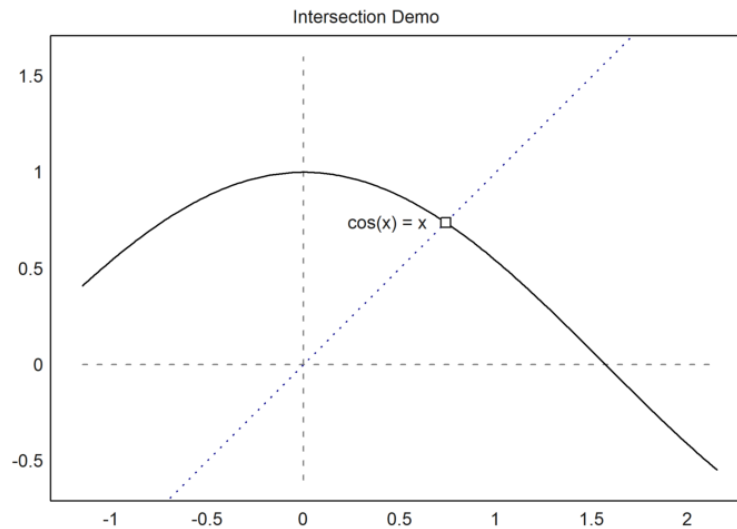


Pada contoh ini, ditambahkan titik potong (intersection point) beserta label pada posisi "cl" (center left), kemudian hasilnya disisipkan ke dalam notebook. Selain itu, ditambahkan pula judul (title) pada plot untuk memperjelas informasi yang ditampilkan.

```
>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):
```

Pada demonstrasi berikut, dilakukan pemetaan fungsi $\text{sinc}(x) = \sin(x)/x$ beserta ekspansi Taylor orde ke-8 dan orde ke-16. Ekspansi ini dihitung menggunakan Maxima melalui ekspresi simbolik.

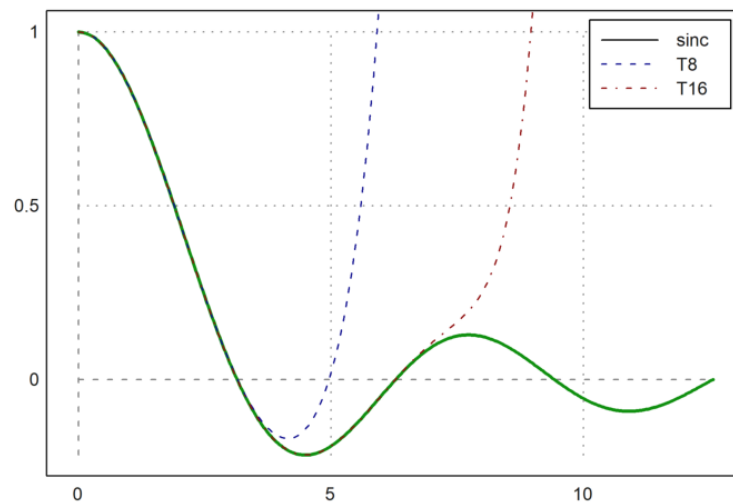
Plot dihasilkan dengan perintah multi-baris yang berisi tiga pemanggilan `plot2d()`. Pada pemanggilan kedua dan ketiga, digunakan flag `>add` sehingga plot baru digabungkan dengan plot sebelumnya serta menggunakan rentang (range) yang sama.

Kita juga menambahkan kotak label untuk menjelaskan mengenai fungsi-fungsinya.

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

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

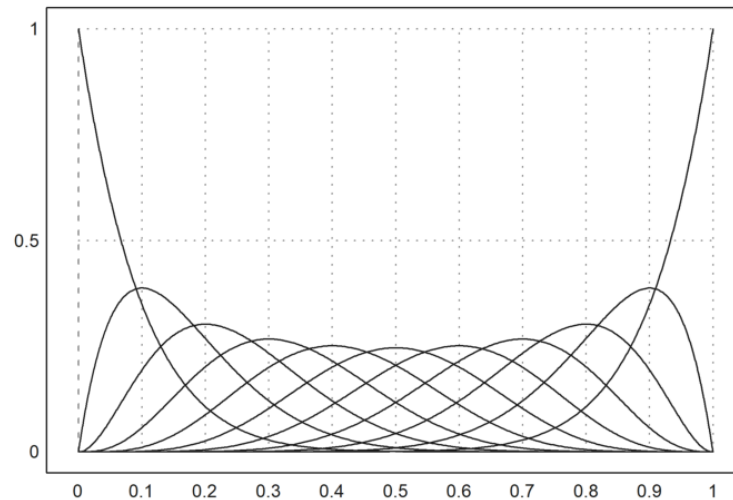
```
>plot2d("sinc(x)",0,4pi,color=green,thickness=2); ...
> plot2d(&taylor(sin(x)/x,x,0,8),>add,color=blue,style="--"); ...
> plot2d(&taylor(sin(x)/x,x,0,16),>add,color=red,style="-.-"); ...
> labelbox(["sinc","T8","T16"],styles=["-", "--", "-.-"], ...
> colors=[black,blue,red]):
```



Pada contoh berikut, dilakukan proses pembuatan dan pemetaan Polinomial Bernstein (Bernstein-Polynomials).

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

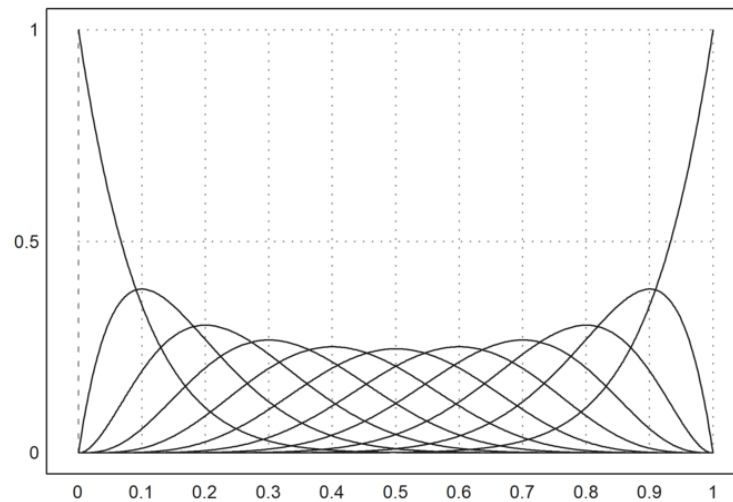
```
>plot2d("(1-x)^10",0,1); // plot first function  
>for i=1 to 10; plot2d("bin(10,i)*x^i*(1-x)^(10-i)",>add); end;  
>insimg;
```



Metode kedua untuk melakukan plotting adalah dengan menggunakan sepasang matriks nilai-x dan matriks nilai-y yang memiliki ukuran sama. Dalam contoh ini, dibentuk sebuah matriks nilai dengan satu Polinomial Bernstein pada setiap barisnya.

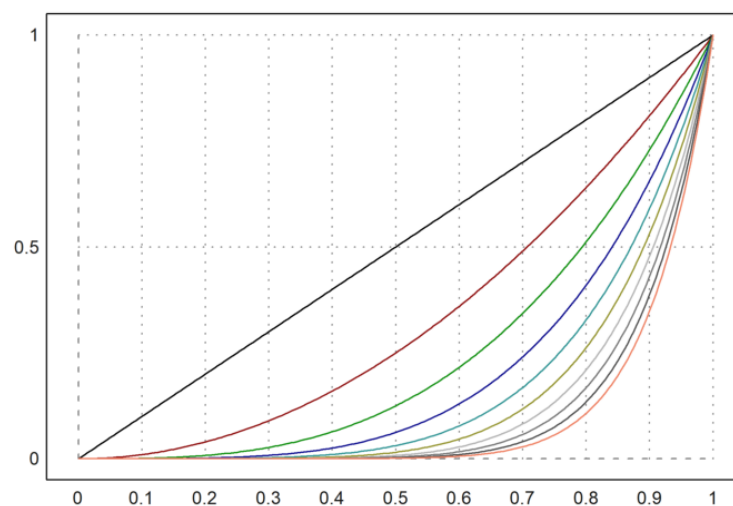
Untuk tujuan tersebut, digunakan vektor kolom i secara sederhana. Untuk memahami lebih rinci mengenai konstruksi matriks dan operasinya, dapat merujuk pada bagian pengantar bahasa matriks.

```
>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):
```



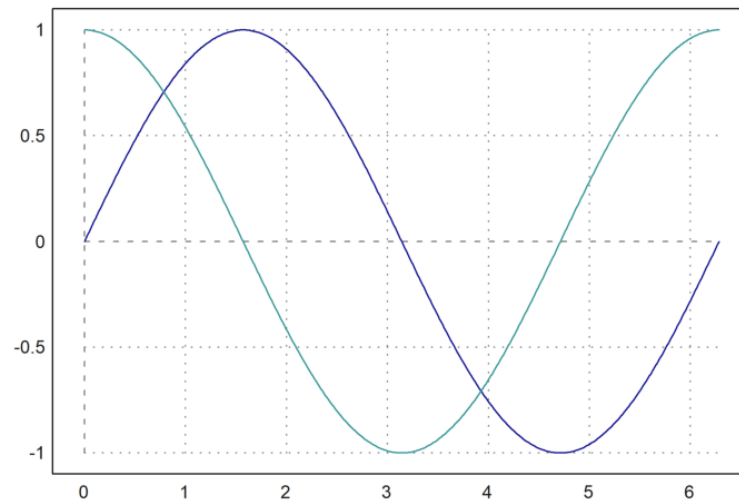
Parameter warna juga bisa merupakan sebuah vektor. Setiap warna digunakan untuk setiap baris matriks.

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

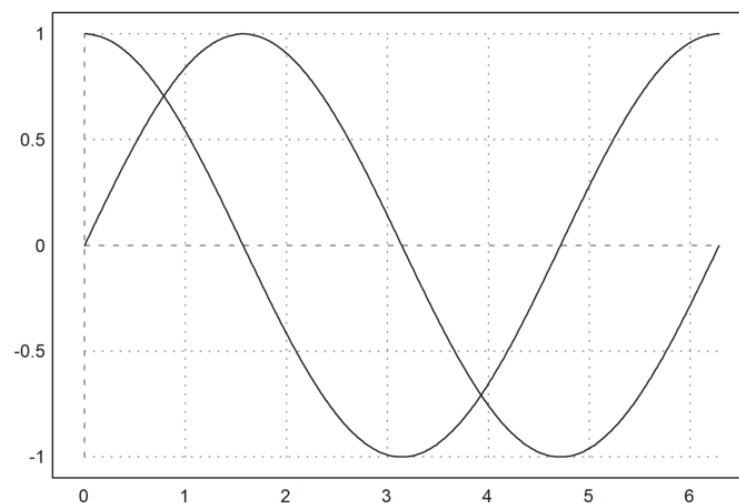


Metode lain adalah menggunakan ekspresi vektor. Anda dapat menggunakan array warna, array style, dan array ketebalan pada panjang yang sama.

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



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



Kita dapat memperoleh sebuah vektor dari Maxima dengan menggunakan perintah makelist() dan mxm2str().

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

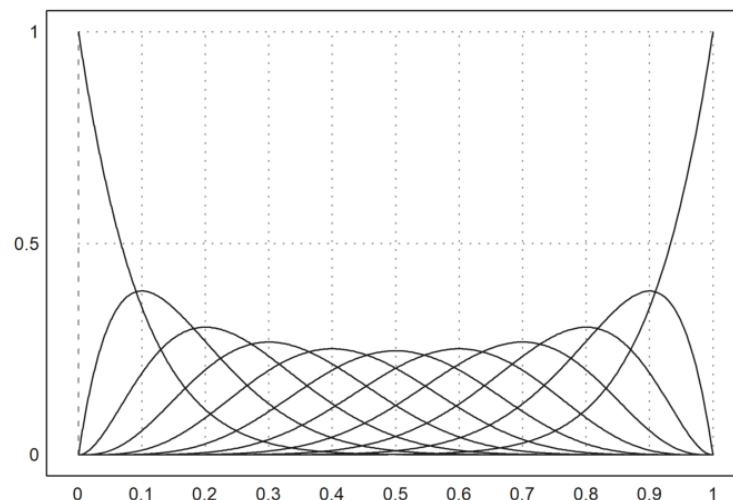
```

      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
```

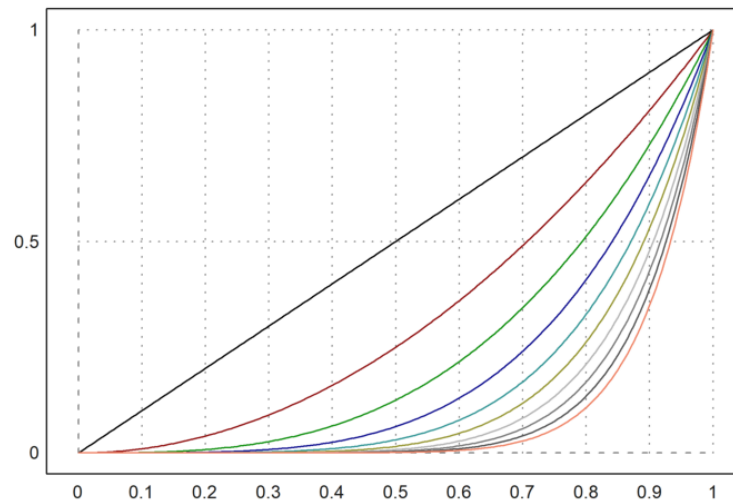


Alternatif lain adalah menggunakan bahasa matriks Euler.

Jika suatu ekspresi menghasilkan matriks fungsi, dengan satu fungsi pada 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):
```

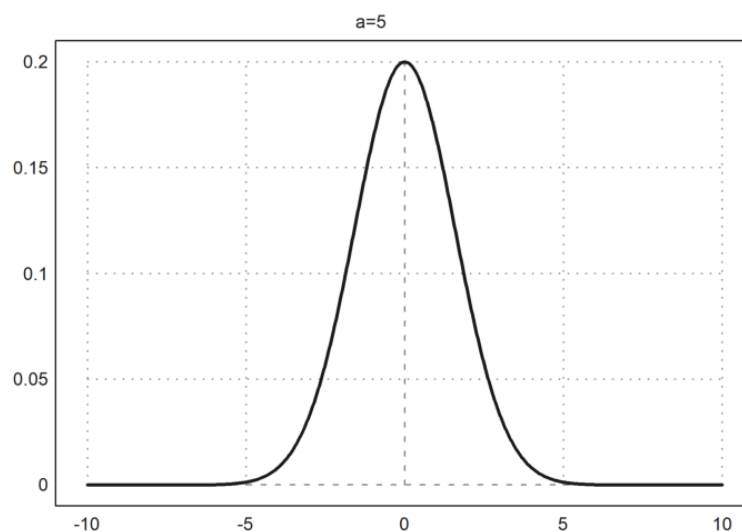


Ekspresi dan fungsi satu baris dapat melihat variabel global.

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

Pastikan 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

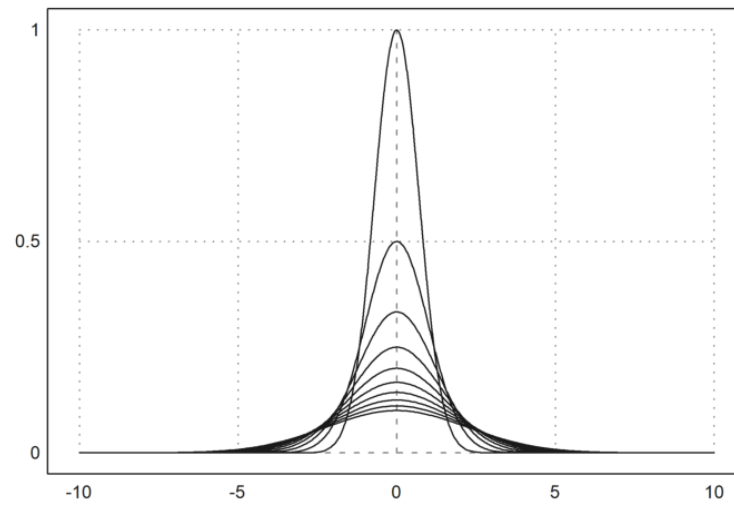
```
>function f(x,a) := 1/a*exp(-x^2/a); ...
>plot2d("f",-10,10;5,thickness=2,title="a=5") :
```



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.

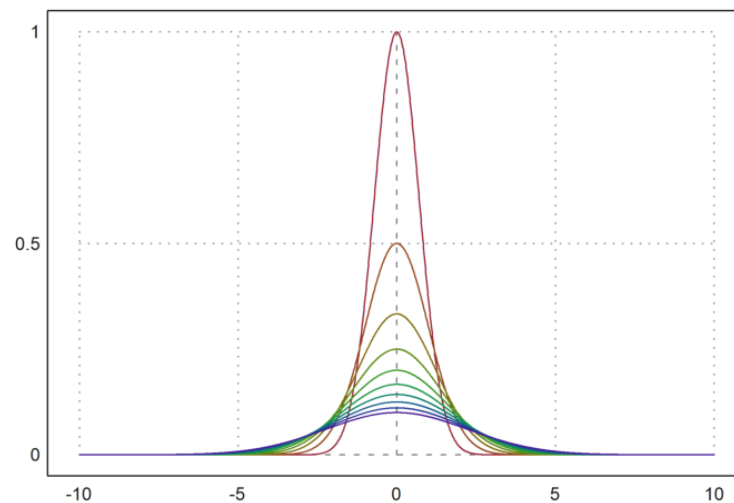
Dalam contoh berikut, kami menggunakan loop untuk memplot beberapa fungsi (lihat tutorial tentang pemrograman untuk loop)

```
>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)):
```



Labels Teks

Dekorasi sederhana dapat berupa

- judul dengan

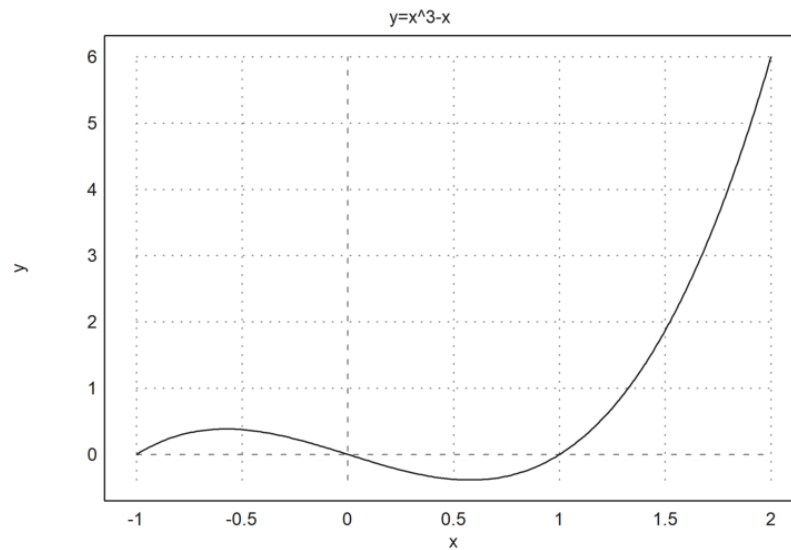
`title="..."`

- label x dan y dengan `xl="...", yl="..."`

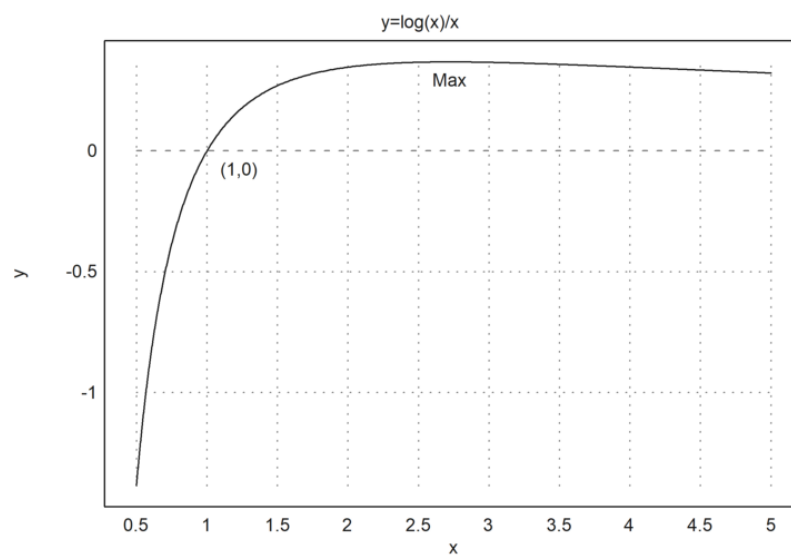
- label teks lain dengan `label("...",x,y)`

Perintah label akan memplot 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") :
```

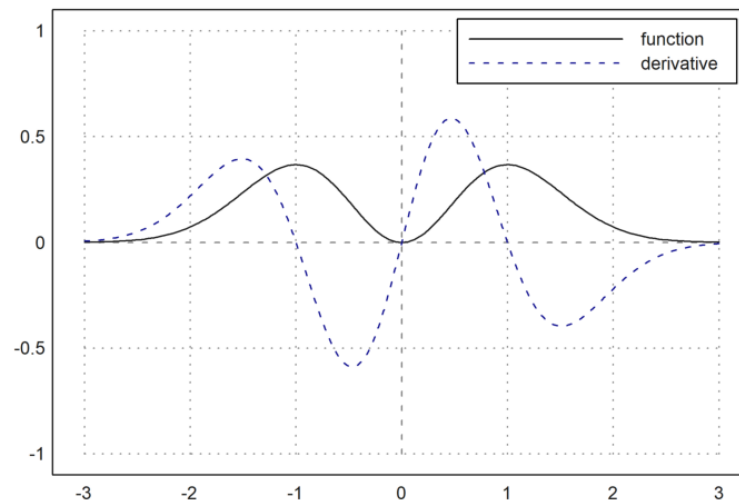


```
>expr := "log(x)/x"; ...  
> plot2d(expr,0.5,5,title="y="+expr,xl="x",yl="y"); ...  
> label("(1,0)",1,0); label("Max",E,expr(E),pos="lc") :
```



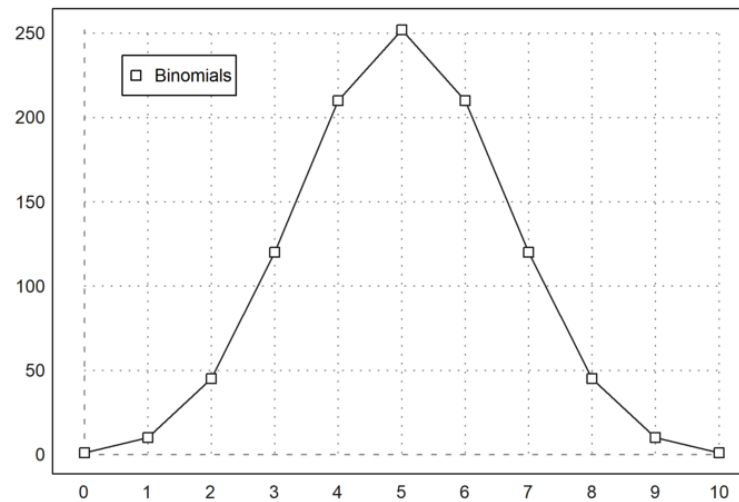
Ada juga fungsi `labelbox()`, yang dapat menampilkan fungsi dan teks. Fungsi ini menerima vektor string dan warna, satu item untuk setiap fungsi.

```
>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):
```



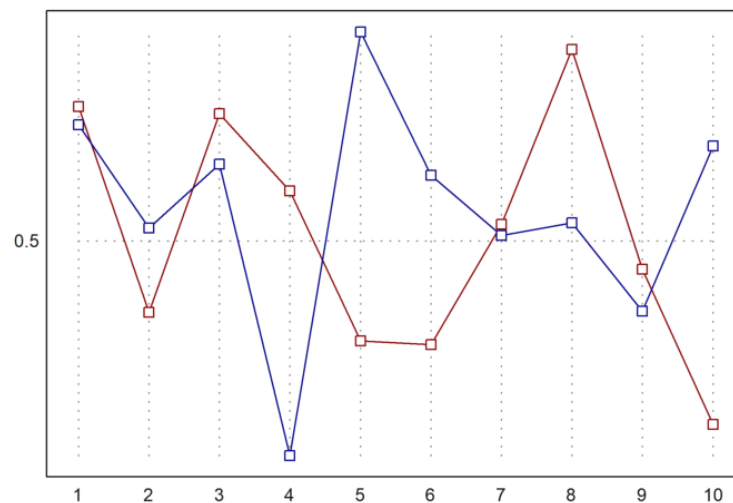
Kotak tersebut dijangkarkan di kanan atas secara default, tetapi `>left` menjangkarkannya di kiri atas. Anda dapat memindahkannya ke tempat mana pun yang Anda inginkan. Posisi jangkar adalah di pojok kanan atas kotak, dan angka-angkanya merupakan pecahan dari ukuran jendela grafis. Lebarnya otomatis.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

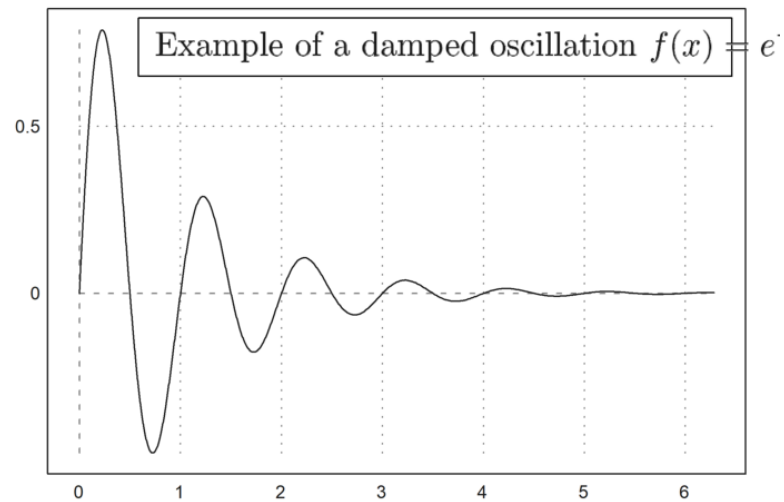
```
>statplot(1:10,random(2,10),color=[red,blue]):
```



Fitur serupa adalah fungsi `textbox()`.

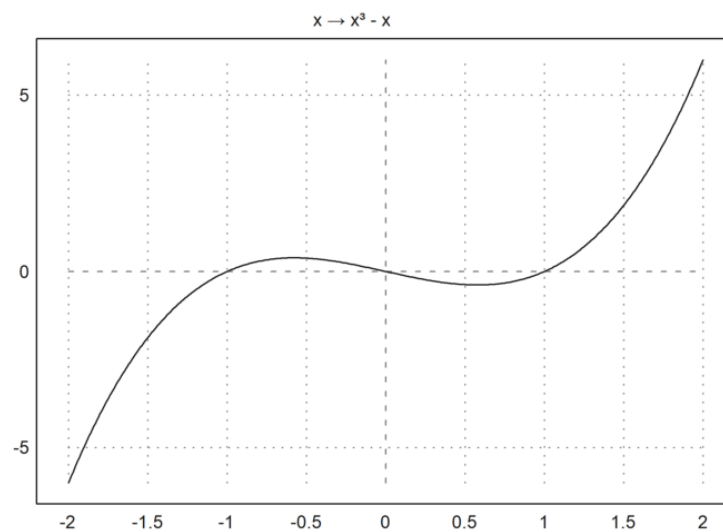
Lebar nya secara default adalah lebar maksimal baris teks. Namun, lebarnya juga dapat diatur oleh pengguna

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



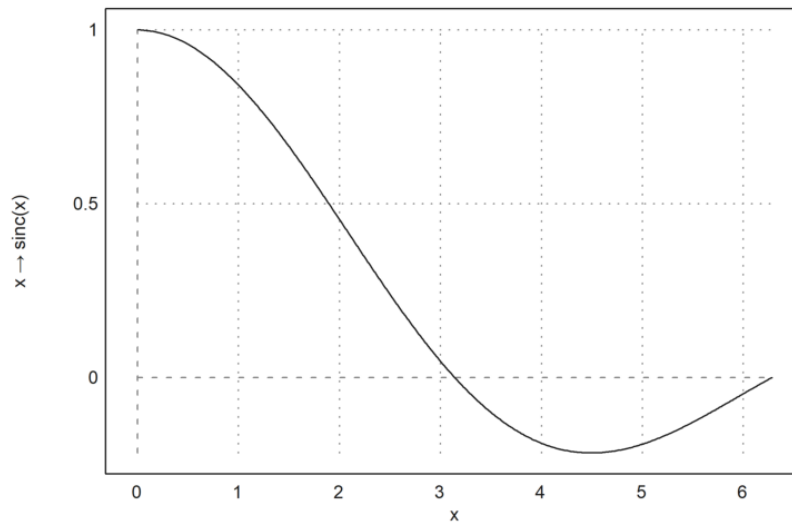
Label teks, judul, kotak label, dan teks lainnya dapat berisi string Unicode (lihat sintaksis EMT untuk informasi lebih lanjut tentang string Unicode).

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



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

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



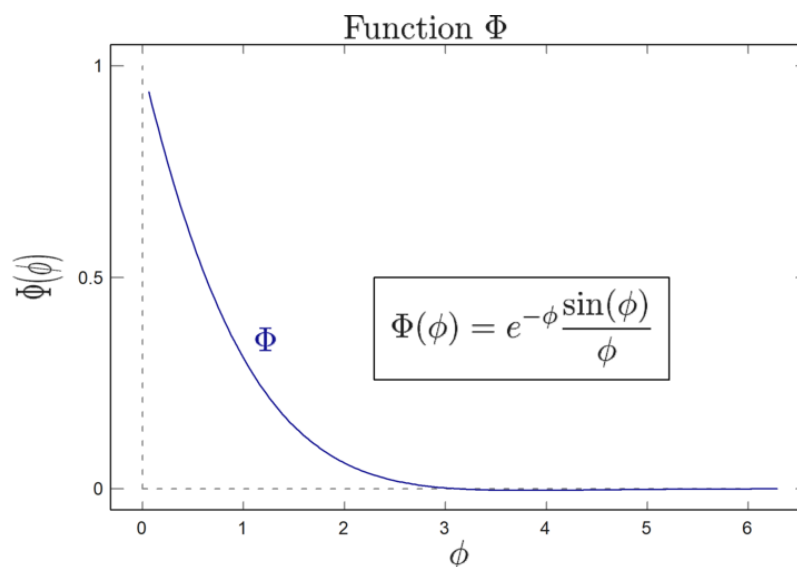
LaTeX

Anda juga dapat memplot rumus LaTeX jika 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 opsi.

Pada plot berikut, kami menggunakan LaTeX untuk label x dan y, label, kotak label, dan judul plot.

Perlu diketahui 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).

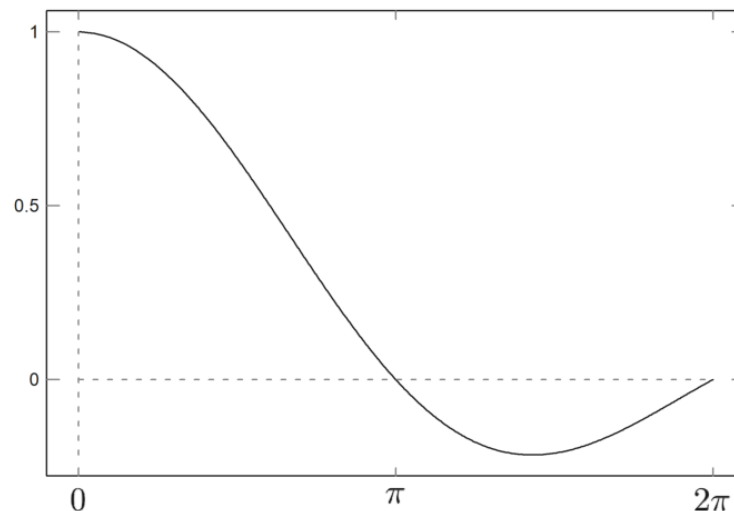
```
>plot2d("exp(-x)*sin(x)/x",a=0,b=2pi,c=0,d=1,grid=6,color=blue, ...
> title=latex("\text{Function } \Phi"), ...
> xl=latex("\phi"),yl=latex("\Phi(\phi)")); ...
>textbox( ...
> latex("\Phi(\phi) = e^{-\phi} \frac{\sin(\phi)}{\phi}"),x=0.8,y=0.5); ...
>label(latex("\Phi",color=blue),1,0.4):
```



Seringkali, kita menginginkan spasi 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, kami menggunakan tiga string LaTeX untuk label pada sumbu x dengan `xtick()`

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



Tentu saja fungsi juga dapat digunakan.

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

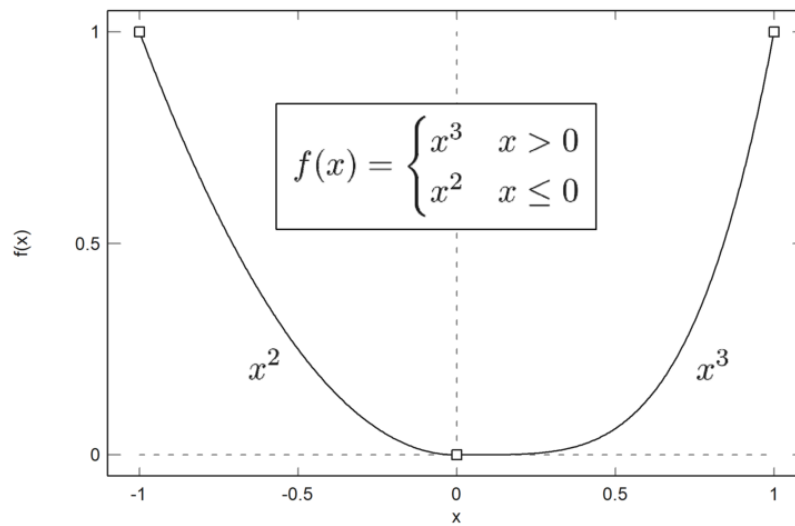
```
  if x>0 then return x^4
  else return x^2
endif
endfunction
```

Parameter "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. Kode ini digunakan untuk dua label dan satu kotak teks. Tentu saja, Anda hanya dapat menggunakan LaTeX jika LaTeX telah terpasang 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 \leq 0 \end{cases}"), ...
> x=0.7,y=0.2):
```



Interaksi Pengguna

Saat memplot fungsi atau ekspresi, parameter `>user` memungkinkan pengguna untuk memperbesar dan menggeser plot dengan tombol

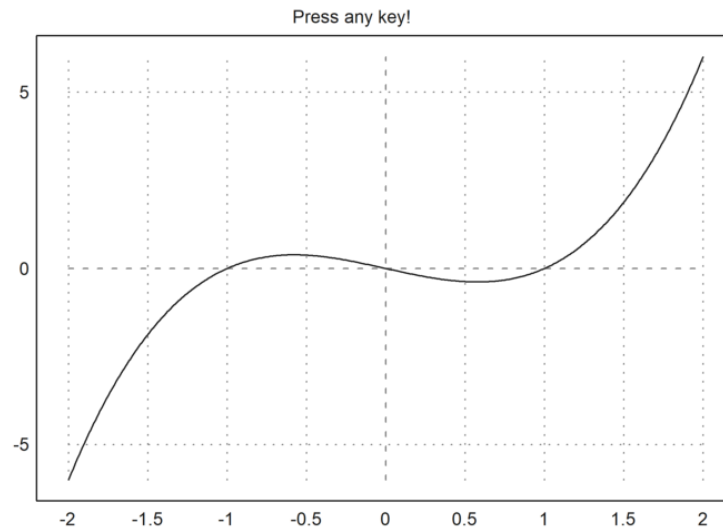
kursor atau tetikus. Pengguna dapat:

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

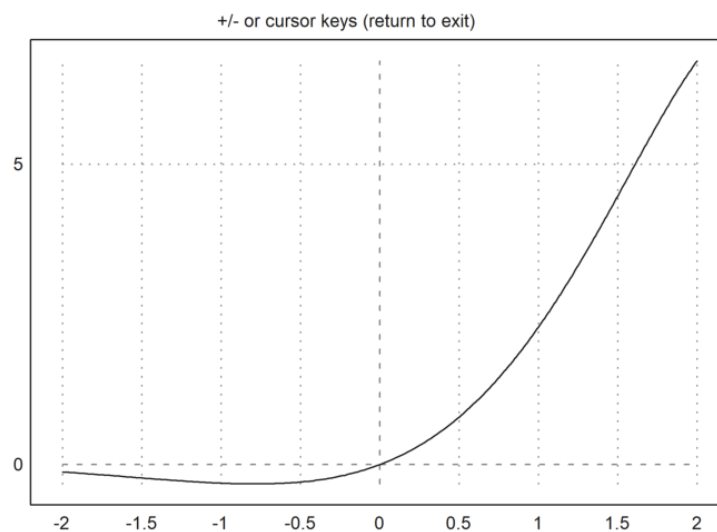
Tombol spasi akan mengatur ulang plot ke jendela plot semula.

Saat memetakan 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, ... // user=true untuk memanggil user
> title="+/- or cursor keys (return to exit)":
```



Berikut ini menunjukkan cara interaksi pengguna yang canggih (lihat tutorial tentang pemrograman untuk detailnya).

Fungsi bawaan `mousedrag()` menunggu kejadian mouse atau keyboard. Fungsi ini melaporkan gerakan mouse ke bawah, gerakan mouse ke atas, atau penekanan tombol. Fungsi `dragpoints()` memanfaatkan hal ini, dan memungkinkan pengguna menyeret titik manapun dalam plot.

Kita perlu fungsi plot terlebih dahulu. Sebagai contoh, kita menginterpolasi 5 titik dengan polinomial. Fungsi tersebut harus diplot 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);
```

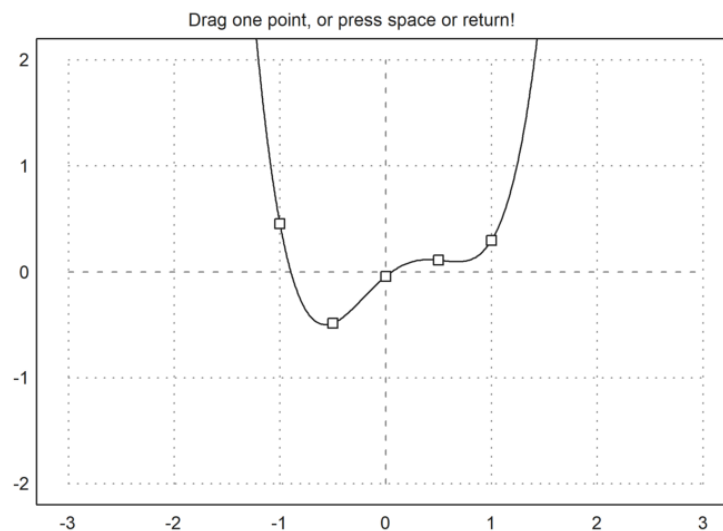
```

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 secara global. Sekarang kita buat beberapa nilai acak, dan biarkan pengguna menyeret titik-titiknya. manfaatkan hal ini, dan memungkinkan pengguna menyeret titik manapun dalam plot.

```
>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. Pertama kita butuh fungsi plot

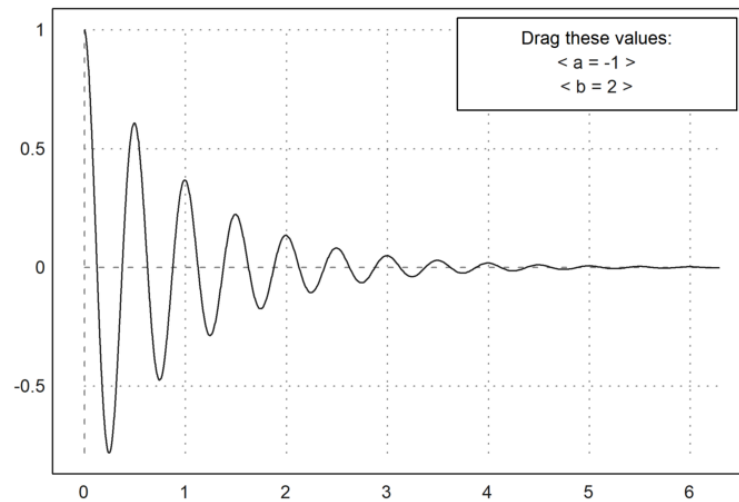
```
>function plotf([a,b]) := plot2d("exp(a*x)*cos(2pi*b*x)",0,2pi;a,b);
```

Kemudian kita perlu nama untuk parameter, nilai awal, dan matriks rentang mx2, secara opsional baris judul. Terdapat slider interkatif yang dapat mengatur nilai sesuai keinginan pengguna. Fungsi dragvalues() menyediakannya.

```

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

```

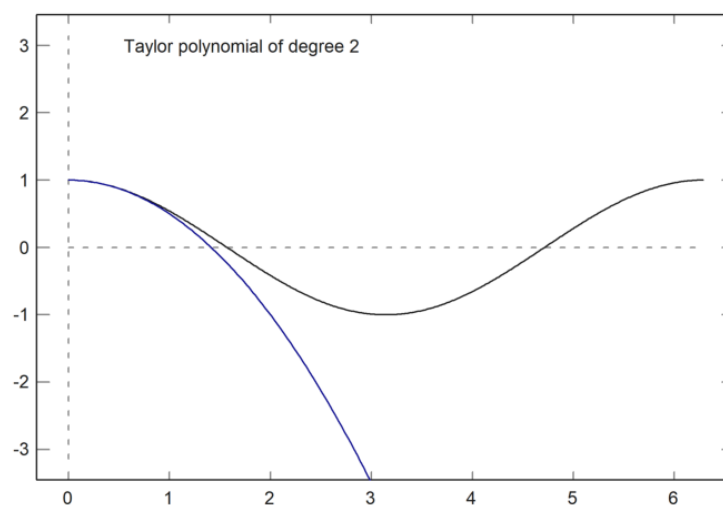
Dimungkinkan untuk membatasi nilai yang diseret ke bilangan bulat. Sebagai contoh, kita menulis fungsi plot, yang memplot polinomial Taylor berderajat n ke fungsi kosinus.

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

```
    plot2d("cos(x)",0,2pi,>square,grid=6);
    plot2d("&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.
an n ini, dan untuk memasukkan plot ke dalam buku catatan.

```
>function dragtest ...

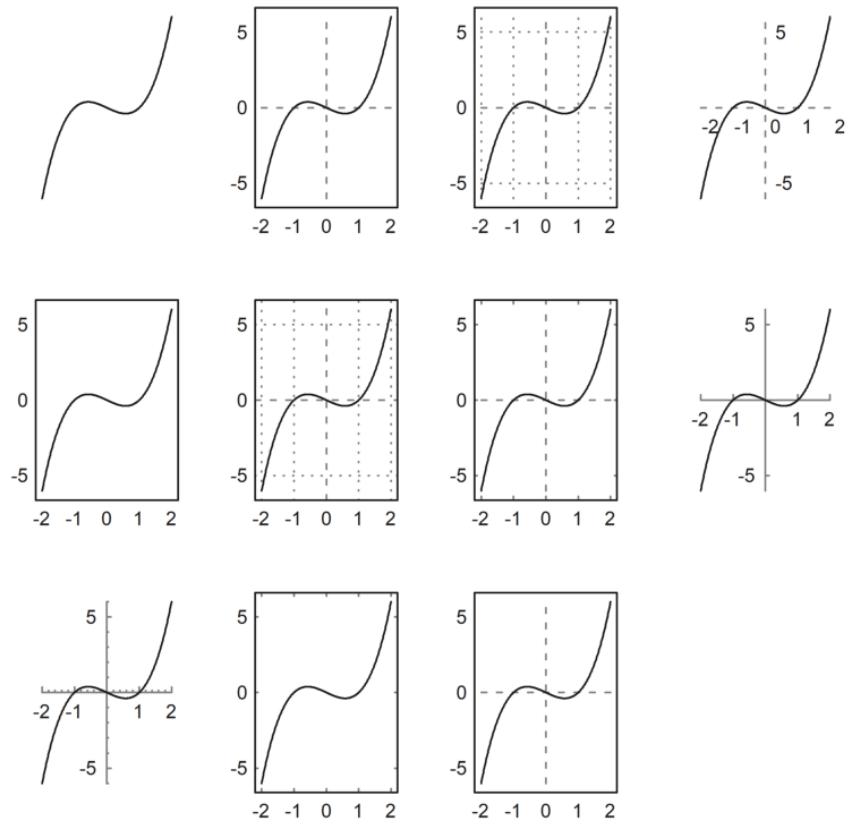
    plot2d(none,r=1,title="Drag with the mouse, or press any key!");
    start=0;
    repeat
        {flag,m,time}=mousedrag();
        if flag==0 then return; endif;
        if flag==2 then
            hold on; mark(m[1],m[2]); hold off;
        endif;
    end
endfunction
```

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

Gaya Plot 2D

Secara default, EMT menghitung tanda centang sumbu otomatis dan menambahkan label ke setiap tanda centang. Hal 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().

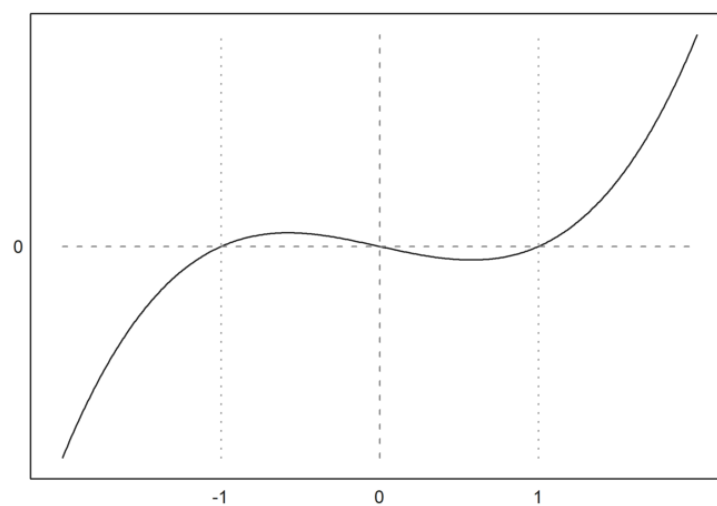
```
>aspect();
>figure(3,4); ...
> figure(1); plot2d("x^3-x",grid=0); ... // no grid, frame or axis
> figure(2); plot2d("x^3-x",grid=1); ... // x-y-axis
> figure(3); plot2d("x^3-x",grid=2); ... // default ticks
> figure(4); plot2d("x^3-x",grid=3); ... // x-y- axis with labels inside
> figure(5); plot2d("x^3-x",grid=4);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

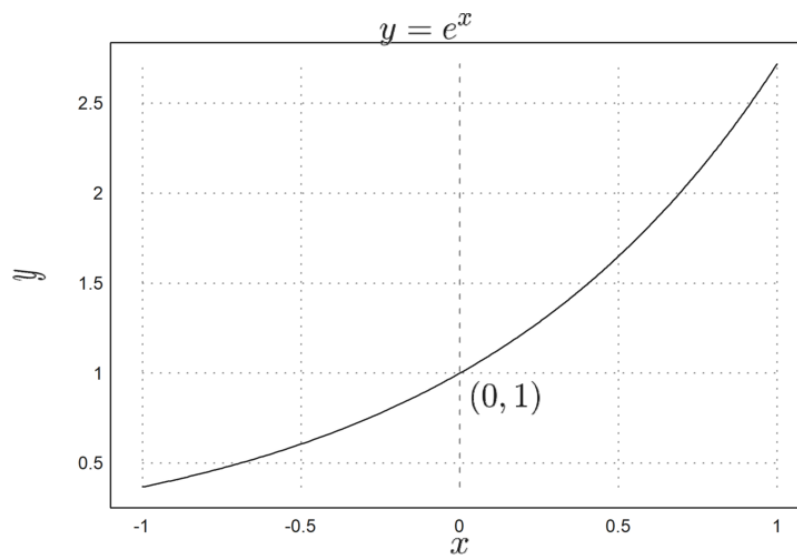
Jika Anda menginginkan tanda centang Anda sendiri, Anda dapat menggunakan `style=0` dan menambahkan semuanya nanti.

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



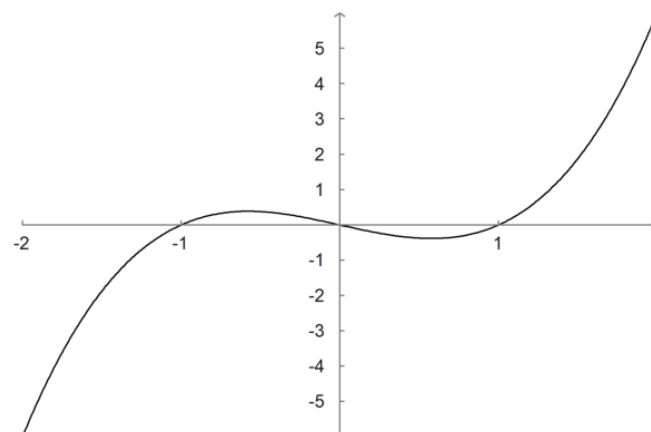
Untuk judul plot dan label sumbu, lihat contoh berikut.

```
>plot2d("exp(x)",-1,1);  
>textcolor(black); // set the text color to black  
>title(latex("y=e^x")); // title above the plot  
>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 `a axis()` dan `y axis()`.

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

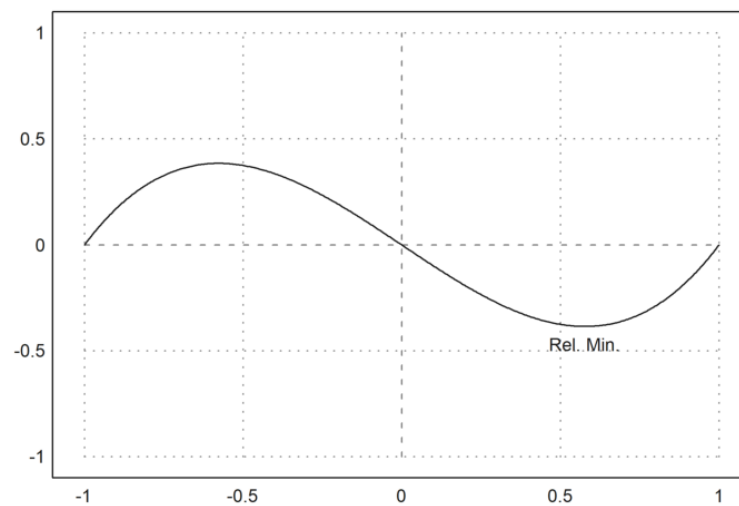


Teks pada plot dapat diatur dengan label(). Dalam contoh berikut, "lc" berarti tengah bawah. Fungsi ini mengatur posisi label relatif terhadap koordinat plot

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

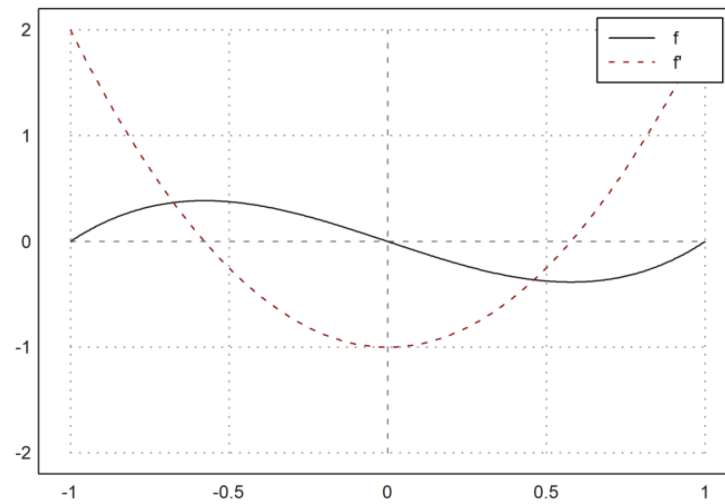
$$x^3 - x$$

```
>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
```

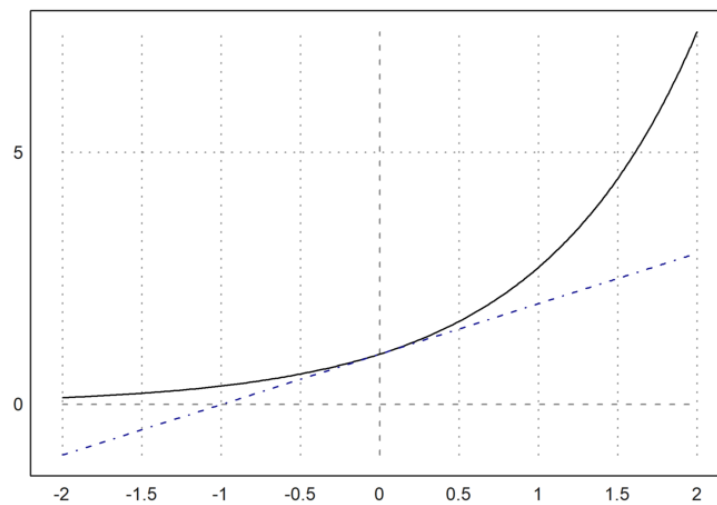


Ada juga kotak teks.

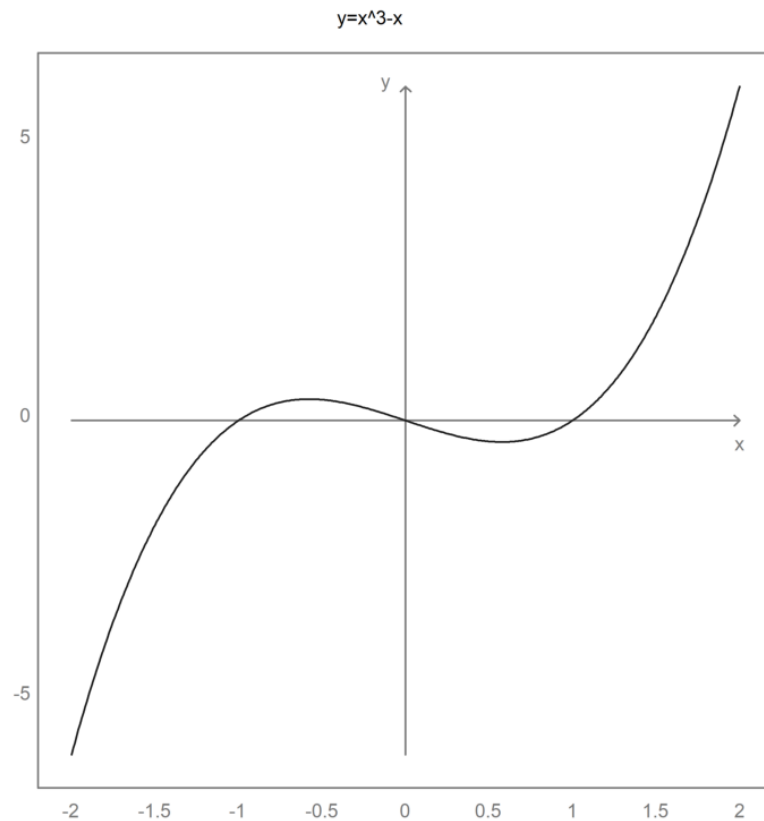
```
>plot2d(&f(x),-1,1,-2,2); // function  
>plot2d(&diff(f(x),x),>add,style="--",color=red); // derivative  
>labelbox(["f","f' "],["-", "--"],[black,red]): // label box
```



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



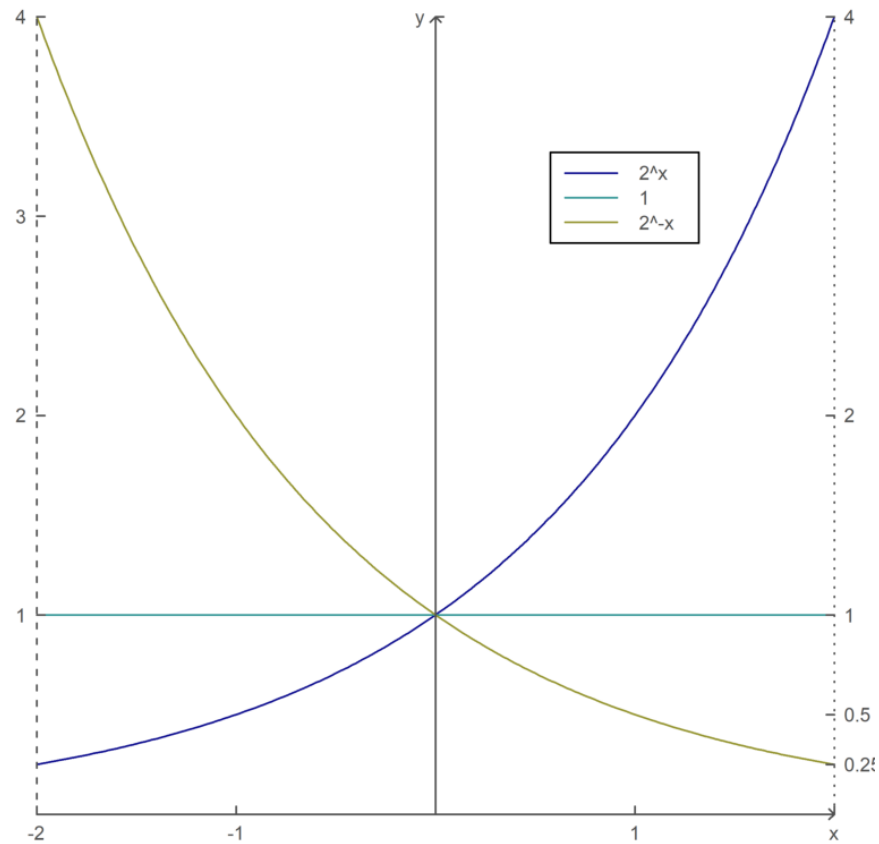
```
>gridstyle("->",color=gray,textcolor=gray,framecolor=gray); ...
> plot2d("x^3-x",grid=1); ...
> setttitle("y=x^3-x",color=black); ...
> label("x",2,0,pos="bc",color=gray); ...
> label("y",0,6,pos="cl",color=gray); ...
> reset():
```



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

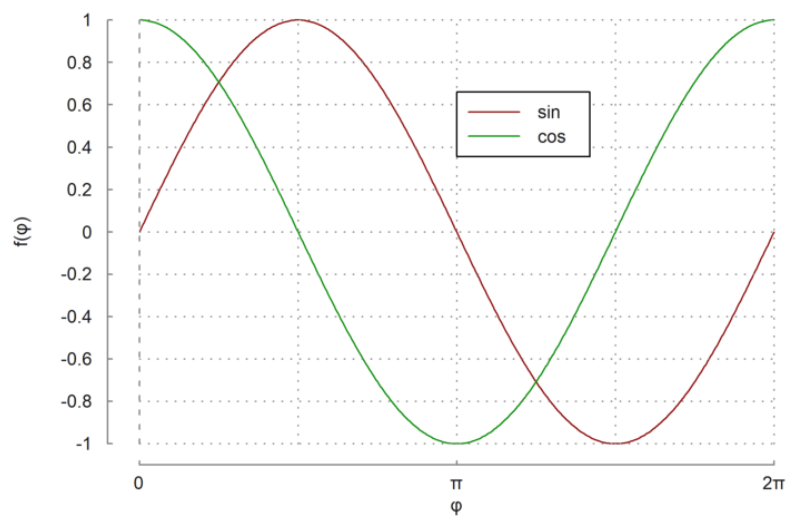
Perintah `fullwindow()` memperluas jendela plot karena kita tidak lagi membutuhkan tempat untuk label di luar jendela plot. Gunakan `shrinkwindow()` atau `reset()` untuk mengatur ulang ke nilai default. engatur ulang ke gaya default, gunakan `reset()`.

```
>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"&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;)"):
```



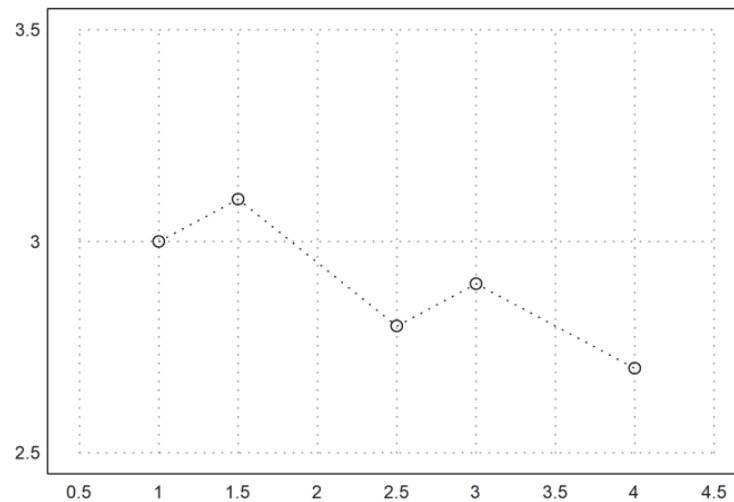
Memplot ekspresi hanyalah singkatan dari plot data. Untuk plot data, Anda memerlukan satu atau beberapa baris nilai-x, dan satu atau beberapa baris nilai-y. Dari rentang dan nilai-x, fungsi plot2d akan menghitung data yang akan diplot, secara default dengan evaluasi adaptif fungsi tersebut. Untuk plot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

- Gunakan vektor baris untuk x dan y untuk satu fungsi.

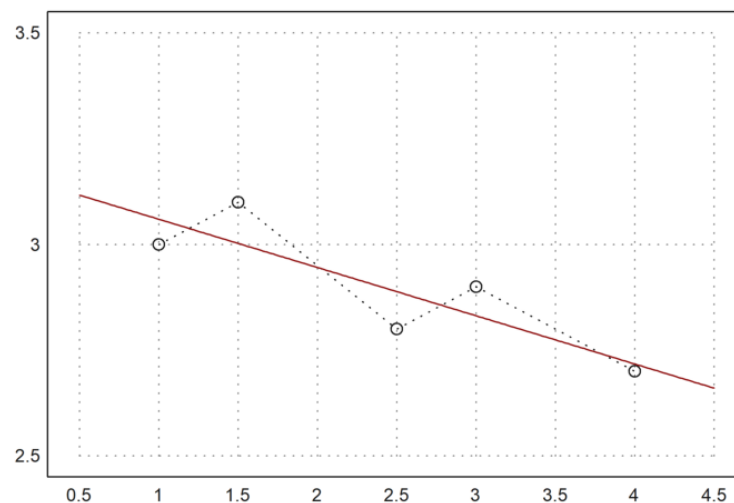
Berikut adalah contoh dengan satu baris untuk x dan y .

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

Parameter >addpoints menambahkan titik ke segmen garis untuk plot data.



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



Menggambar Daerah Yang Dibatasi Kurva

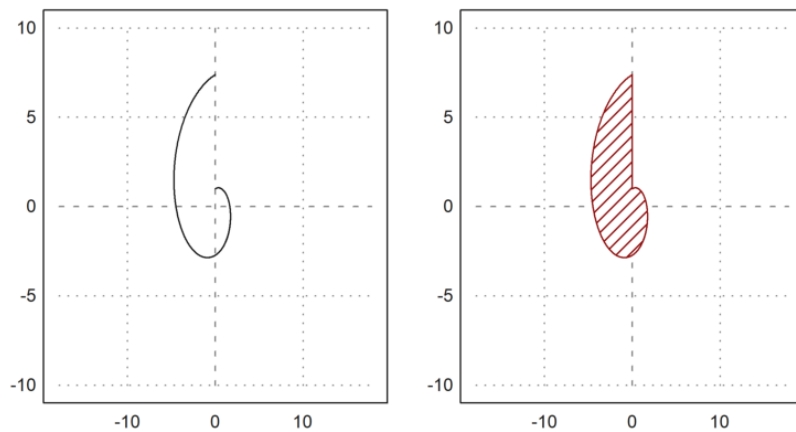
Plot data sebenarnya adalah poligon. Kita juga dapat memplot kurva atau kurva terisi.

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

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

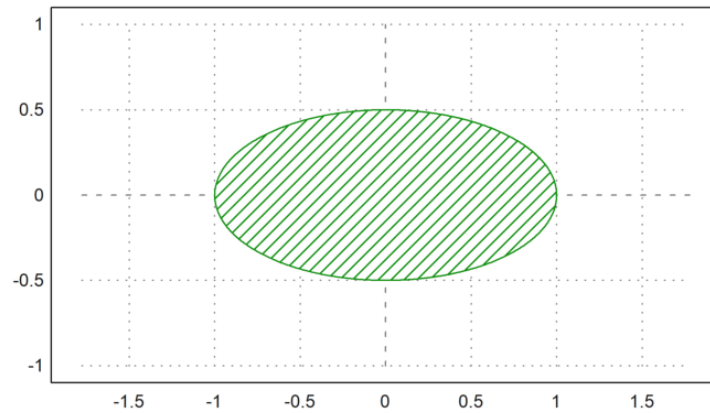
```
>t=linspace(0,2pi,1000); // parameter for curve
>x=sin(t)*exp(t/pi); y=cos(t)*exp(t/pi); // x(t) and y(t)
>figure(1,2); aspect(16/9)
>figure(1); plot2d(x,y,r=10); // plot curve
```

```
>figure(2); plot2d(x,y,r=10,>filled,style="/",fillcolor=red); // fill curve
>figure(0):
```

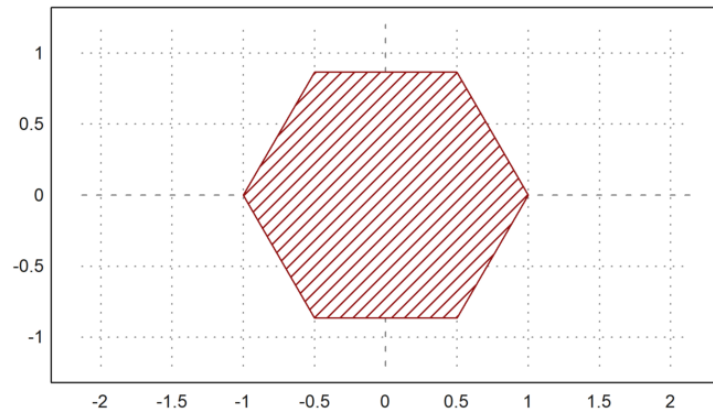


Dalam contoh berikut ini kami memplot elips yang terisi dan dua heksagon yang terisi menggunakan kurva tertutup dengan 6 titik dengan gaya isian yang berbeda.

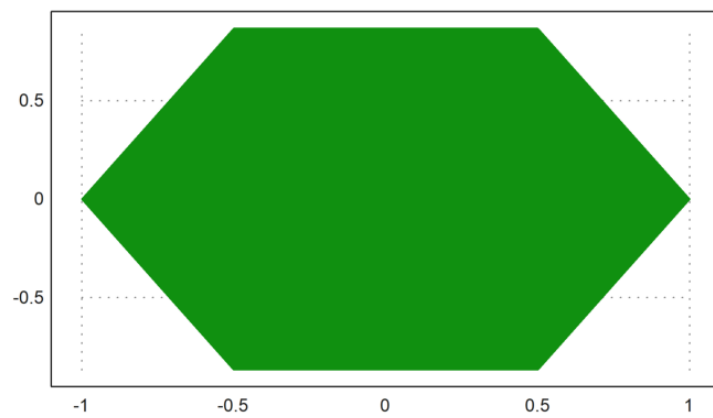
```
>x=linspace(0,2pi,1000); plot2d(sin(x),cos(x)*0.5,r=1,>filled,style="/"):
```



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

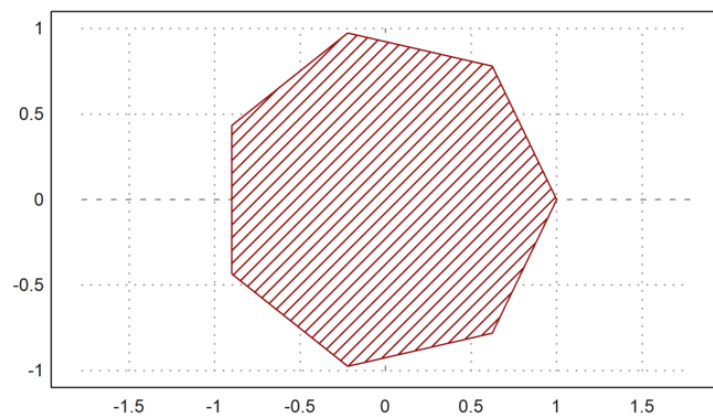


```
>t=linspace(0,2pi,6); plot2d(cos(t),sin(t),>filled,style="#"):
```



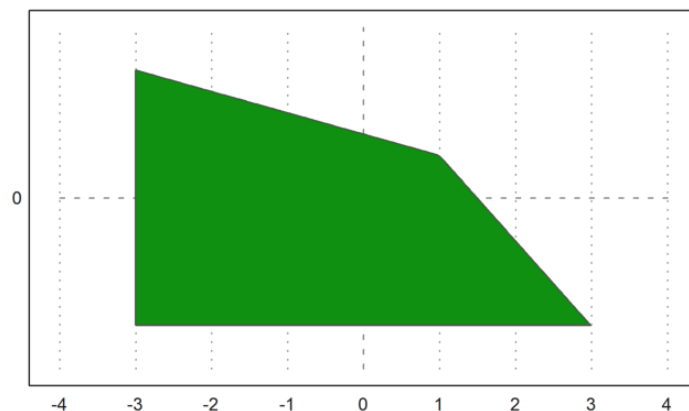
Contoh lain adalah septagon, yang kita buat dengan 7 titik lingkaran satuan.

```
>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):
```

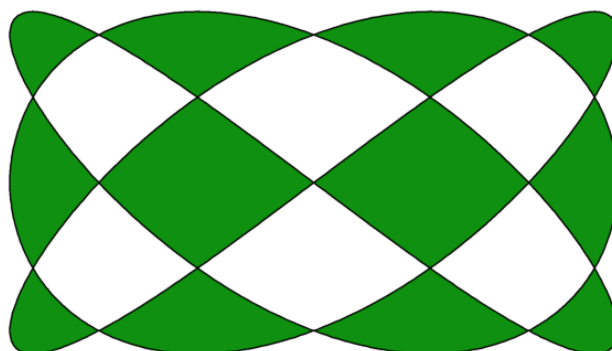


Poin utama dari bahasa matriks adalah memungkinkan pembuatan tabel fungsi dengan mudah.

```
>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, hasilnya akan bagus berkat aturan lilitan yang digunakan untuk pengisian.

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

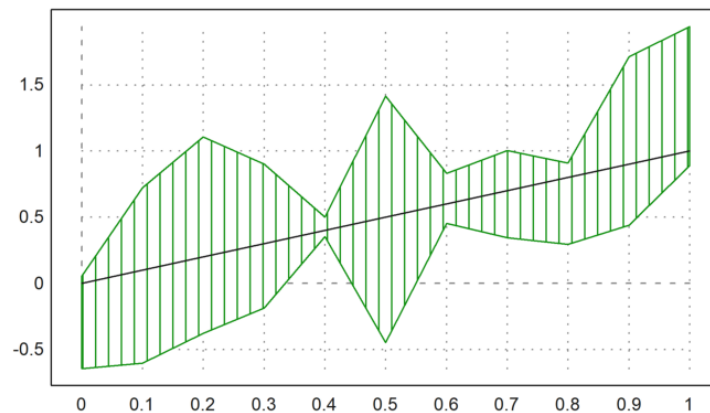


Vektor interval diplot terhadap nilai x sebagai daerah terisi antara nilai interval bawah dan atas.

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

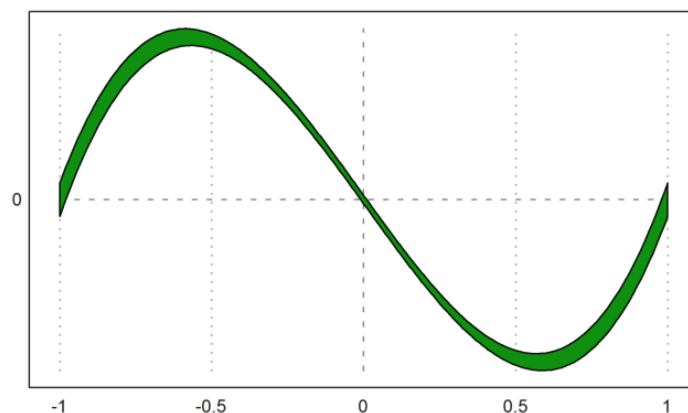
istical errors.

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



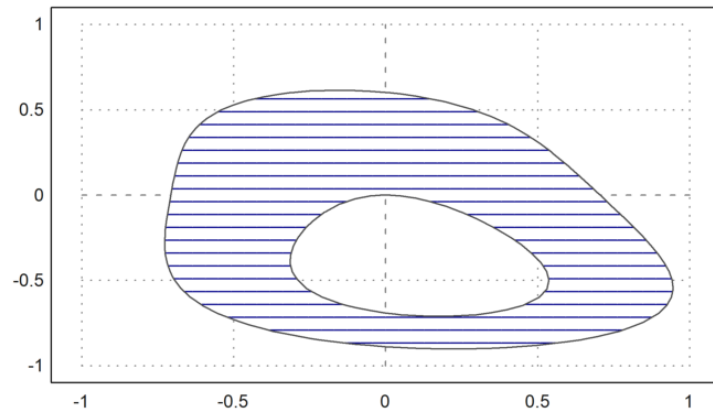
Jika x adalah vektor yang diurutkan, dan y adalah vektor interval, maka `plot2d` akan memplot rentang interval yang terisi pada bidang. Gaya isian sama dengan gaya poligon.

```
>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 level ini harus berupa matriks $2 \times n$. Baris pertama adalah batas bawah dan baris kedua adalah batas atas.

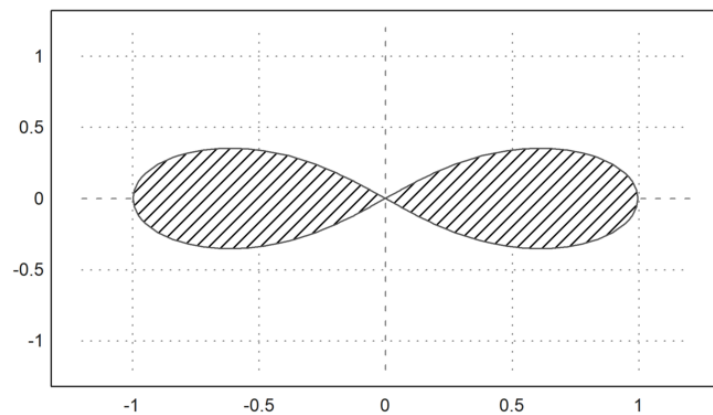
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=[0;1],style="-",color=blue): // 0 <= f(x,y) <= 1
```



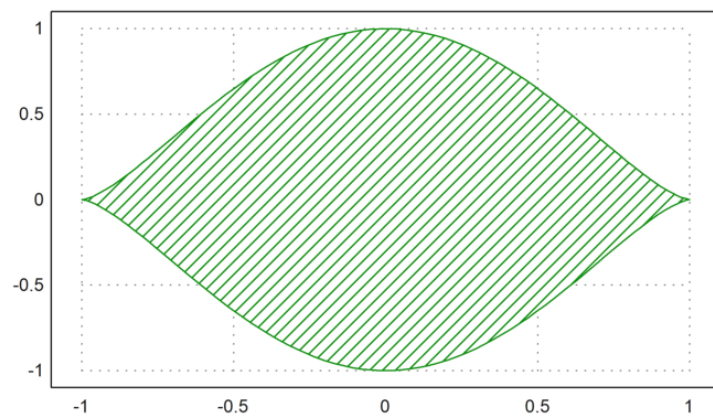
Kita juga dapat mengisi rentang nilai seperti

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

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



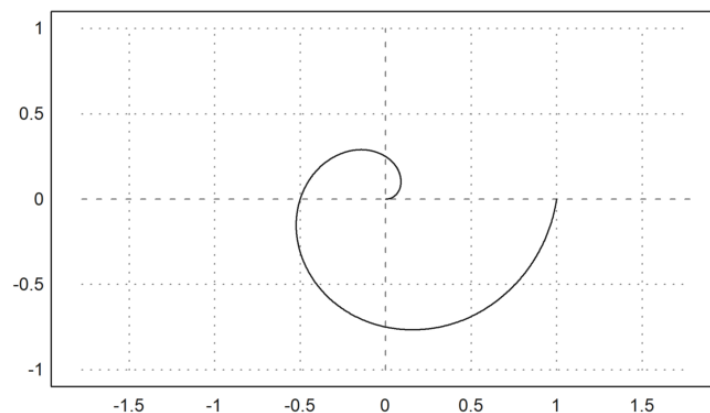
Nilai-nilai x tidak perlu diurutkan. (x,y) hanya menggambarkan sebuah kurva. Jika x diurutkan, kurva tersebut merupakan grafik suatu fungsi.

Pada contoh berikut, kami memplot spiral

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

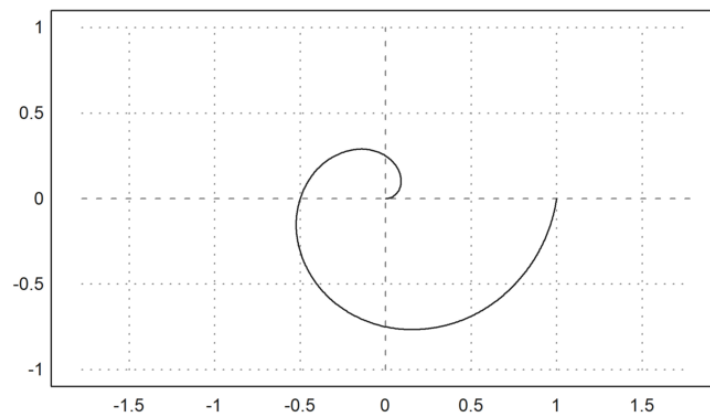
Kita perlu menggunakan banyak titik agar terlihat halus atau fungsi `adaptive()` untuk mengevaluasi ekspresi (lihat fungsi `adaptive()` untuk detail lebih lanjut).

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

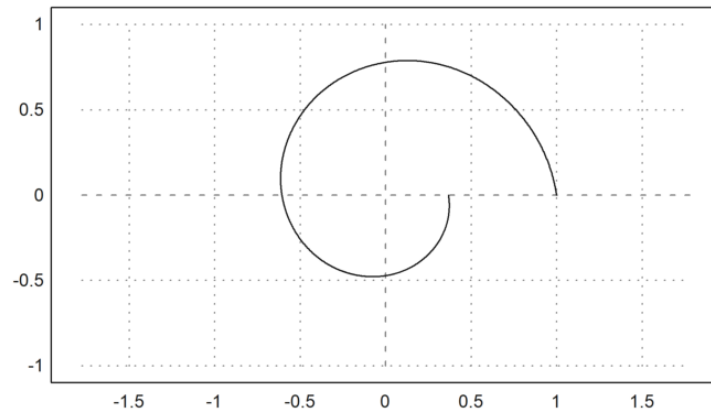


Sebagai alternatif, dimungkinkan untuk menggunakan dua ekspresi untuk kurva. Berikut ini memplot kurva yang sama seperti di atas.

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




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



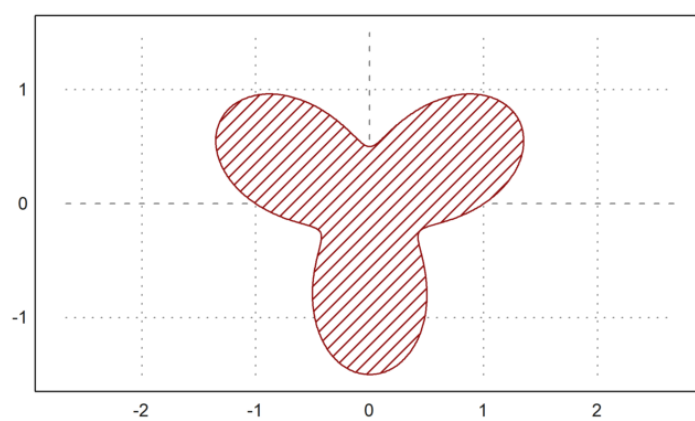
Pada contoh berikutnya, kita memplot kurva

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

dengan

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

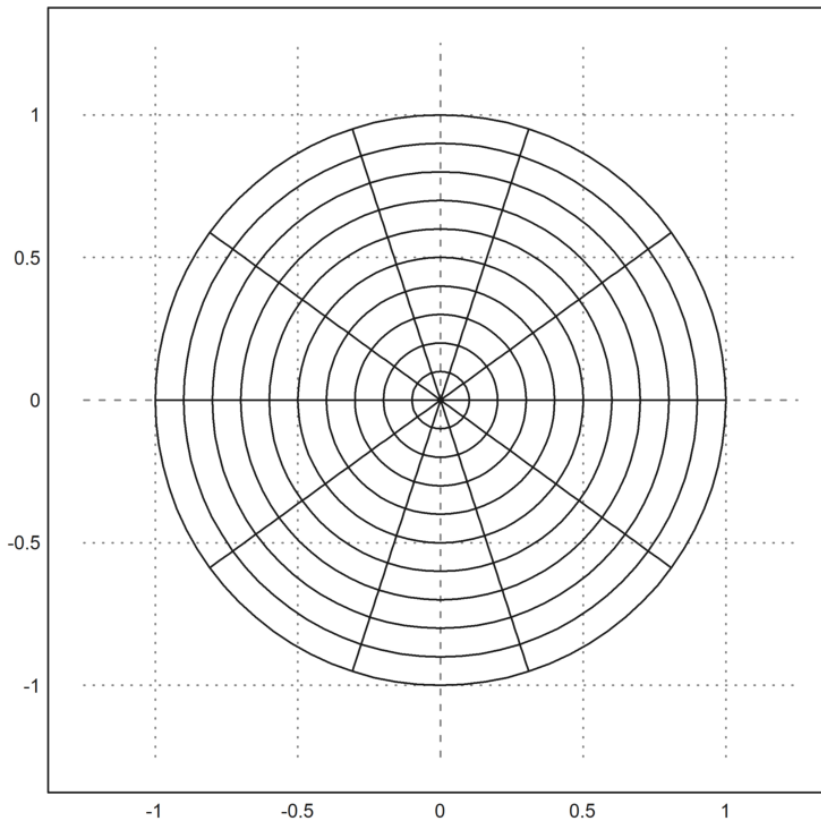
```
>t=linspace(0,2pi,1000); r=1+sin(3*t)/2; x=r*cos(t); y=r*sin(t); ...
>plot2d(x,y,>filled,fillcolor=red,style="/",r=1.5):
```



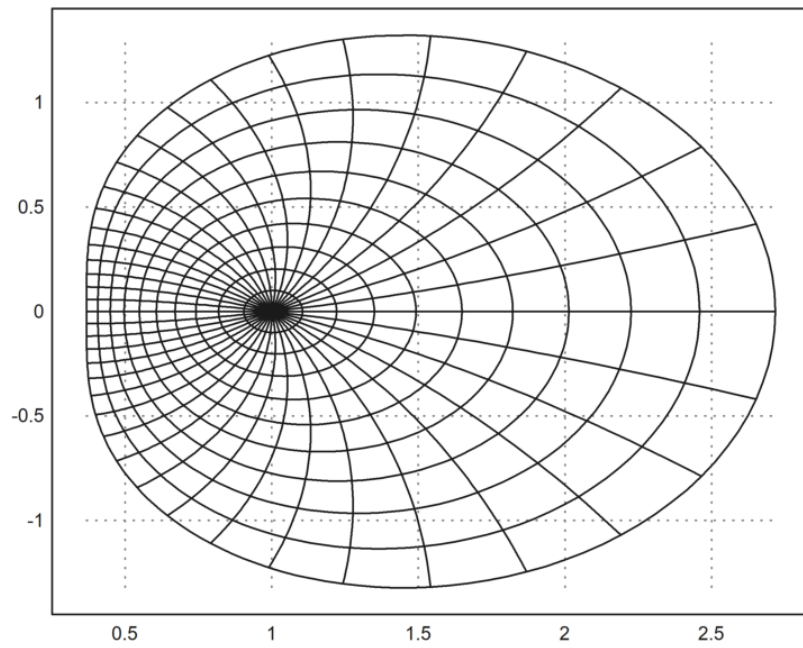
Menggambar Grafik Bilangan Kompleks

Deretan 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 kisi-kisi pada bidang kompleks. Dalam contoh berikut, kami memplot citra lingkaran satuan di bawah fungsi eksponensial. Parameter `cgrid` menyembunyikan beberapa kurva grid.

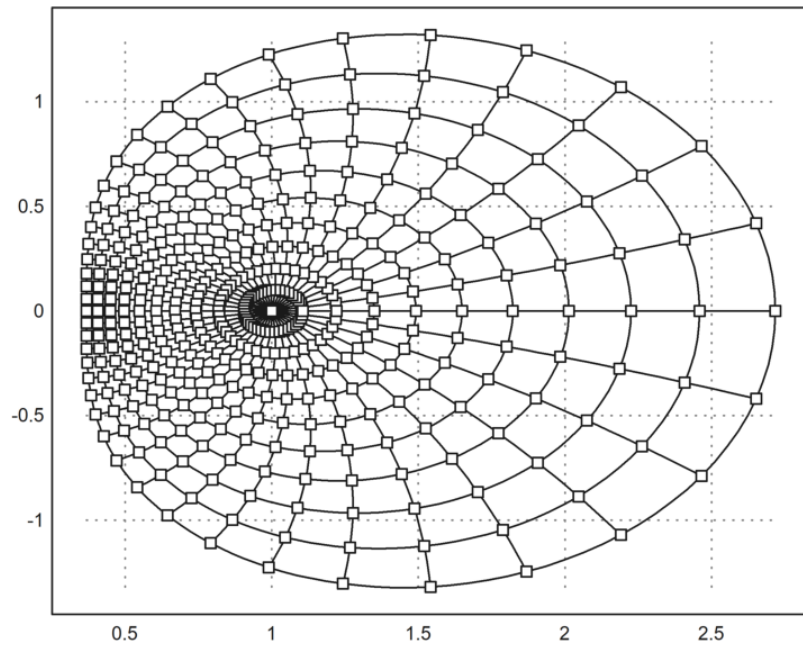
```
>aspect(); r=linspace(0,1,50); a=linspace(0,2pi,80)'; z=r*exp(I*a);...
>plot2d(z,a=-1.25,b=1.25,c=-1.25,d=1.25,cgrid=10):
```



```
>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):
```

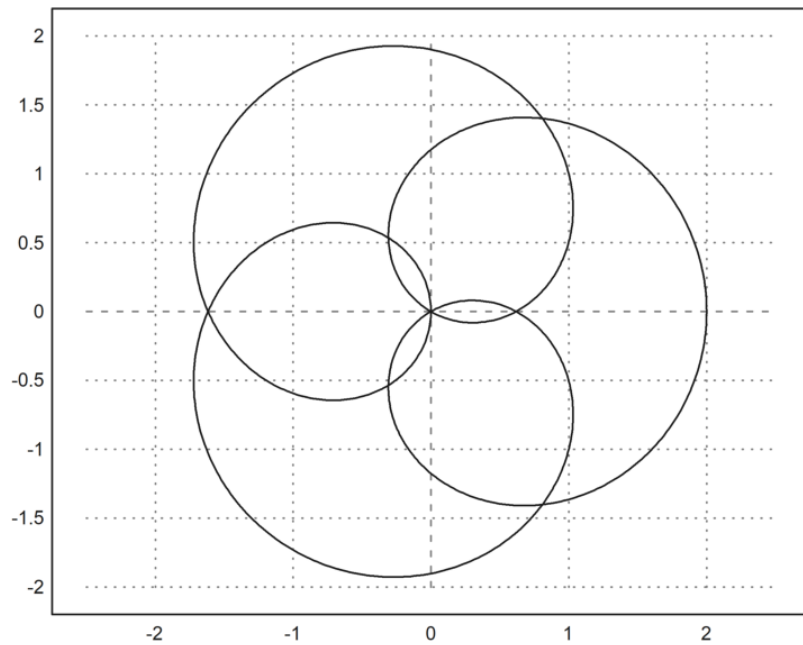


Vektor bilangan kompleks secara otomatis diplot sebagai kurva pada bidang kompleks dengan bagian riil dan bagian imajiner.

Dalam contoh ini, kita memplot lingkaran satuan dengan

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

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



Plot Statistik

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

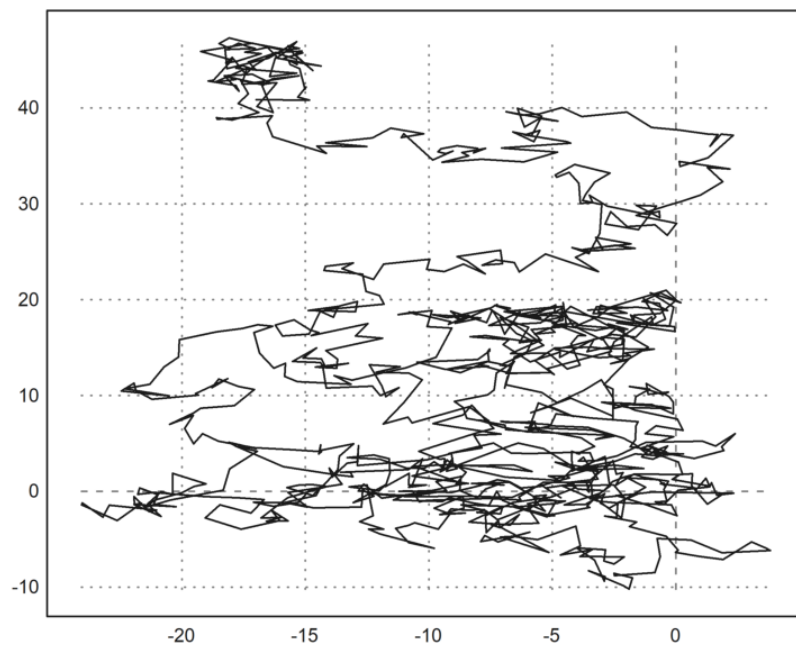
Jumlah kumulatif dari nilai-nilai yang terdistribusi normal 0-1 menghasilkan jalan acak.

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

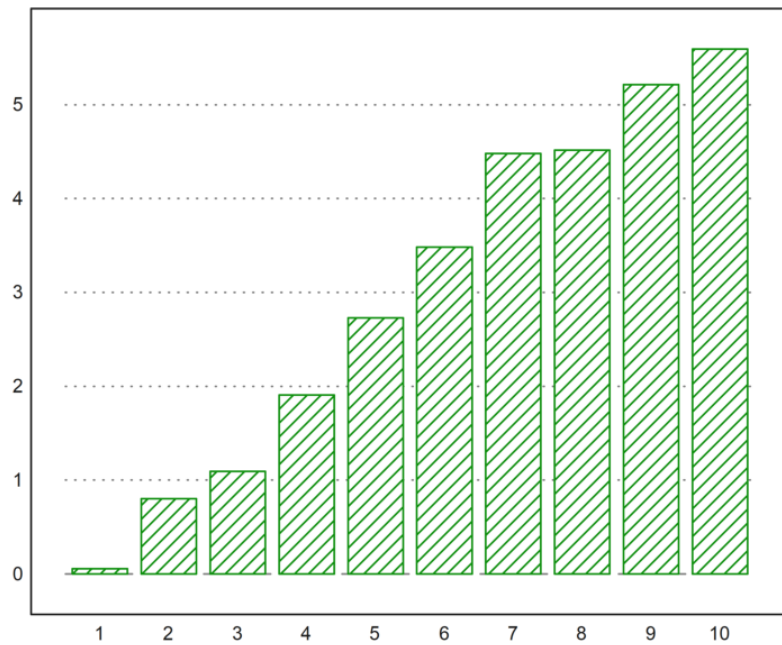


Menggunakan dua baris menunjukkan jalan dalam dua dimensi.

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

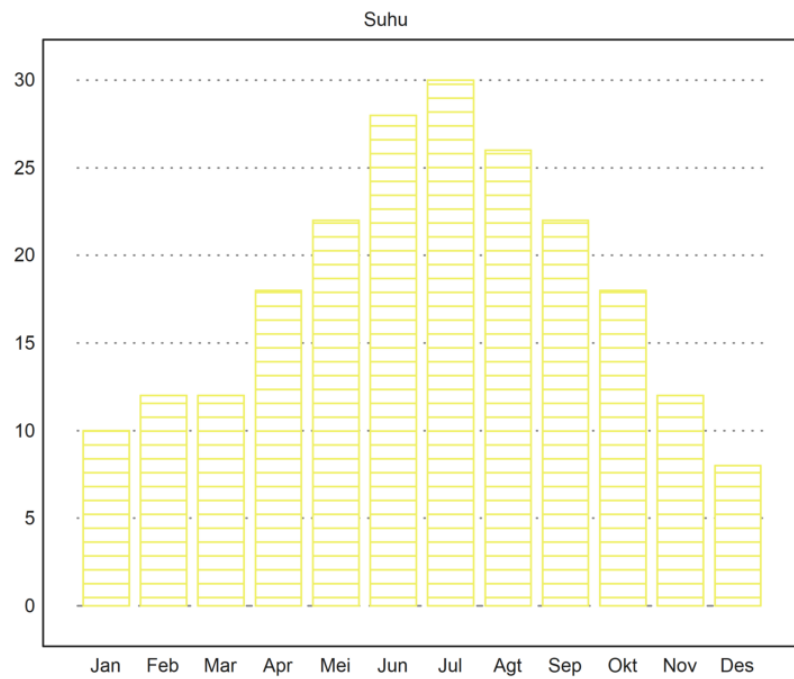


```
>columnsplot(cumsum(random(10)),style="/",color=green):
```

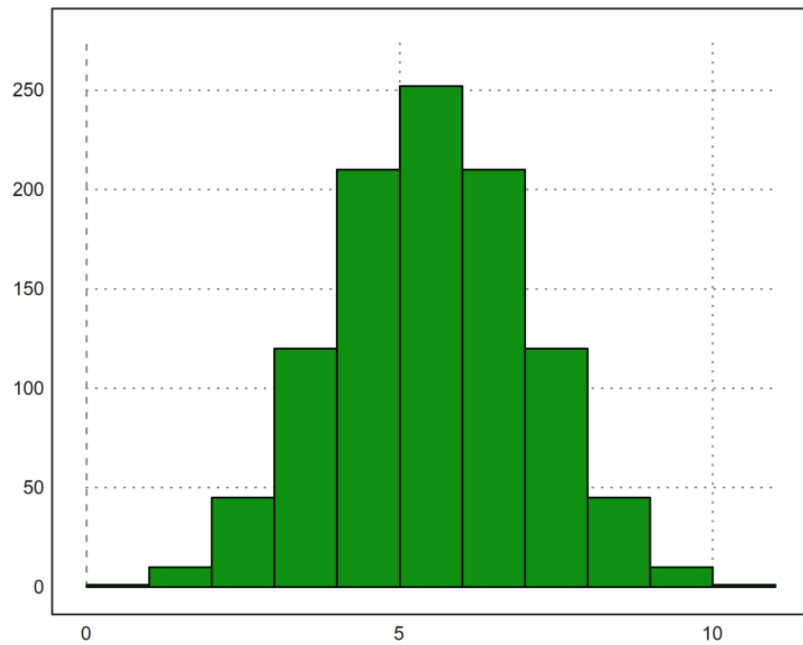


Ia juga dapat menampilkan string sebagai tabel.

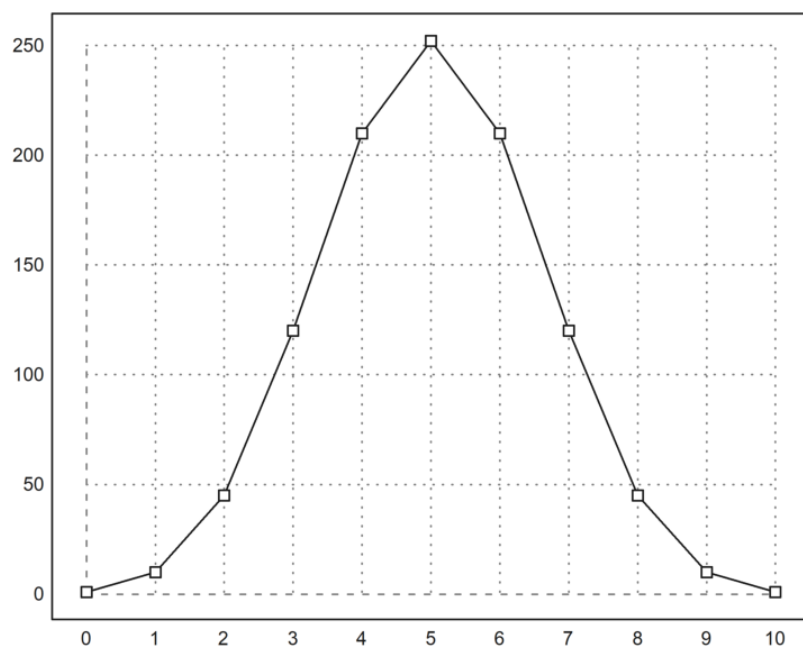
```
>bulan=["Jan", "Feb", "Mar", "Apr", "Mei", "Jun", ...
> "Jul", "Agt", "Sep", "Okt", "Nov", "Des"];
>values=[10,12,12,18,22,28,30,26,22,18,12,8];
>columnsplot(values,lab=bulan,color=yellow,style="-");
>title("Suhu"):
```



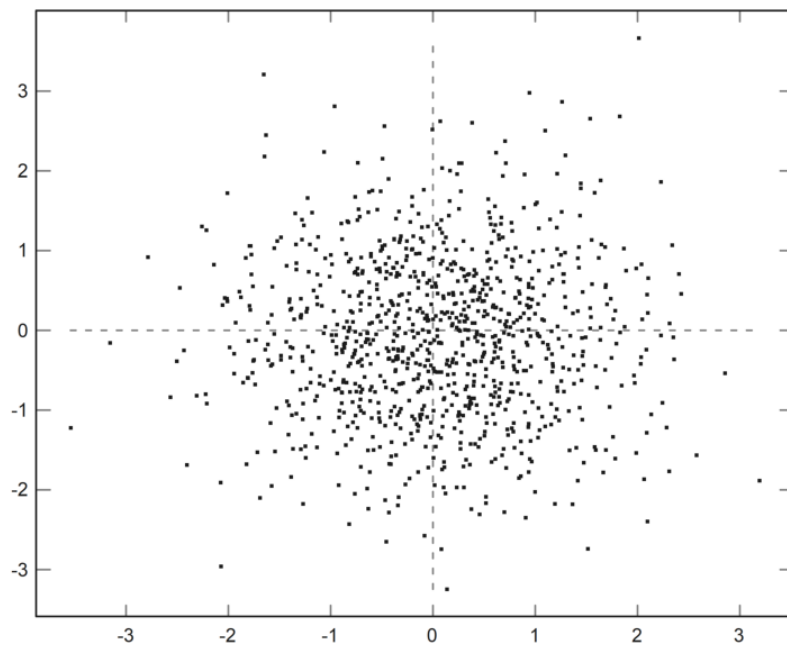
```
>k=0:10;
>plot2d(k,bin(10,k),>bar):
```



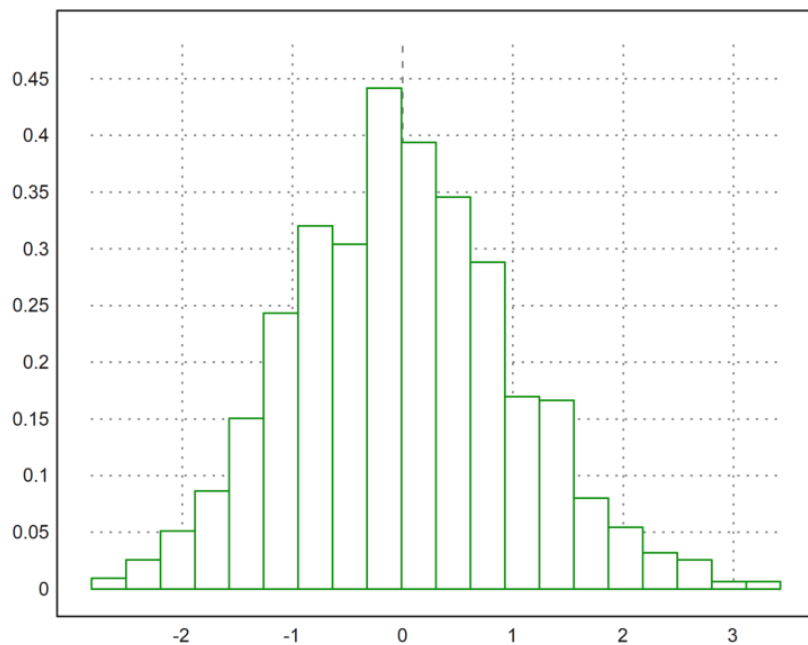
```
>plot2d(k,bin(10,k)); plot2d(k,bin(10,k),>points,>add):
```



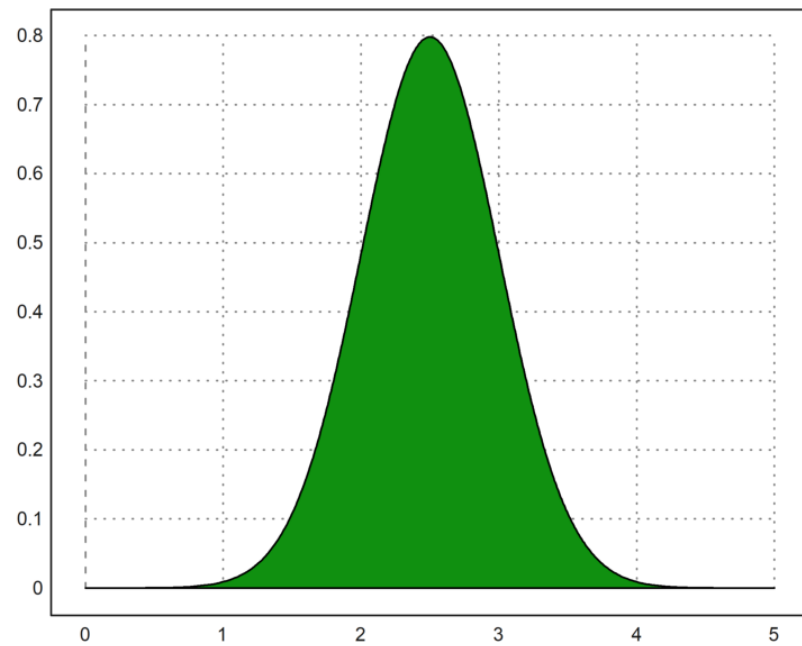
```
>plot2d(normal(1000),normal(1000),>points,grid=6,style=".."):
```



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

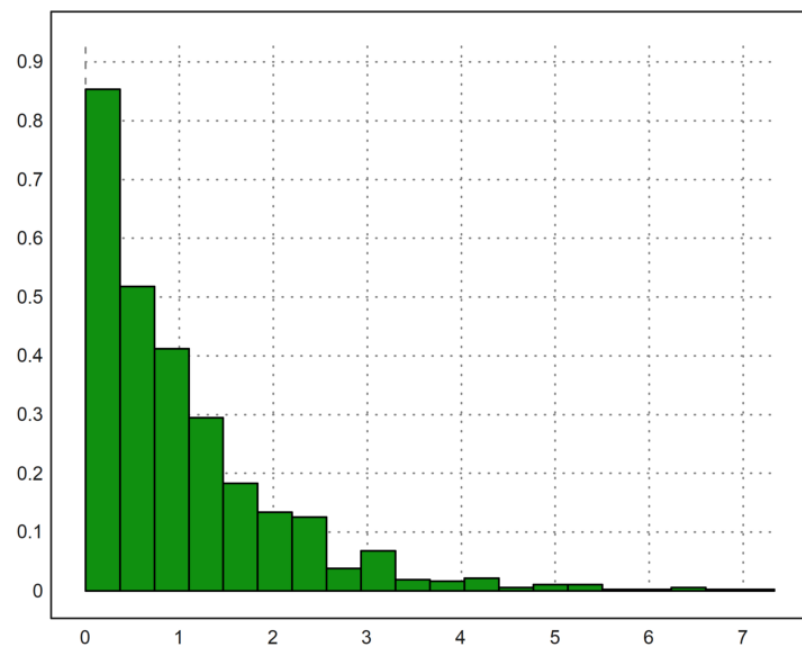



```
>plot2d("qnormal",0,5;2.5,0.5,>filled):
```



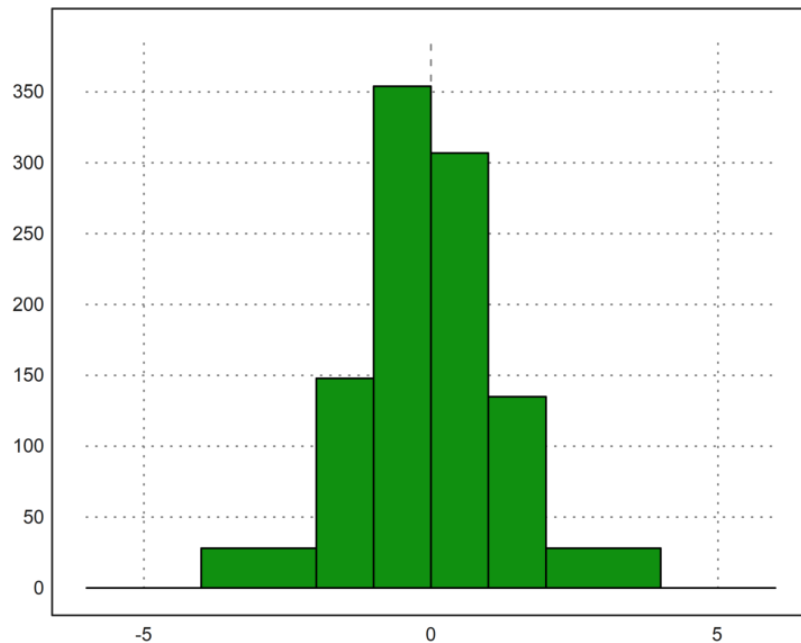
Untuk memplot distribusi statistik eksperimental, Anda dapat menggunakan `distribution=n` dengan `plot 2d`.

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



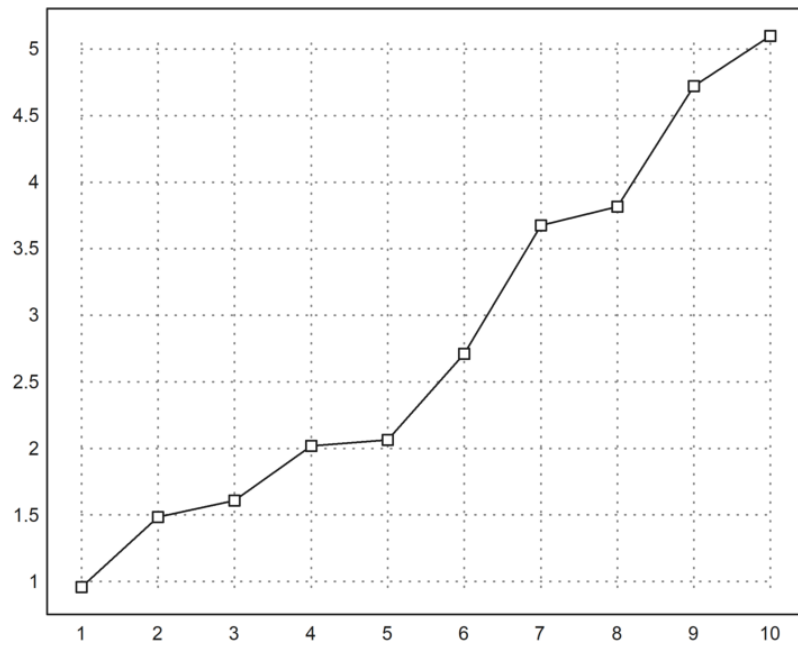
Atau Anda dapat menghitung distribusi dari data dan memplot hasilnya dengan `>bar` di `plot3d`, atau dengan `plot` kolom.

```
>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):
```

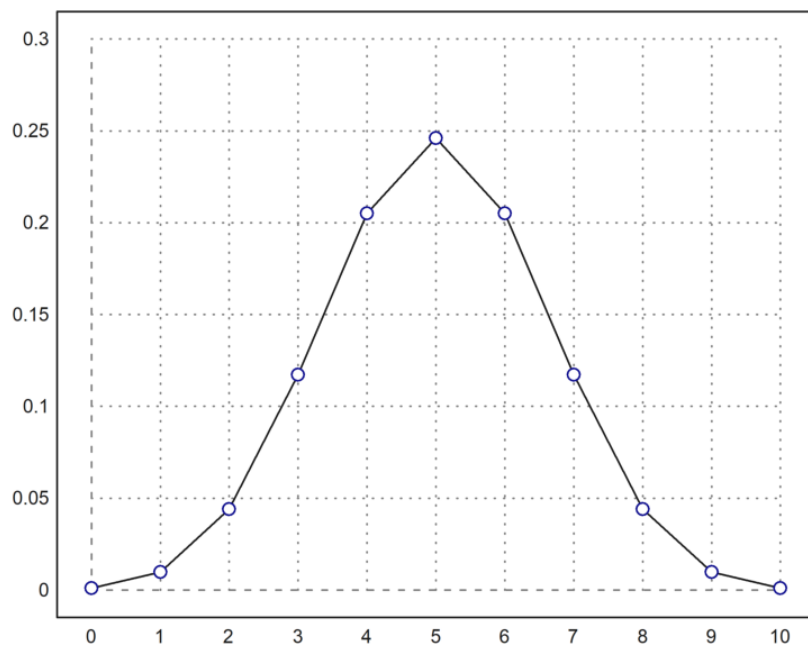


Fungsi `statplot()` mengatur gaya dengan string sederhana.

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



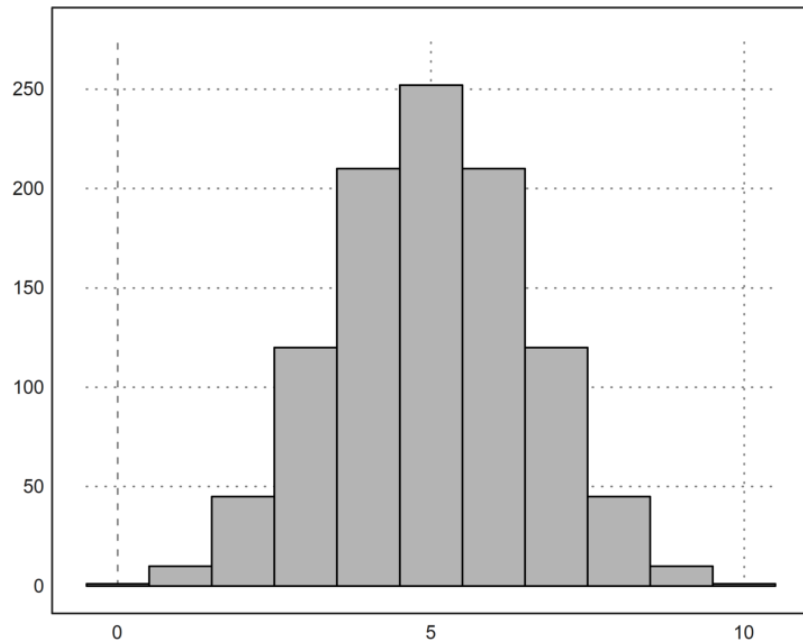
```
>n=10; i=0:n; ...
>plot2d(i,bin(n,i)/2^n,a=0,b=10,c=0,d=0.3); ...
>plot2d(i,bin(n,i)/2^n,points=true,style="ow",add=true,color=blue):
```



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.

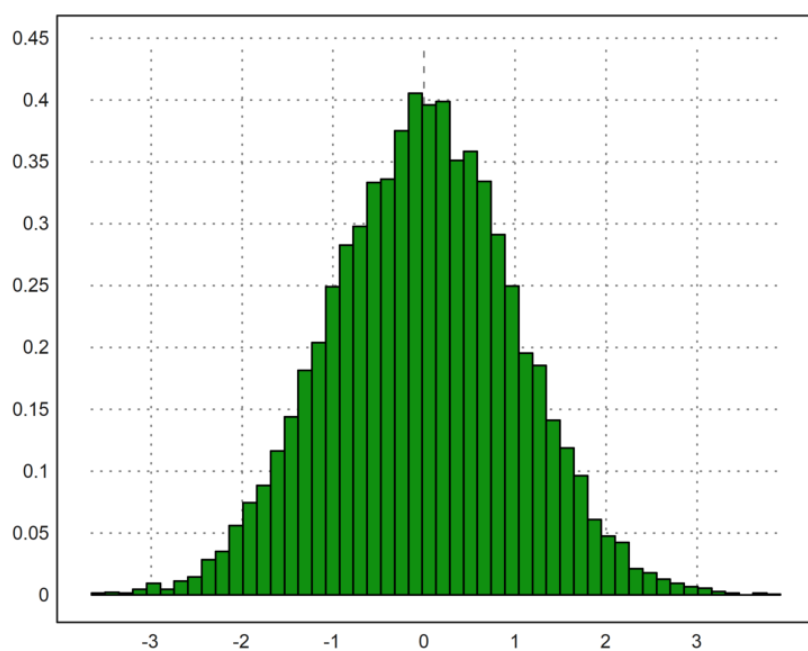
Gaya isian dapat digunakan seperti di atas.

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

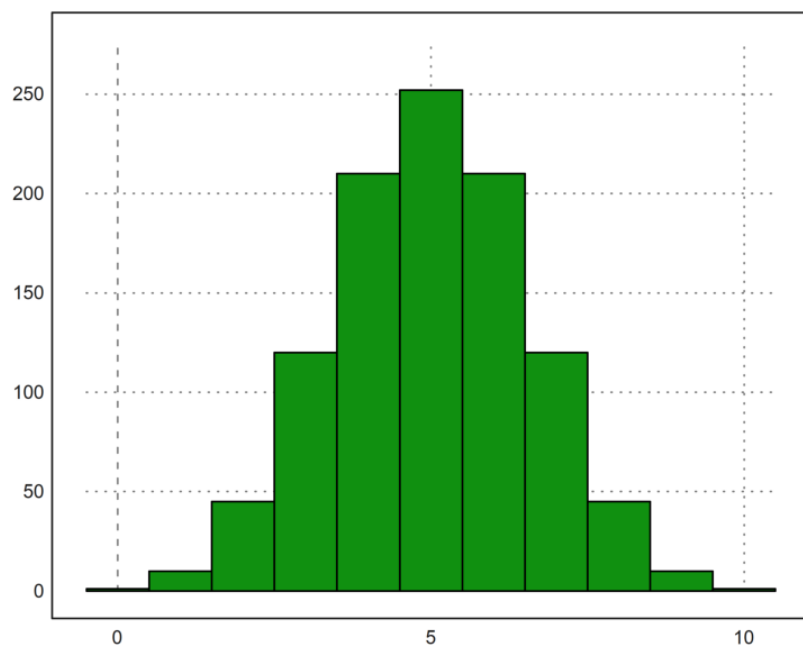


Data untuk diagram batang (`batang=1`) dan histogram (`histogram=1`) dapat diberikan secara eksplisit dalam `xv` dan `yv`, atau dapat dihitung dari distribusi empiris dalam `xv` dengan distribusi `>` (atau `distribusi=n`). Histogram nilai `xv` akan dihitung secara otomatis dengan histogram `>`. Jika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

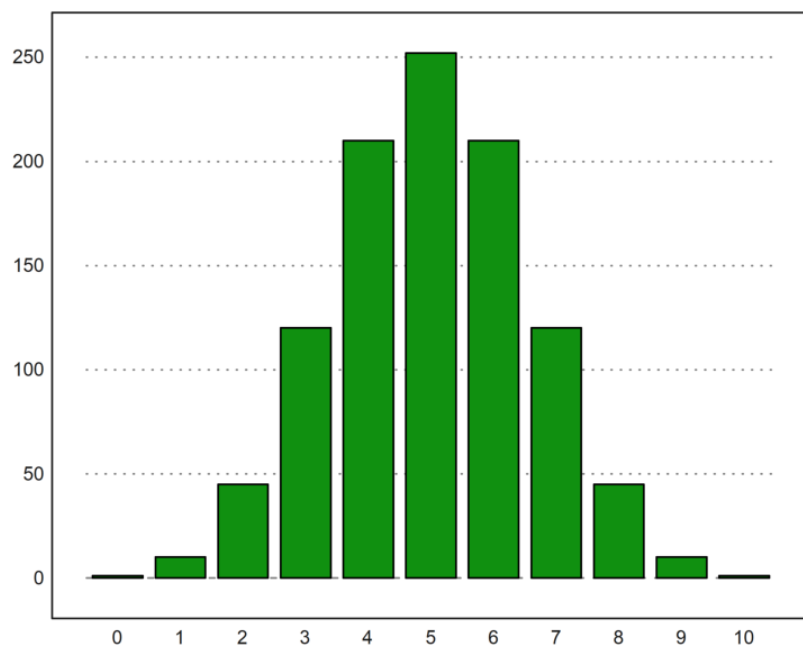
```
>plot2d(normal(10000),distribution=50):
```



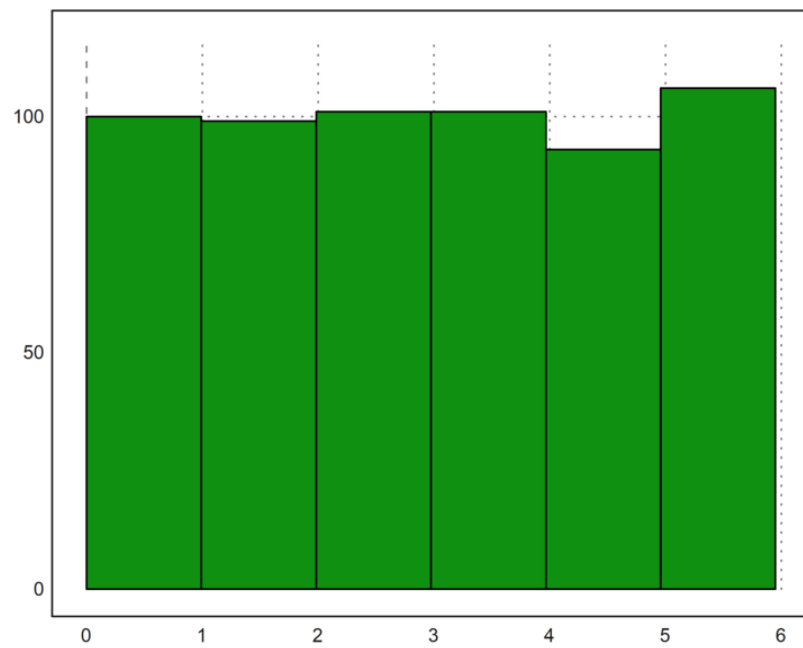
```
>k=0:10; m=bin(10,k); x=(0:11)-0.5; plot2d(x,m,>bar):
```



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

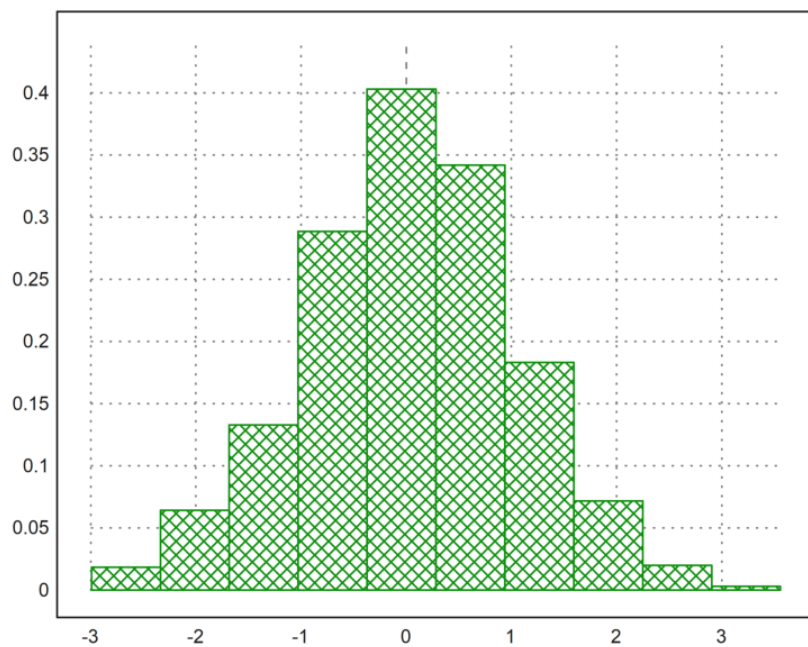


```
>plot2d(random(600)*6,histogram=6):
```



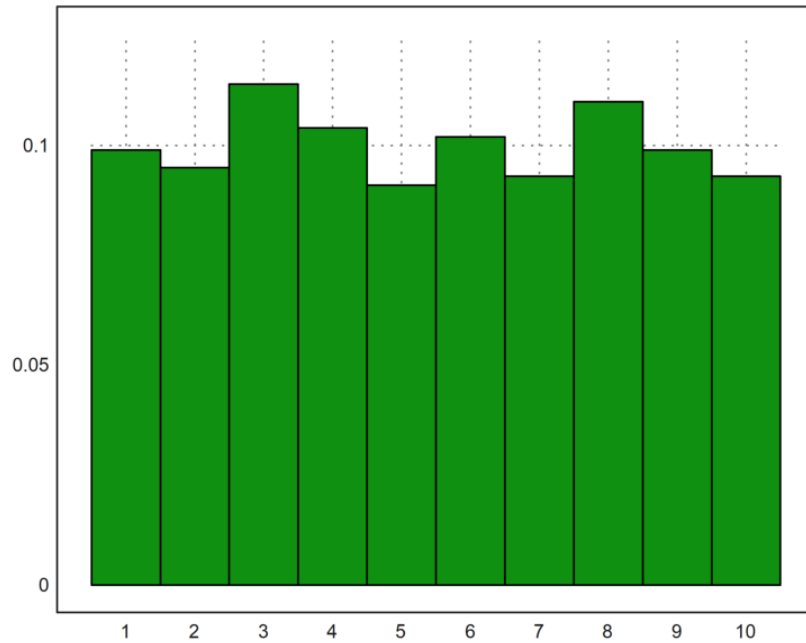
Untuk distribusi, ada parameter `distribution=n`, yang menghitung nilai secara otomatis dan mencetak distribusi relatif dengan `n` sub-interval.

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



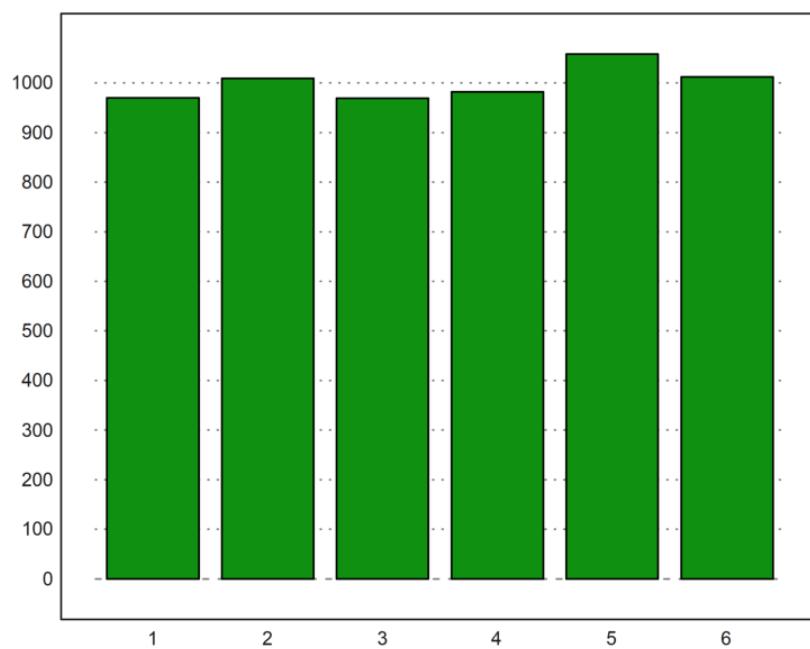
Dengan parameter `even=true`, ini akan menggunakan interval integer.

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

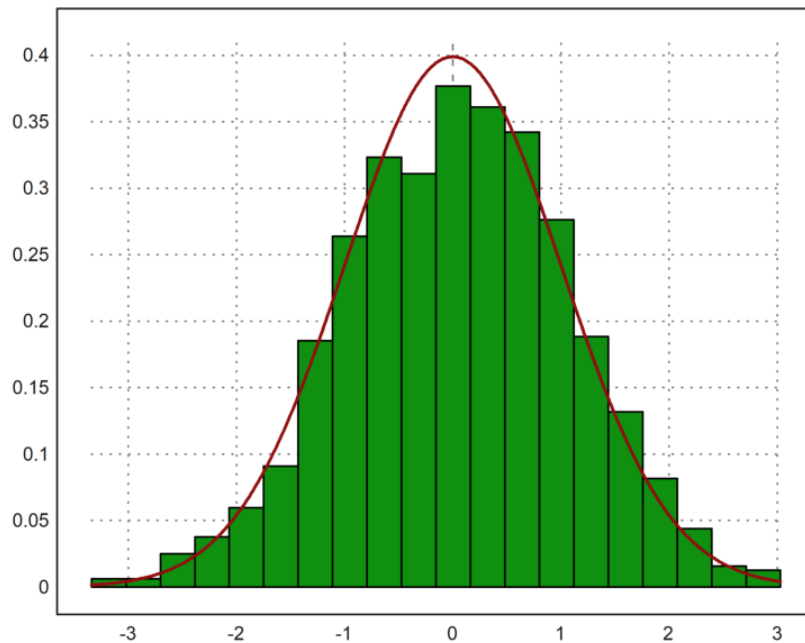


Perlu diketahui bahwa ada banyak plot statistik yang mungkin berguna. Lihat tutorial tentang statistik.

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

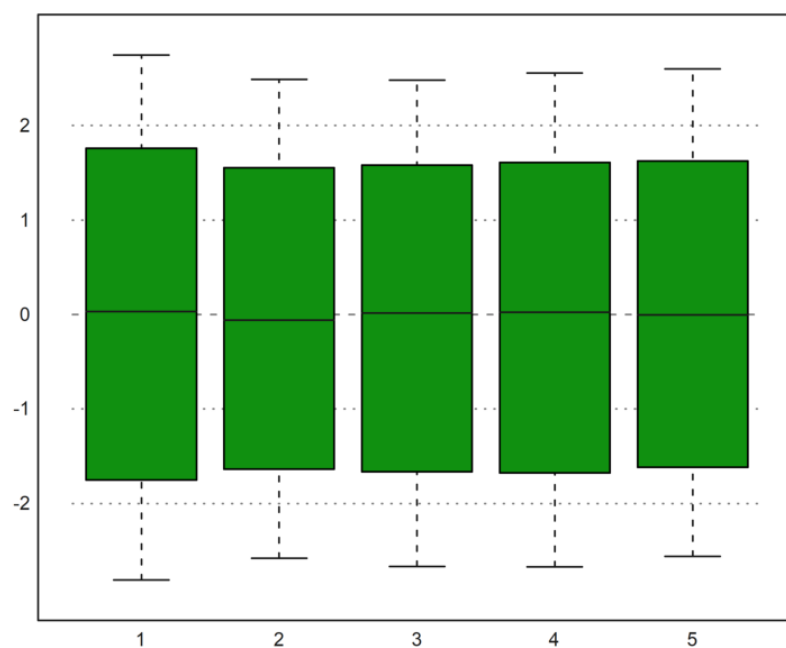


```
>plot2d(normal(1,1000),>distribution); ...
> plot2d("qnormal(x)",color=red,thickness=2,>add):
```



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)):
```



Plot implisit menunjukkan garis level yang menyelesaikan $f(x,y)=\text{level}$, di mana "level" dapat berupa nilai tunggal atau vektor nilai. Jika

`level="auto"`, akan ada garis level `nc`, 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.

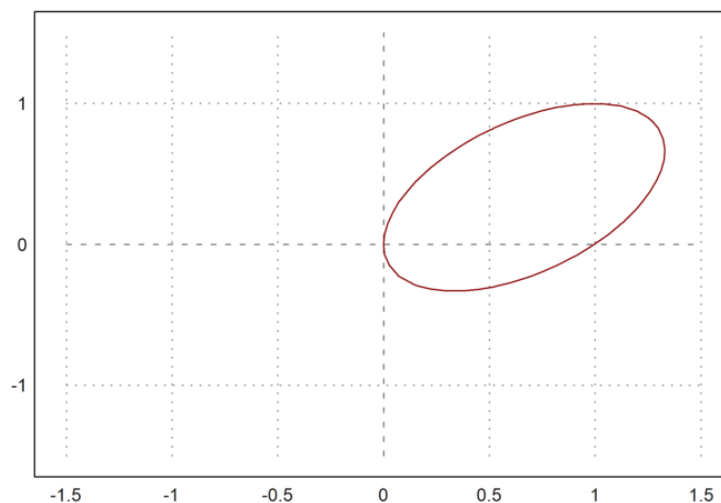
Euler dapat menandai garis level

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

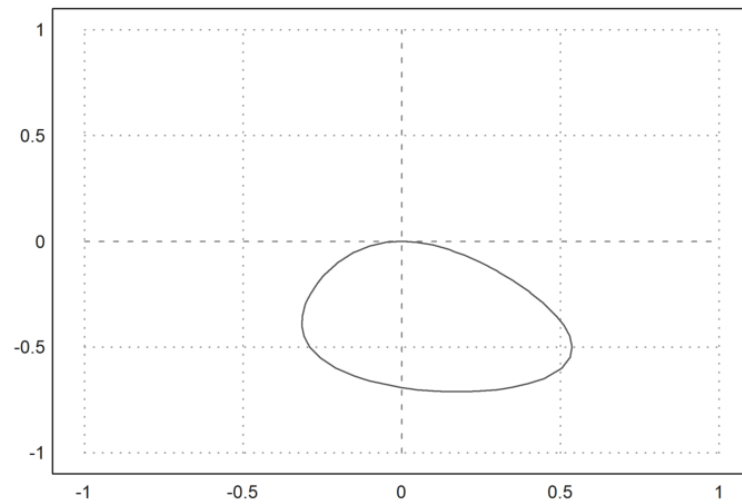
fungsi apapun.

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 `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.

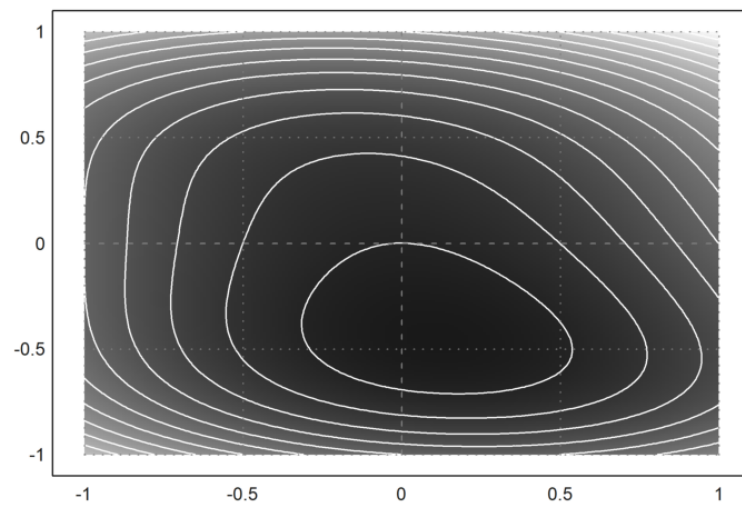
```
>aspect(1.5);  
>plot2d("x^2+y^2-x*y-x",r=1.5,level=0,contourcolor=red):
```



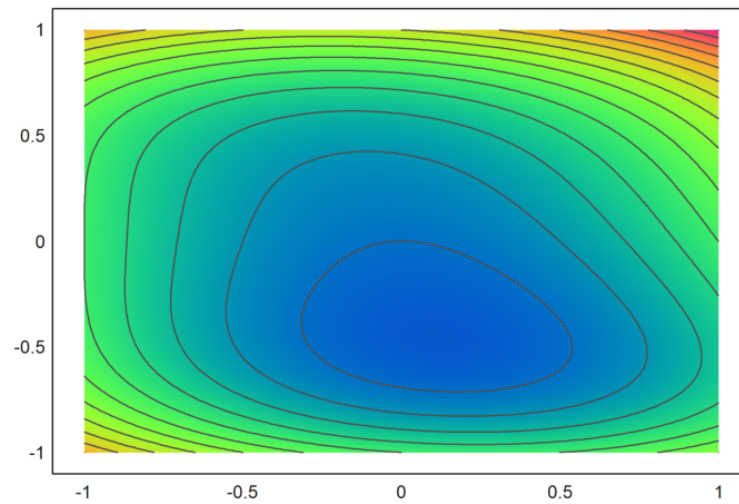
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)  
>plot2d(expr,level=0): // Solutions of f(x,y)=0
```



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

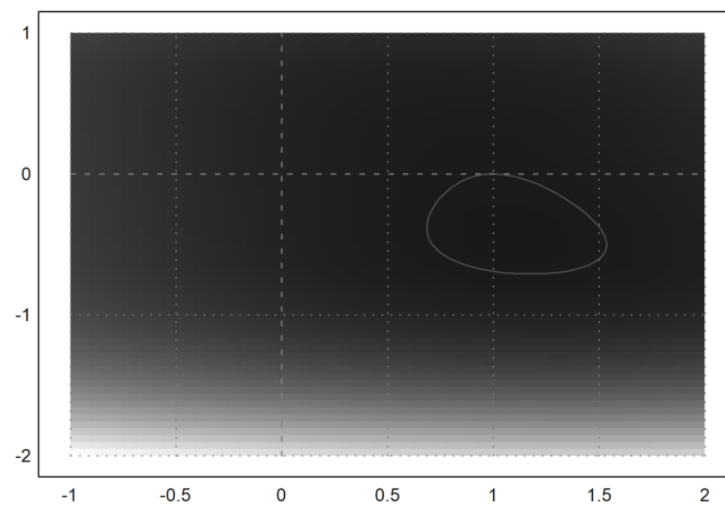


```
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=200,grid=4): // nicer
```

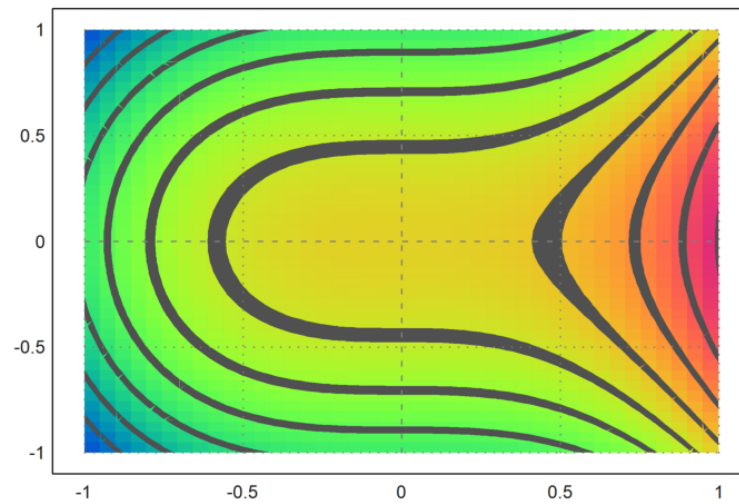


Ini juga berlaku untuk plot data. Namun, Anda harus menentukan rentang untuk label sumbu.

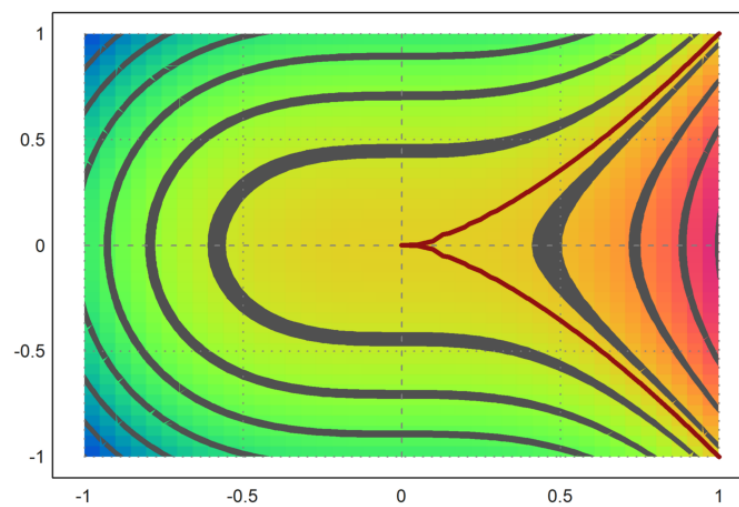
```
>x=-2:0.05:1; y=x'; z=expr(x,y);
>plot2d(z,level=0,a=-1,b=2,c=-2,d=1,>hue):
```



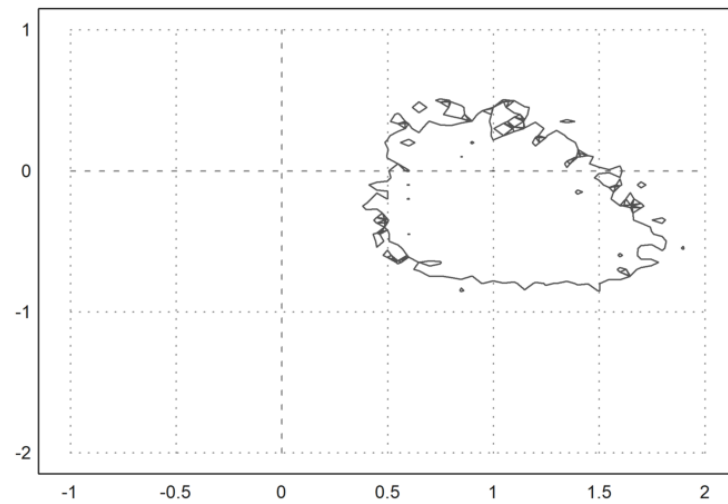
```
>plot2d("x^3-y^2",>contour,>hue,>spectral):
```



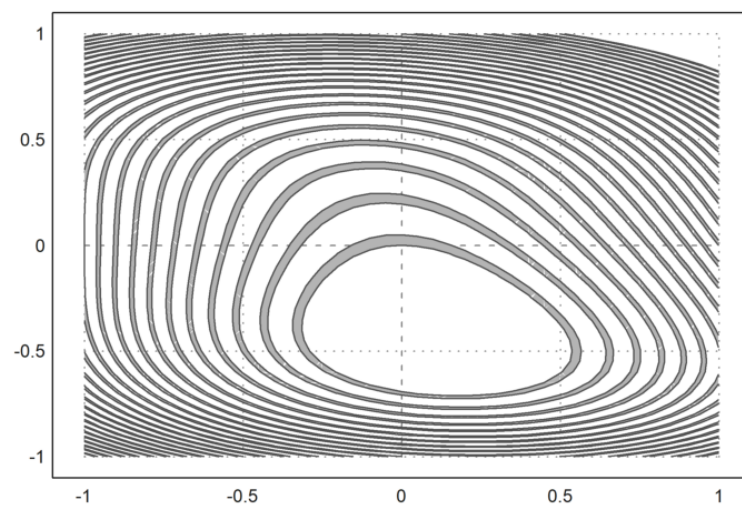
```
>plot2d("x^3-y^2",level=0,contourwidth=3,>add,contourcolor=red):
```



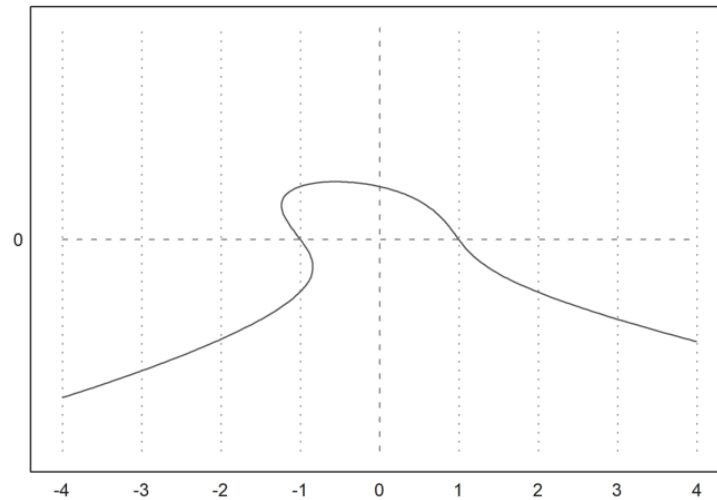
```
>z=z+normal(size(z))*0.2;
>plot2d(z,level=0.5,a=-1,b=2,c=-2,d=1):
```



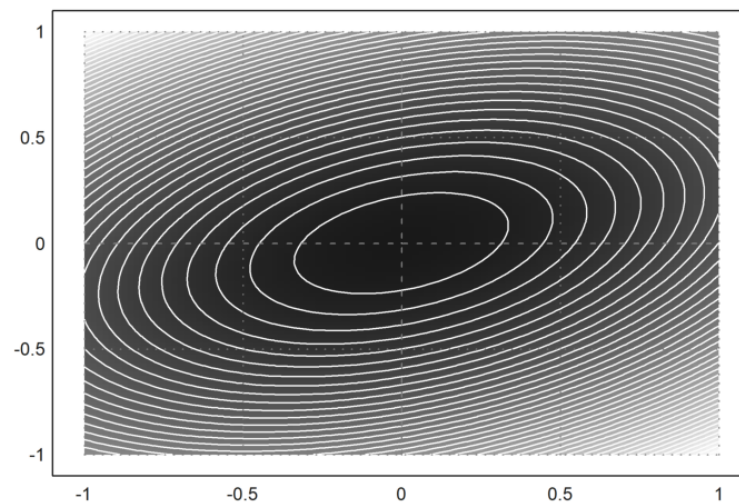
```
>plot2d(expr,level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
```



```
>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):
```



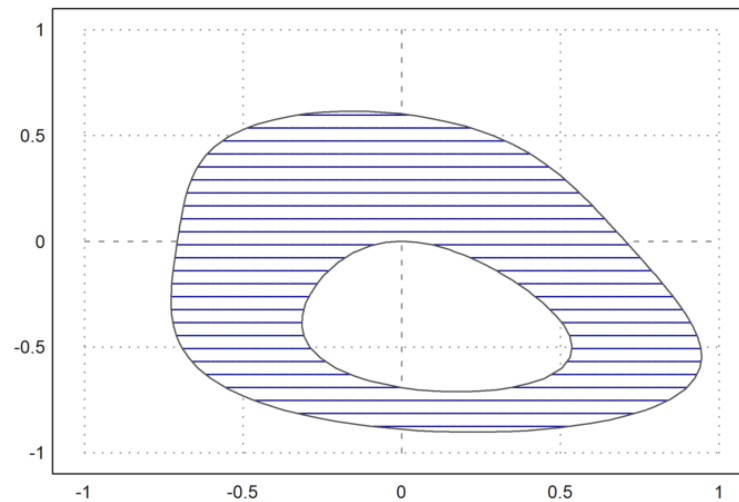
Bisa juga untuk mengisi set

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

dengan rentang level.

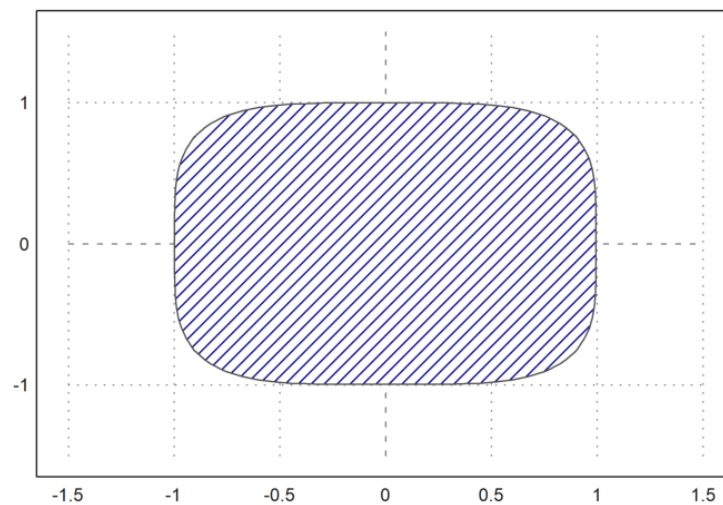
Dimungkinkan untuk mengisi wilayah nilai untuk fungsi tertentu. Untuk ini, levelnya harus berupa matriks 2xn. Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

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

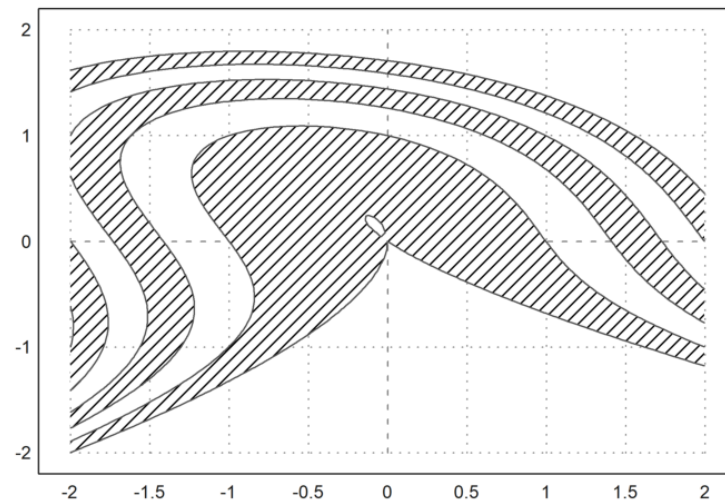


Plot implisit juga dapat menunjukkan rentang level. Level harus berupa matriks 2xn interval level, dengan baris pertama berisi awal dan baris kedua berisi akhir setiap interval. Sebagai alternatif, vektor baris sederhana dapat digunakan untuk level, dan parameter `dl` memperluas nilai level ke interval.

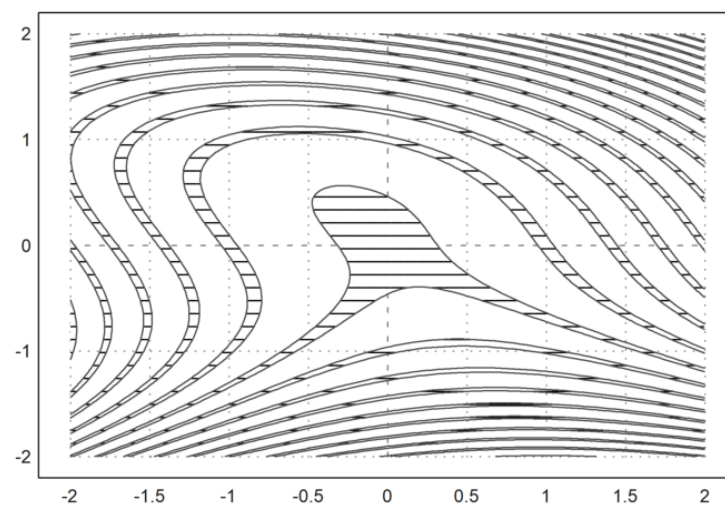
```
>plot2d("x^4+y^4",r=1.5,level=[0;1],color=blue,style="/") :
```



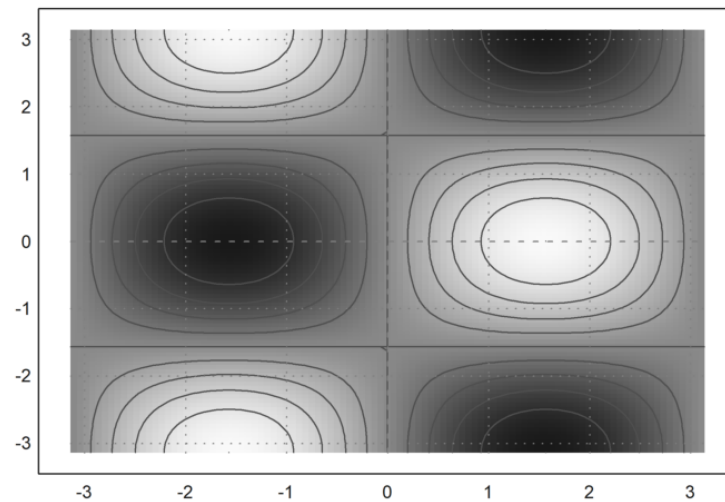
```
>plot2d("x^2+y^3+x*y",level=[0,2,4;1,3,5],style="/",r=2,n=100) :
```



```
>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):
```

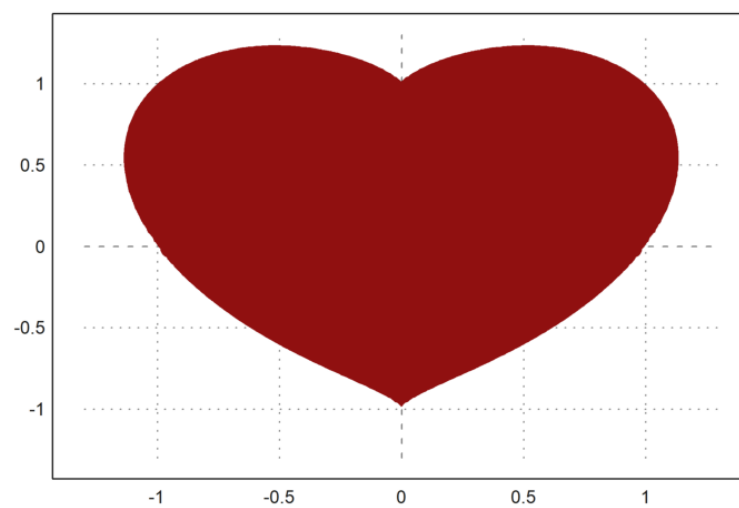



Anda juga dapat menandai suatu wilayah

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

Hal ini dilakukan dengan menambahkan level dengan dua baris.

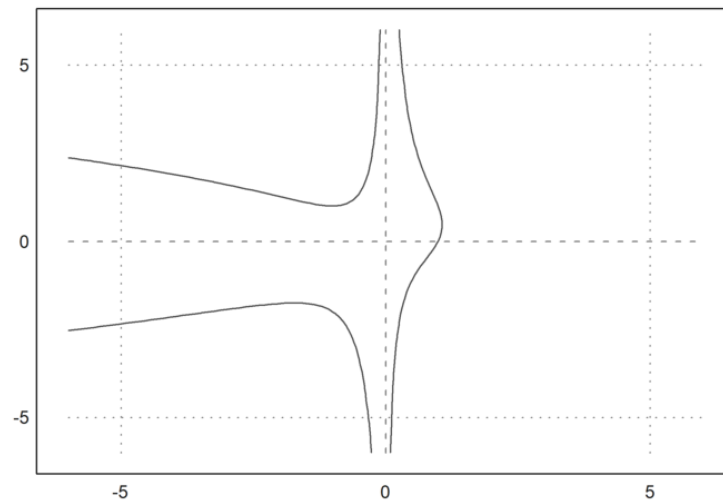
```
>plot2d("(x^2+y^2-1)^3-x^2*y^3",r=1.3, ...
> style="#",color=red,<outline, ...
> level=[-2;0],n=100):
```



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

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

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



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

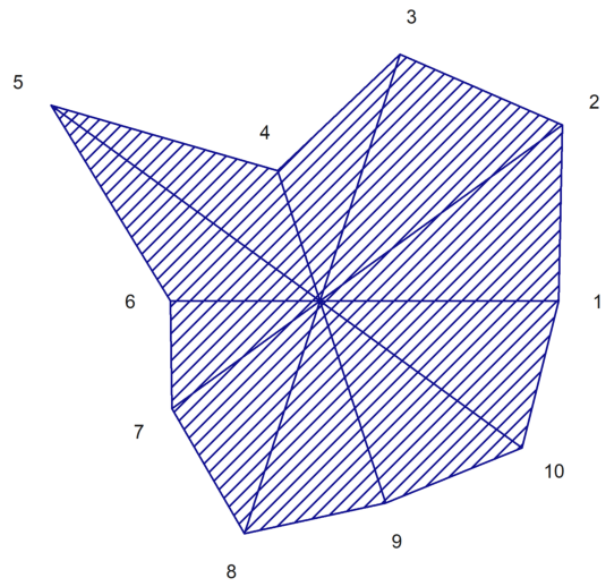
```

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

```

Tidak ada grid atau tanda centang sumbu di sini. Selain itu, kami menggunakan jendela penuh untuk plot. Kami akan melakukan reset sebelum menguji plot ini untuk mengembalikan pengaturan grafis default. Hal ini tidak perlu dilakukan jika Anda yakin plot Anda berfungsi.

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



Terkadang, Anda mungkin ingin merencanakan sesuatu yang tidak dapat dilakukan plot2d, tetapi hampir. Dalam fungsi berikut, kita membuat plot impuls logaritmik. plot2d dapat membuat plot logaritmik, tetapi tidak untuk batang impuls.

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

```

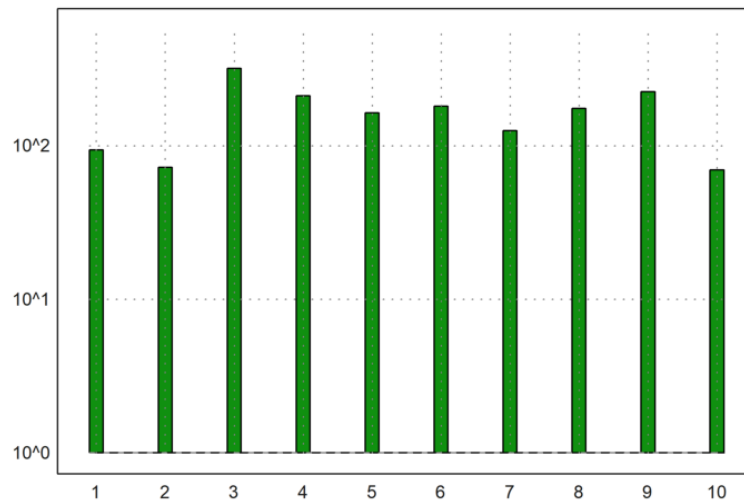
    {x0,y0}=makeimpulse(x,log(y)/log(10));
    plot2d(x0,y0,>bar,grid=0);
    h=holding(1);
    frame();
    xgrid(ticks(x));
    p=plot();
    for i=-10 to 10;
        if i<=p[4] and i>=p[3] then
            ygrid(i,yt="10^"+i);
        endif;
    end;
    holding(h);
endfunction
```

Let us test it with exponentially distributed values.

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

Mari kita mengujinya dengan nilai yang didistribusikan secara eksponensial.

```
>logimpulseplot1(x,y):
```



Mari kita animasikan kurva 2D menggunakan plot langsung. Perintah `plot(x,y)` cukup 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.

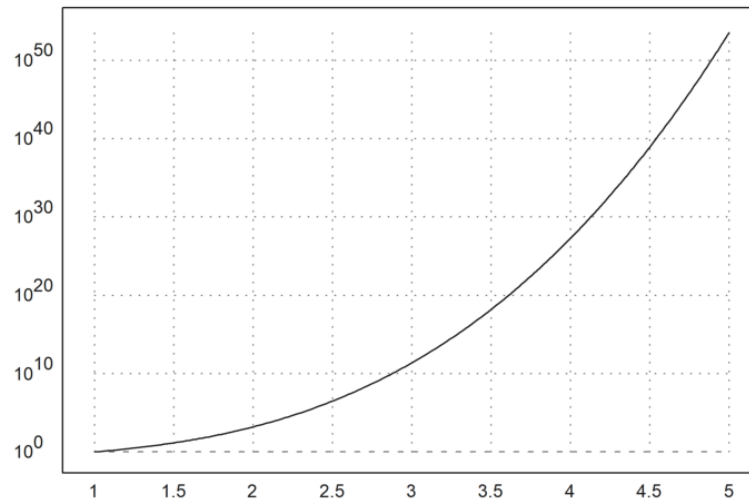
```
>animliss(3,5); // lihat hasilnya, jika sudah puas, tekan ENTER
```

EMT menggunakan parameter "logplot" untuk skala logaritmik.

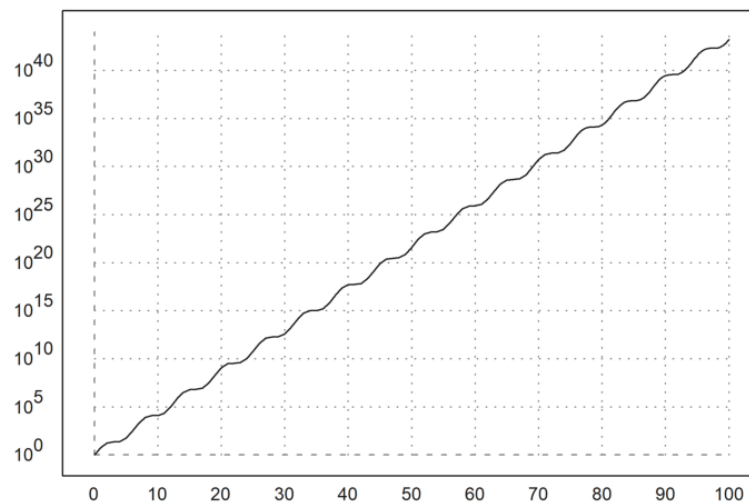
Plot logaritma dapat diplot menggunakan skala logaritma dalam y dengan logplot=1, atau menggunakan skala logaritma dalam x dan y dengan logplot=2, atau dalam x dengan logplot=3.

- logplot=1: y-logarithmic
- logplot=2: x-y-logarithmic
- logplot=3: x-logarithmic

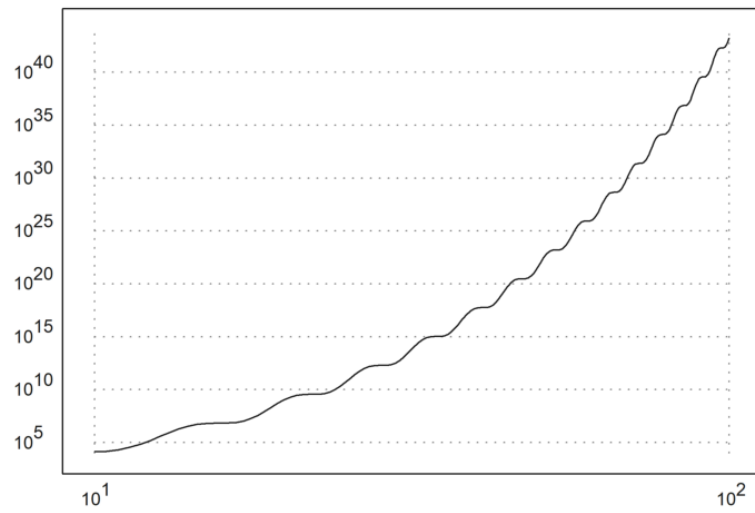
```
>plot2d("exp(x^3-x)*x^2",1,5,logplot=1):
```



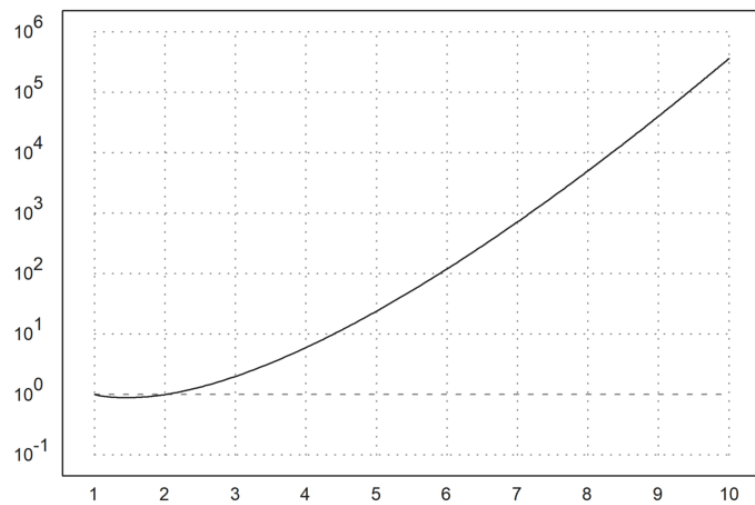
```
>plot2d("exp(x+sin(x))",0,100,logplot=1):
```



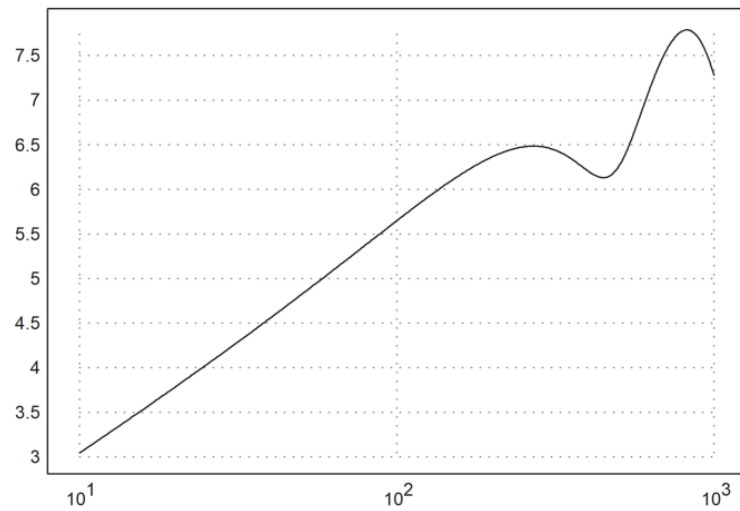
```
>plot2d("exp(x+sin(x))",10,100,logplot=2):
```



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

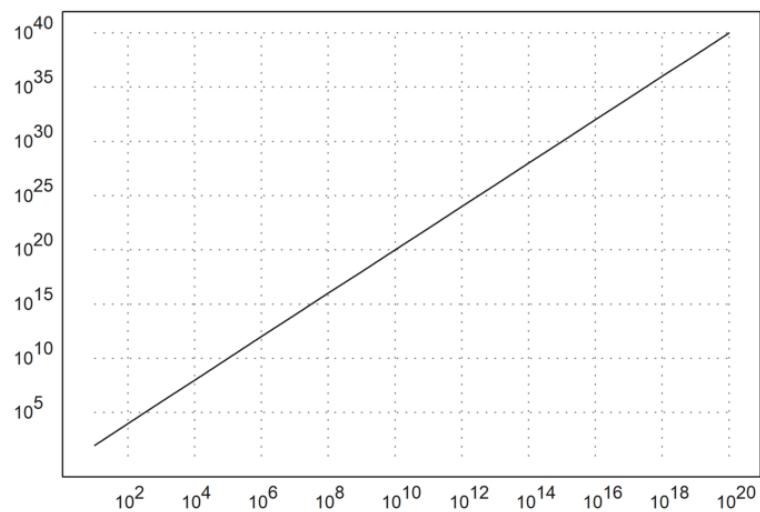


```
>plot2d("log(x*(2+sin(x/100))) ",10,1000,logplot=3):
```



Ini juga berlaku untuk plot data.

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



Rujukan Lengkap Fungsi plot2d()

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

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

Parameters

x,y : equations, functions or data vectors

a,b,c,d : Plot area (default $a=-2,b=2$)

r : if r is set, then $a=cx-r, b=cx+r, c=cy-r, d=cy+r$

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

$xmin,xmax$: range of the parameter for curves

$auto$: Determine y-range automatically (default)

$square$: if true, try to keep square x-y-ranges

n : number of intervals (default is adaptive)

$grid$: 0 = no grid and labels,

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

$frame$: 0 = no frame

$framecolor$: color of the frame and the grid

$margin$: number between 0 and 0.4 for the margin around the plot

$color$: Color of curves. If this is a vector of colors,

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

$thickness$: line thickness for curves

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

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

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

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

value

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

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

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

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

own :

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

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

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

cgrid :

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

Overview

The function can plot

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

If a function or expression for `xv` is given, `plot2d()` will compute values in the given range using the function or expression. The expression must be an expression in the variable `x`. The range must be defined in the parameters `a` and `b` unless the default range should be used. The y-range will be computed automatically, unless `c` and `d` are specified, or a radius `r`, which yields the range `r,r`

for `x` and `y`. For plots of functions, `plot2d` will use an adaptive evaluation of the function by default. To speed up the plot for complicated functions, switch this off with `<adaptive`, and optionally decrease the number of intervals `n`. Moreover, `plot2d()` will by default use mapping. I.e., it will compute the plot element for element. If your expression or your functions can handle a vector `x`, you can switch that off with `<maps` for faster evaluation.

Note that adaptive plots are always computed element for element.

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