

Tugas Aplikasi Komputer

Nama : Theresia Selvina Vanny M.
NIM : 22305141029
Kelas : Matematika B

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-fungsi dasar dari plot. Ada koordinat layar, yang selalu berkisar dari 0 hingga 1024 di setiap sumbu, tidak peduli apakah layar tersebut berbentuk persegi atau tidak. Dan ada koordinat plot, yang dapat diatur dengan menggunakan `setplot()`. Pemetaan antara koordinat-koordinat ini bergantung pada jendela plot saat ini. Misalnya, `shrinkwindow()` default menyisakan ruang untuk label sumbu dan judul plot.

Pada contoh ini, kami hanya menggambar beberapa garis acak dengan berbagai warna. Untuk detail tentang fungsi-fungsi ini, pelajari fungsi-fungsi inti dari EMT.

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



```
>reset;
```

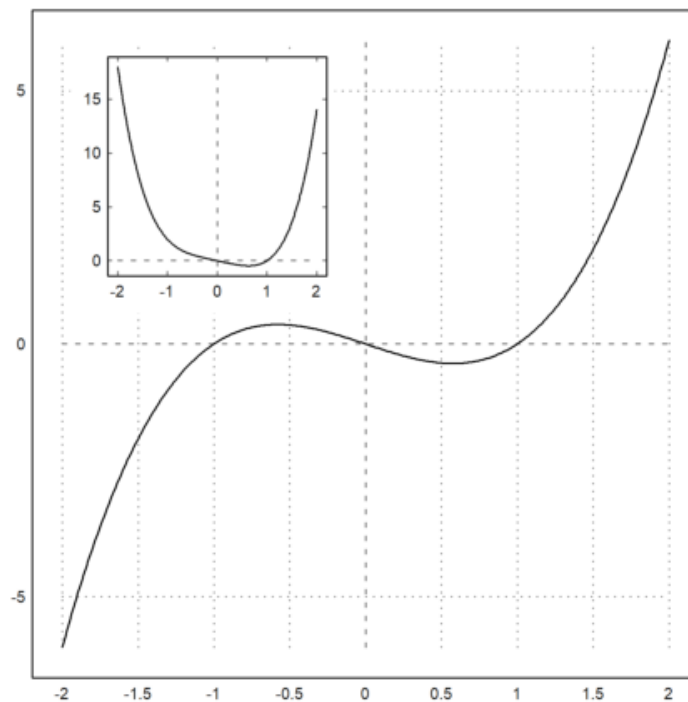
Penting untuk menahan grafik, karena perintah `plot()` akan menghapus jendela plot.

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

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

Sebagai contoh lain, kita menggambar plot sebagai sebuah inset dalam plot lainnya. Ini dilakukan dengan menentukan jendela plot yang lebih kecil. Perhatikan bahwa jendela ini tidak memberikan ruang untuk label sumbu di luar jendela plot. Kita harus menambahkan beberapa margin sesuai kebutuhan. Perhatikan juga bahwa kita menyimpan dan mengembalikan jendela penuh, dan menahan plot saat ini saat kita menggambar 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);
```

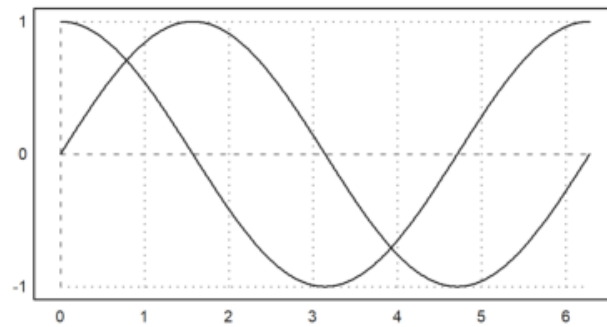
Grafik dengan beberapa gambar dapat dicapai dengan cara yang sama. Ada fungsi utilitas `figure()` untuk ini.

Aspek Plot

Plot default menggunakan jendela plot berbentuk persegi. Anda dapat mengubahnya dengan menggunakan fungsi `aspect()`. Jangan lupa untuk mengembalikan aspek ini ke setelan default nanti. Anda juga dapat mengubah aspek default ini dalam menu dengan "Set Aspek" menjadi rasio aspek tertentu atau sesuai dengan ukuran saat ini dari jendela grafik.

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

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



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

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

Plot 2D dalam Euler

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

Mungkin untuk melakukan plotting dalam Maxima menggunakan Gnuplot atau dalam Python menggunakan Math Plot Lib.

Euler dapat melakukan plotting plot 2D dari:

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

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

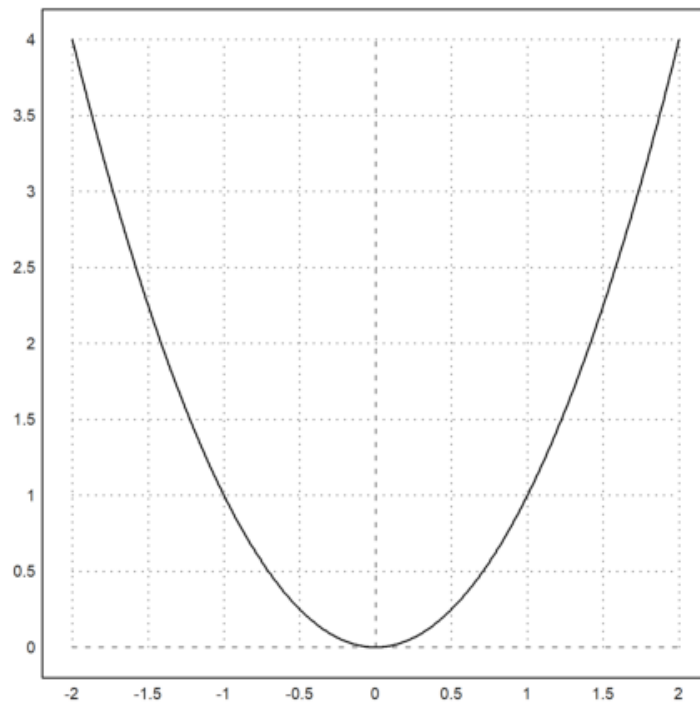
Plot Ekspresi atau Variabel

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

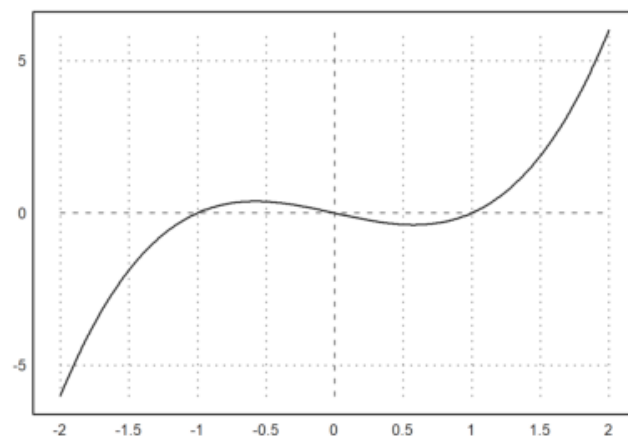
Berikut adalah contoh paling dasar, yang menggunakan rentang default dan mengatur rentang y yang sesuai untuk menyesuaikan plot fungsi.

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

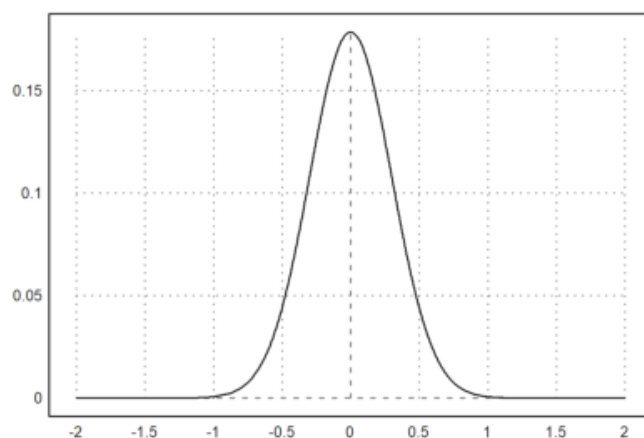
```
>plot2d("x^2"):
```



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



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

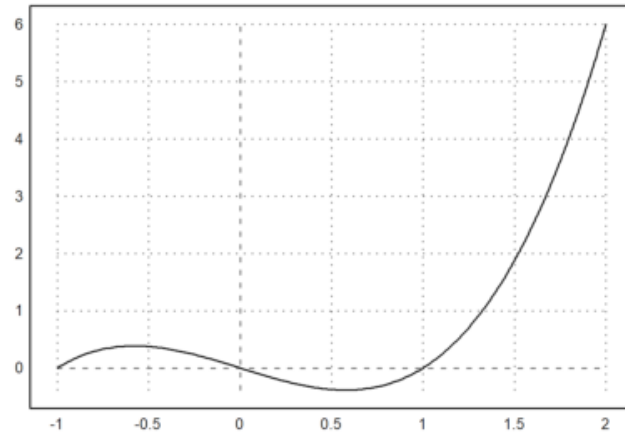


Dari beberapa contoh sebelumnya Anda dapat melihat bahwa aslinya gambar plot menggunakan sumbu X dengan rentang nilai dari -2 sampai dengan 2. Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan nilai-nilai batas X (dan Y) di belakang ekspresi yang digambar.

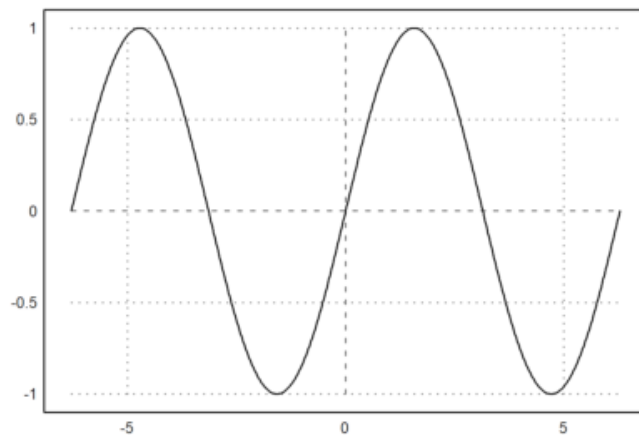
Rentang plot dapat diatur dengan parameter-parameter yang ditugaskan berikut:

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

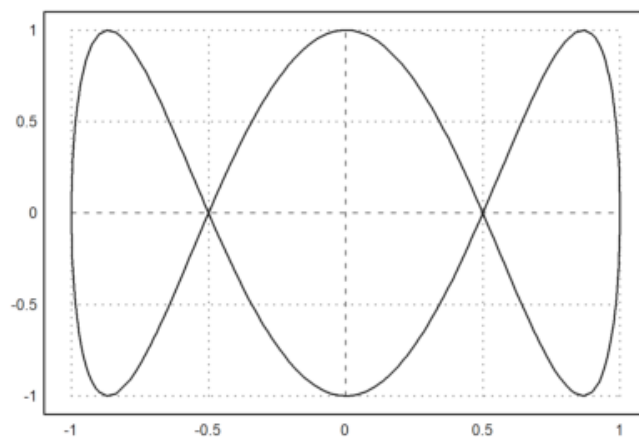
```
>plot2d("x^3-x",-1,2):
```



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



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



Sebagai alternatif dari tanda titik dua (:), Anda dapat menggunakan perintah `insimg(lines)`, yang memasukkan plot dan mengisi jumlah baris teks yang ditentukan.

Dalam opsi, plot dapat diatur untuk muncul:

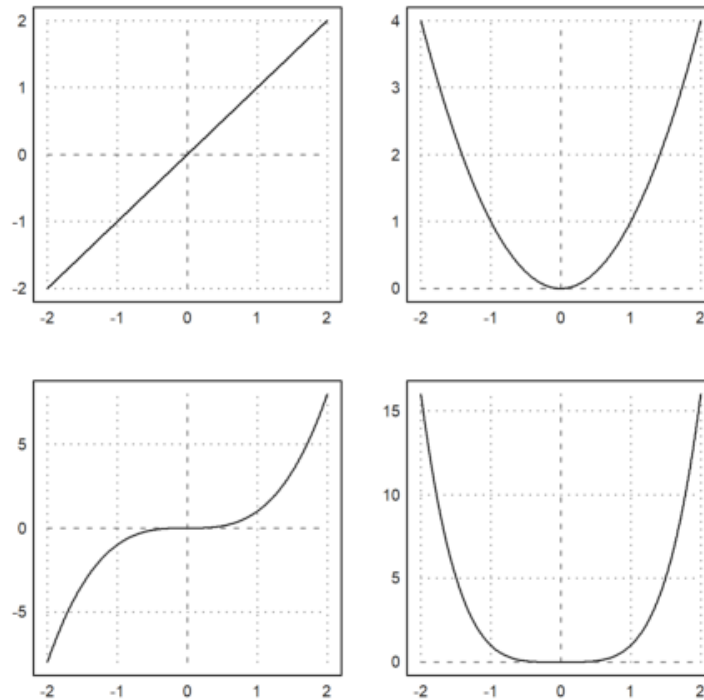
- dalam jendela terpisah yang dapat diubah ukurannya,
- dalam jendela notebook.

Gaya-gaya tambahan dapat dicapai dengan perintah plot khusus.

Dalam semua kasus, tekan tombol tabulator untuk melihat plot jika itu tersembunyi.

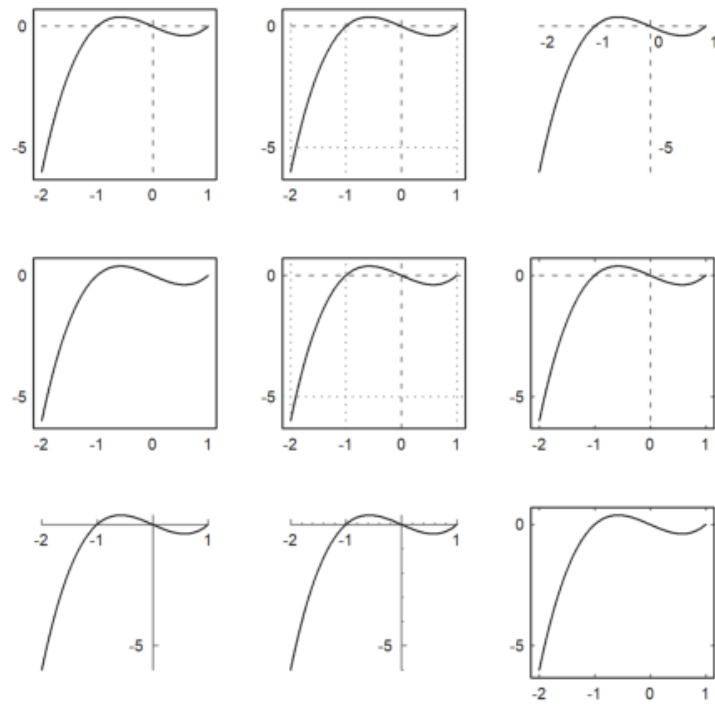
Untuk membagi jendela menjadi beberapa plot, gunakan perintah `figure()`. Dalam contoh tersebut, kita memplot x^1 hingga x^4 ke dalam 4 bagian jendela. `figure(0)` mengatur ulang jendela default.

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



Dalam `plot2d()`, terdapat gaya-gaya alternatif yang tersedia dengan `grid=x`. Untuk gambaran umum, kami menunjukkan berbagai gaya grid dalam satu gambar (lihat di bawah untuk perintah `figure()`). Gaya `grid=0` tidak termasuk. Ini tidak menampilkan grid dan frame.

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

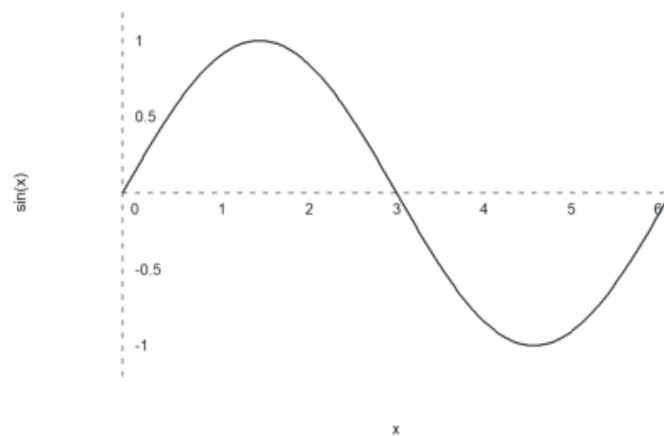


Jika argumen-argumen untuk `plot2d()` adalah ekspresi yang diikuti oleh empat angka, angka-angka ini adalah rentang x dan y untuk plot tersebut.

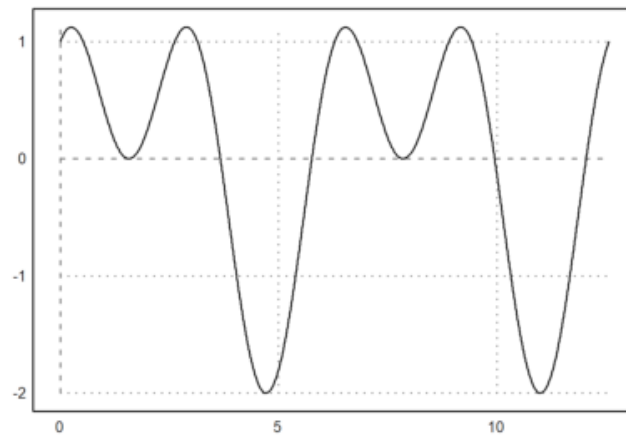
Atau, a , b , c , d dapat ditentukan sebagai parameter yang ditugaskan, seperti `a=... dll`.

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

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



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

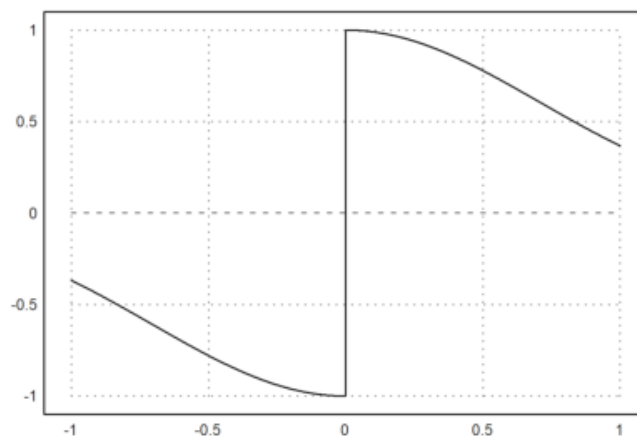


Gambar yang dihasilkan dengan menyisipkan plot ke dalam jendela teks disimpan dalam direktori yang sama dengan notebook, secara default dalam subdirektori yang bernama "images". Gambar-gambar ini juga digunakan oleh ekspor HTML.

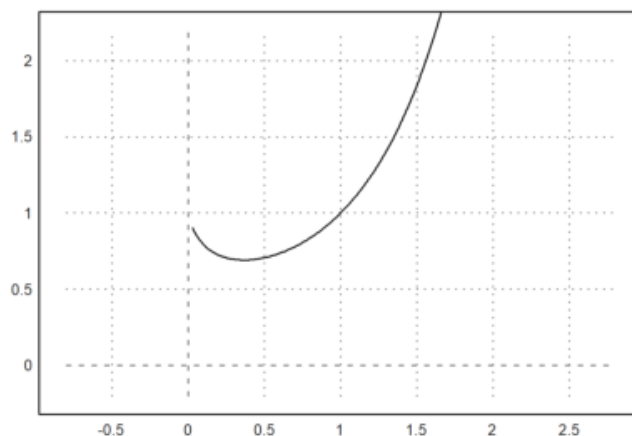
Anda dapat dengan mudah menandai gambar apa pun dan menyalinnya ke clipboard dengan Ctrl-C. Tentu saja, Anda juga dapat mengeksport grafik saat ini dengan fungsi-fungsi dalam menu File.

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

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



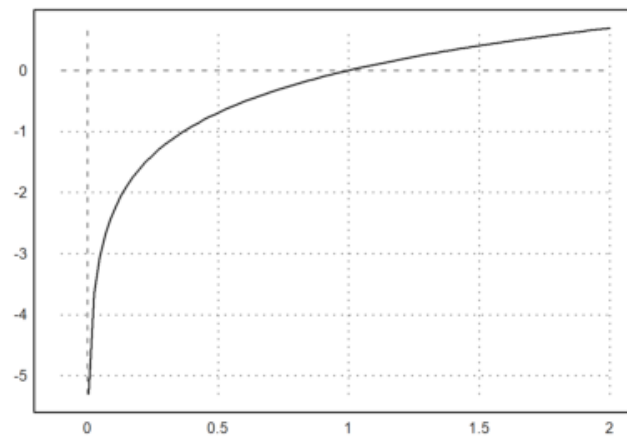
```
>plot2d("x^x",r=1.2,cx=1,cy=1):
```



Perlu dicatat bahwa x^x tidak terdefinisi untuk $x \leq 0$. Fungsi plot2d menangkap kesalahan ini dan mulai memplot segera setelah fungsi didefinisikan. Ini berlaku untuk semua fungsi yang

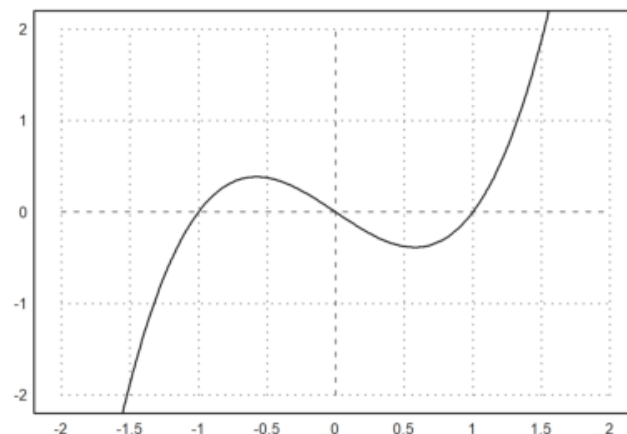
mengembalikan nilai NAN di luar rentang definisinya.

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

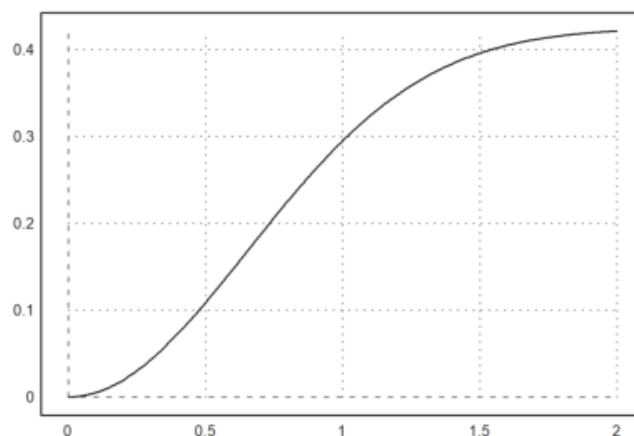


Parameter `square=true` (atau `>square`) memilih rentang y secara otomatis sehingga hasilnya adalah jendela plot berbentuk persegi. Perlu diingat bahwa secara default, Euler menggunakan ruang berbentuk persegi di dalam jendela plot.

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

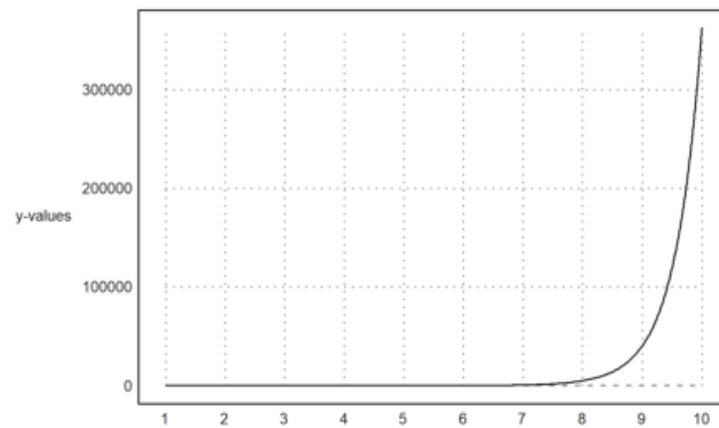


```
>plot2d('integrate("sin(x)*exp(-x^2)","0,x")',0,2): // plot integral
```



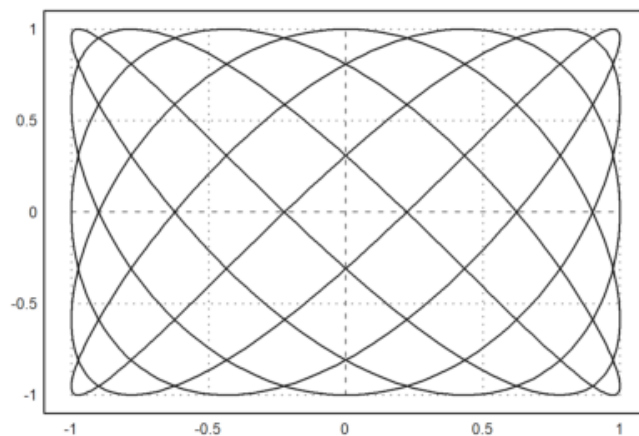
Jika Anda memerlukan lebih banyak ruang untuk label-label sumbu y, panggil `shrinkwindow()` dengan parameter yang lebih kecil, atau atur nilai positif untuk "smaller" dalam `plot2d()`. Ini akan membantu Anda mengatur lebih banyak ruang untuk label-label tersebut dalam jendela plot.

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

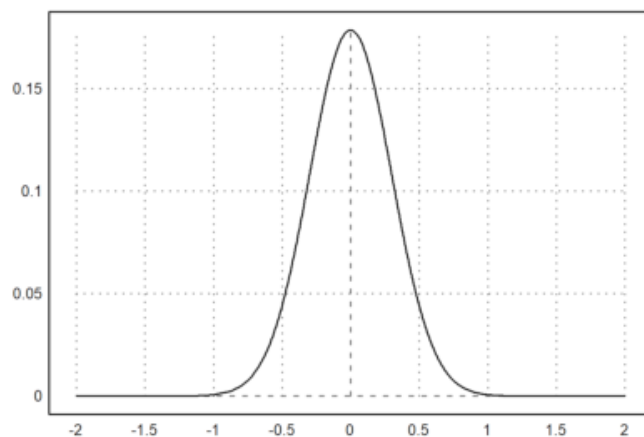


Ekspresi simbolik juga dapat digunakan, karena mereka disimpan sebagai ekspresi string sederhana.

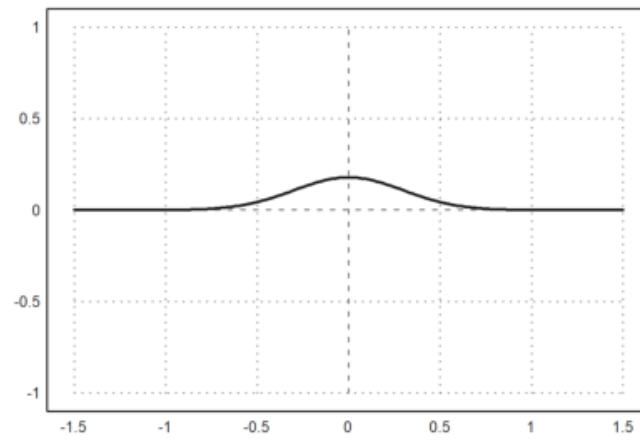
```
>x=linspace(0,2pi,1000); plot2d(sin(5x),cos(7x)):
```



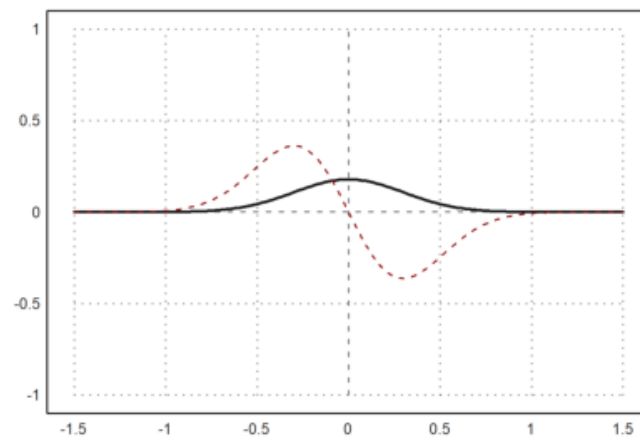
```
>a:=5.6; expr &= exp(-a*x^2)/a; // define expression
>plot2d(expr,-2,2): // plot from -2 to 2
```



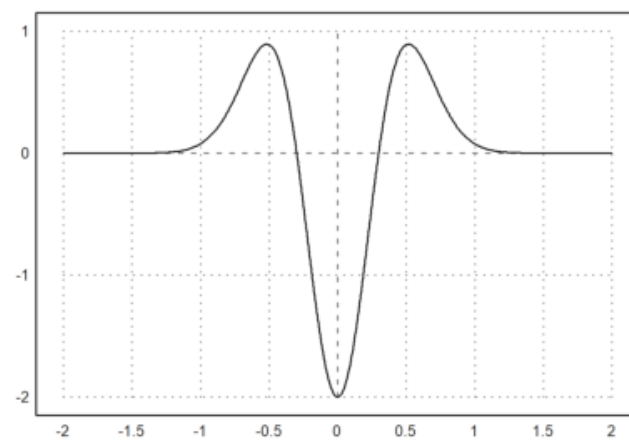
```
>plot2d(expr,r=1,thickness=2): // plot in a square around (0,0)
```



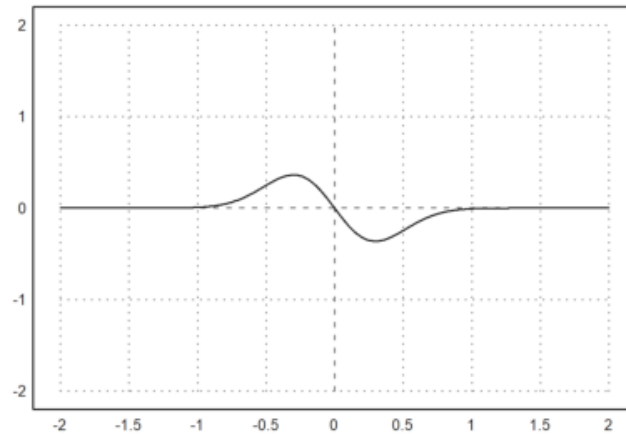
```
>plot2d(&diff(expr,x),>add,style="--",color=red): // add another plot
```



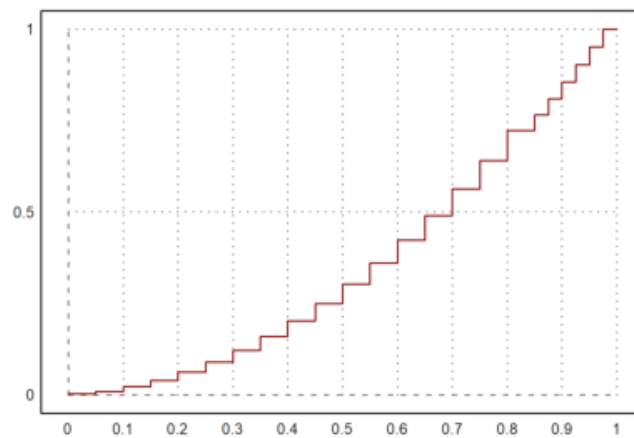
```
>plot2d(&diff(expr,x,2),a=-2,b=2,c=-2,d=1): // plot in rectangle
```



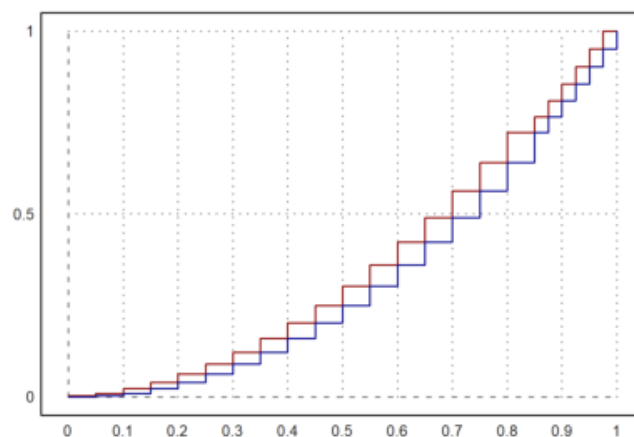
```
>plot2d(&diff(expr,x),a=-2,b=2,>square): // keep plot square
```



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



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



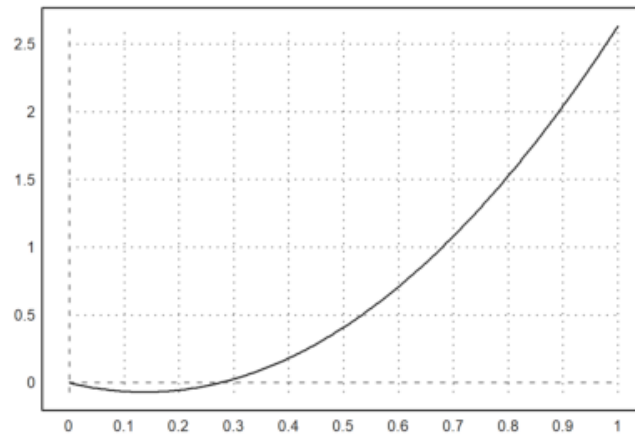
Fungsi dalam Satu Parameter

Fungsi plotting paling penting untuk plot dalam dua dimensi adalah `plot2d()`. Fungsi ini diimplementasikan dalam bahasa Euler dalam berkas "plot.e", yang dimuat saat program dimulai.

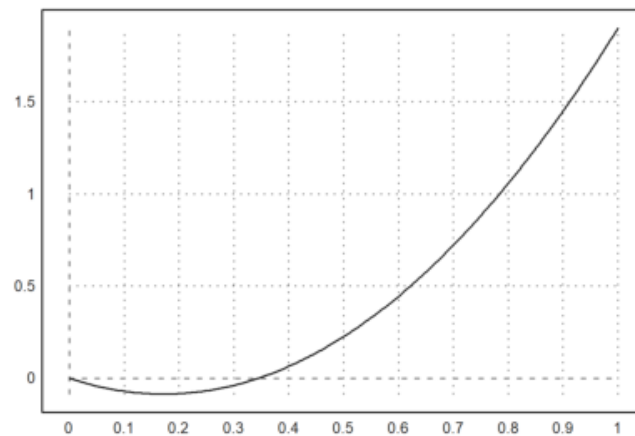
Berikut beberapa contoh menggunakan sebuah fungsi. Seperti biasa dalam EMT, fungsi-fungsi yang berfungsi untuk fungsi atau ekspresi lainnya, Anda dapat mengirimkan parameter tambahan (selain x) ke fungsi tersebut dengan menggunakan parameter semikolon atau dengan koleksi panggilan.

```
>function f(x,a) := x^2/a+a*x^2-x; // define a function
```

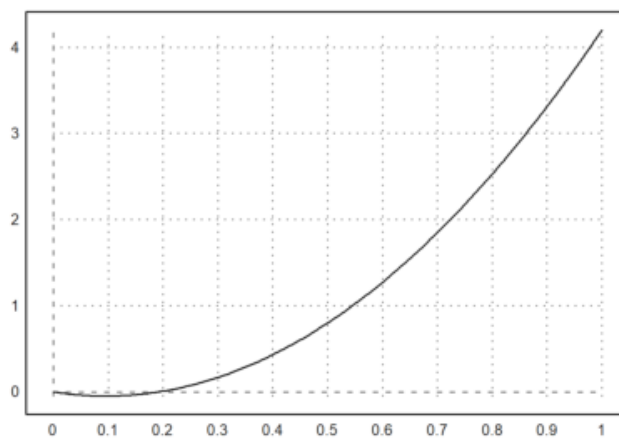
```
>a=0.3; plot2d("f",0,1;a): // plot with a=0.3
```



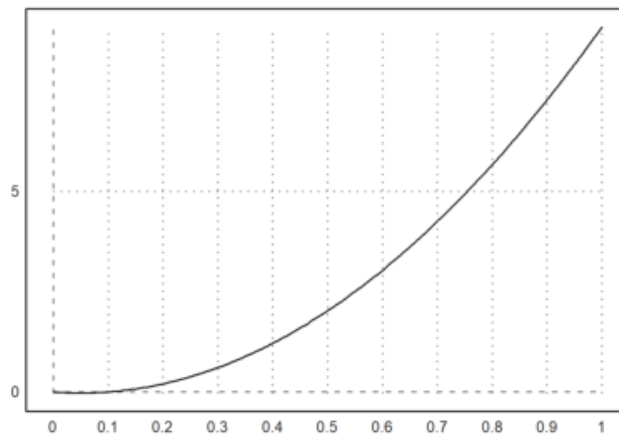
```
>plot2d("f",0,1;0.4): // plot with a=0.4
```



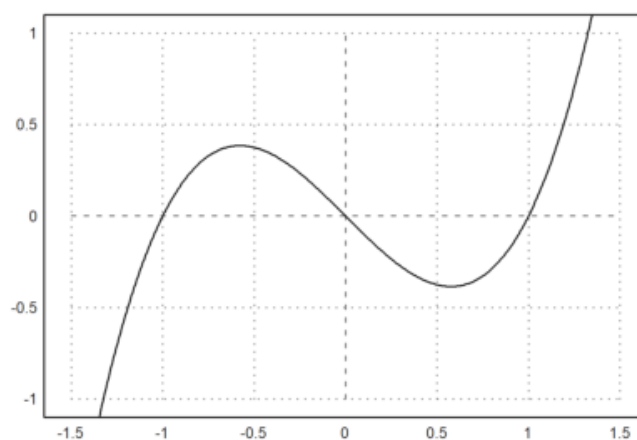
```
>plot2d({"f",0.2},0,1): // plot with a=0.2
```



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



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



Berikut adalah ringkasan fungsi-fungsi yang dapat diterima:

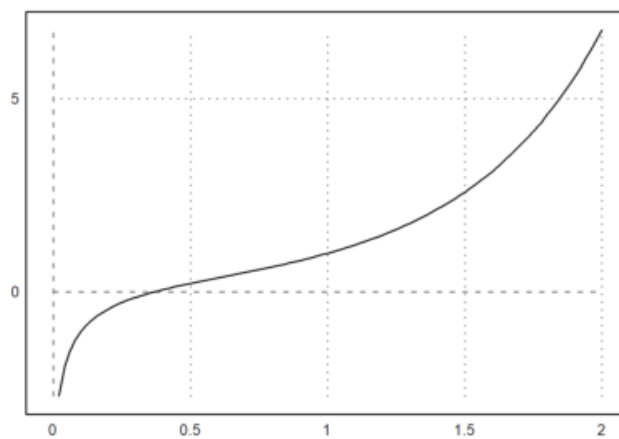
- ekspresi atau ekspresi simbolik dalam x
- fungsi atau fungsi simbolik berdasarkan nama seperti "f"
- fungsi simbolik hanya berdasarkan nama f

Fungsi plot2d() juga menerima fungsi simbolik. Untuk fungsi simbolik, hanya nama tersebut yang diperlukan.

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

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

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

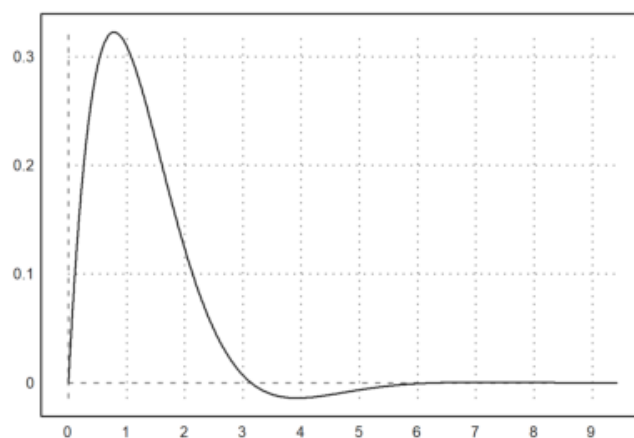


Tentu saja, untuk ekspresi atau ekspresi simbolik, nama variabelnya sudah cukup untuk memplotnya.

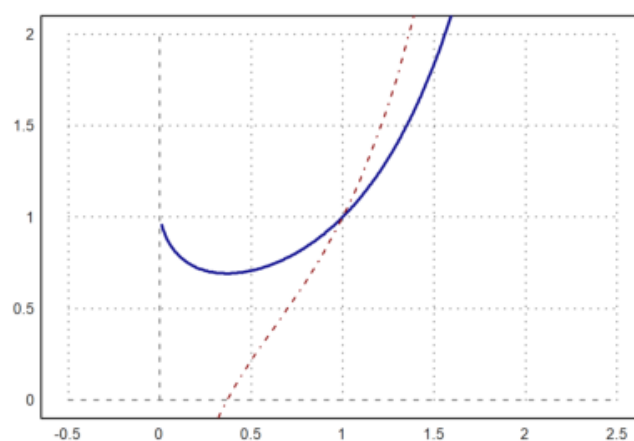
```
>expr &= sin(x)*exp(-x)
```

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

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



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



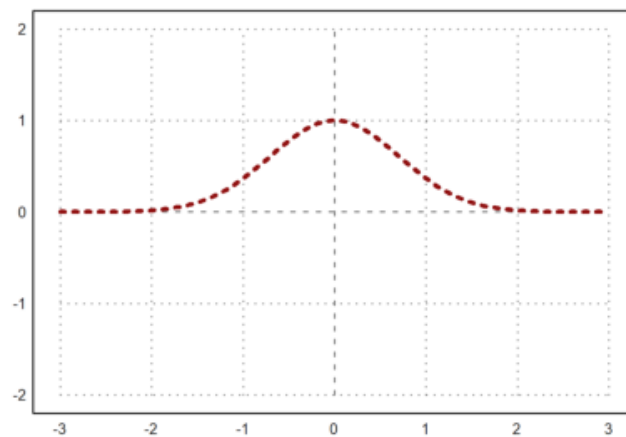
Untuk gaya garis, ada berbagai pilihan:

- `style="..."`. Pilih dari "-", "--", "-.", ".", "-.-", "-.-".
- warna: Lihat di bawah untuk warna-warna.
- ketebalan: Defaultnya adalah 1.

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

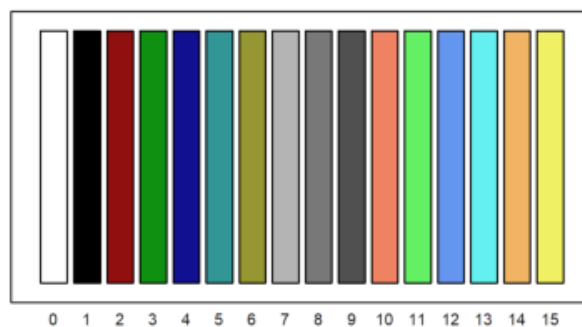
- 0..15: indeks warna default.
- konstanta warna: white (putih), black (hitam), red (merah), green (hijau), blue (biru), cyan (sian), olive (zaitun), lightgray (abu-abu terang), gray (abu-abu), darkgray (abu-abu gelap), orange (jeruk), lightgreen (hijau terang), turquoise (turkis), lightblue (biru terang), lightorange (jeruk terang), yellow (kuning)
- `rgb(red, green, blue)`: parameter adalah bilangan real dalam [0,1].

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



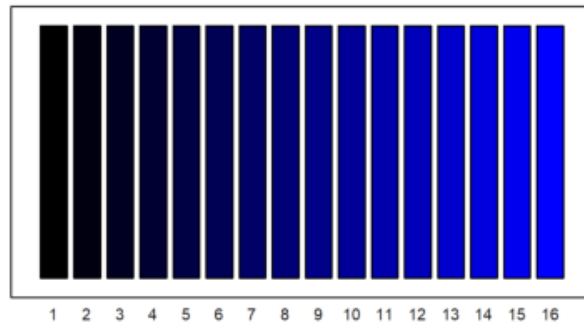
Berikut adalah tampilan warna-warna yang telah ditentukan sebelumnya dalam EMT.

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



Tapi Anda dapat menggunakan warna lain

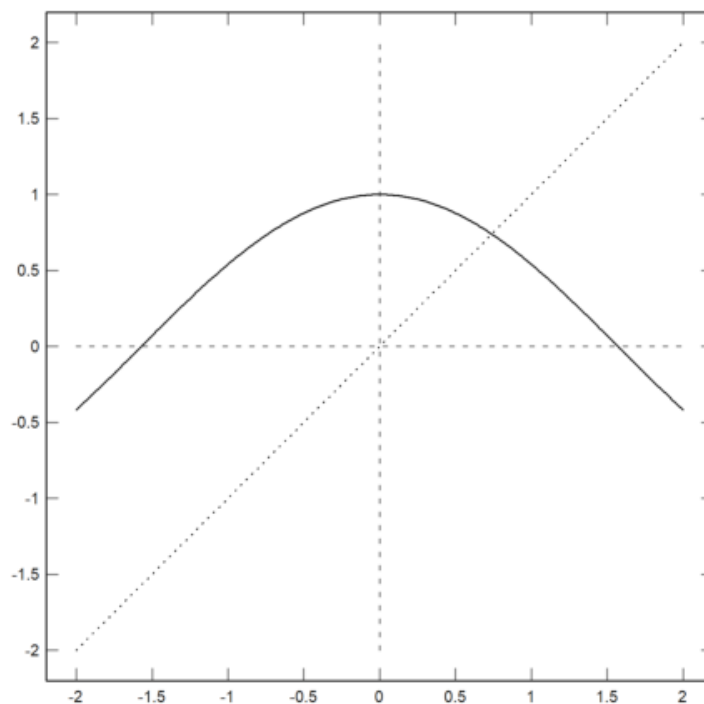
```
>columnsplot(ones(1,16),grid=0,color=rgb(0,0,linspace(0,1,15))):
```

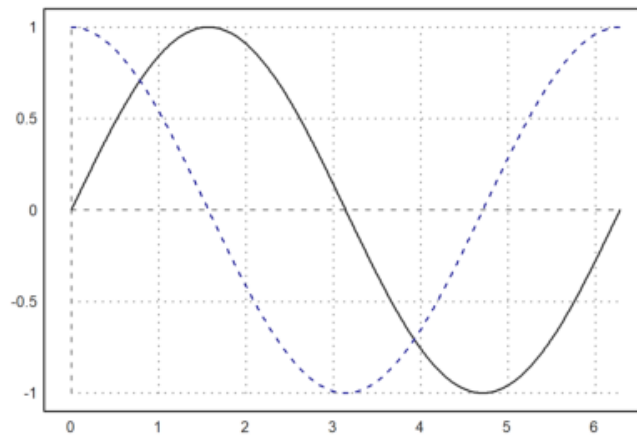
Menggambar Beberapa Kurva pada bidang koordinat yang sama

Menggambar lebih dari satu fungsi (beberapa fungsi) dalam satu jendela dapat dilakukan dengan berbagai cara. Salah satu metode adalah dengan menggunakan `>add` untuk beberapa panggilan `plot2d` pada semua panggilan, kecuali panggilan pertama. Kami telah menggunakan fitur ini dalam contoh-contoh di atas.

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

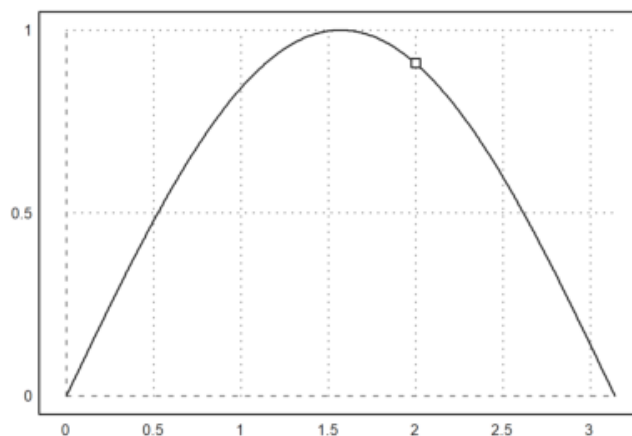


```
>aspect(1.5); plot2d("sin(x)",0,2pi); plot2d("cos(x)",color=blue,style="--",>add):
```



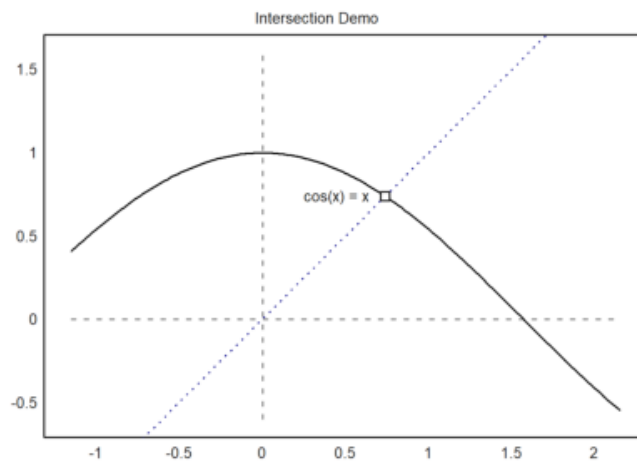
Salah satu kegunaan `>add` adalah untuk menambahkan titik pada kurva.

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



Kami menambahkan titik potong dengan label (pada posisi "cl" untuk tengah kiri), dan menyisipkan hasilnya ke dalam notebook. 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, kami memplot fungsi $\text{sinc}(x) = \sin(x)/x$ dan ekspansi Taylor ke-8 dan ke-16 nya. Kami menghitung ekspansi ini menggunakan Maxima melalui ekspresi simbolik.

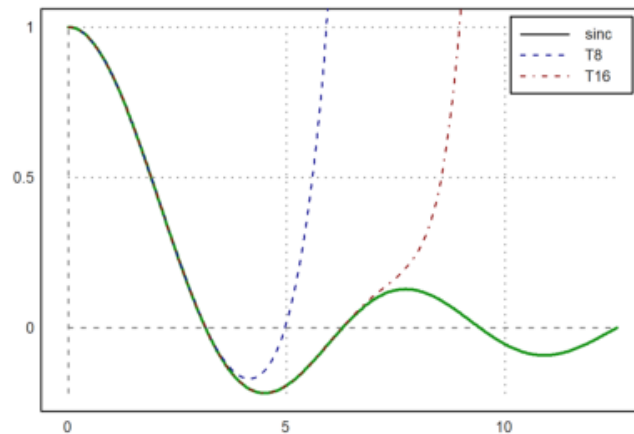
Plot ini dilakukan dalam perintah multi-baris berikut dengan tiga panggilan plot2d(). Panggilan kedua dan ketiga memiliki flag >add yang diatur, sehingga plot menggunakan rentang sebelumnya.

Kami menambahkan kotak label yang menjelaskan fungsi-fungsi tersebut.

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

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

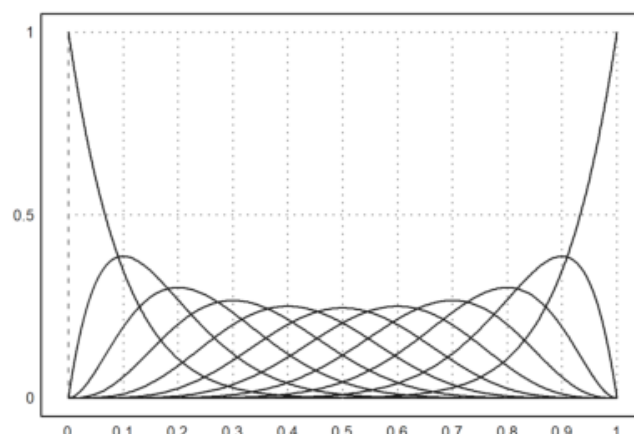
```
>plot2d("sinc(x)",0,4pi,color=green,thickness=2); ...
  plot2d(&taylor(sin(x)/x,x,0,8),>add,color=blue,style="--"); ...
  plot2d(&taylor(sin(x)/x,x,0,16),>add,color=red,style="-.-"); ...
  labelbox(["sinc","T8","T16"],styles=["-","--","-.-"], ...
    colors=[black,blue,red]):
```



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

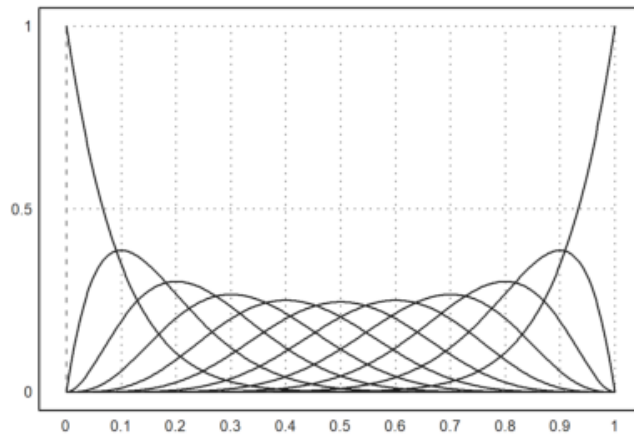


Metode kedua adalah menggunakan pasangan matriks nilai-nilai x dan matriks nilai-nilai y dengan ukuran yang sama.

Kami menghasilkan matriks nilai dengan satu Polinomial Bernstein dalam setiap baris. Untuk ini, kami hanya menggunakan vektor kolom i. Silakan lihat pengantar tentang bahasa matriks untuk mempelajari lebih lanjut tentang rincian ini.

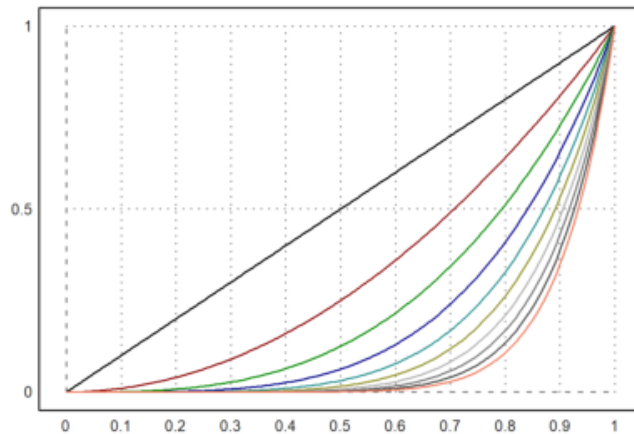
```
>x=linspace(0,1,500);
>n=10; k=(0:n)'; // n is row vector, k is column vector
```

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



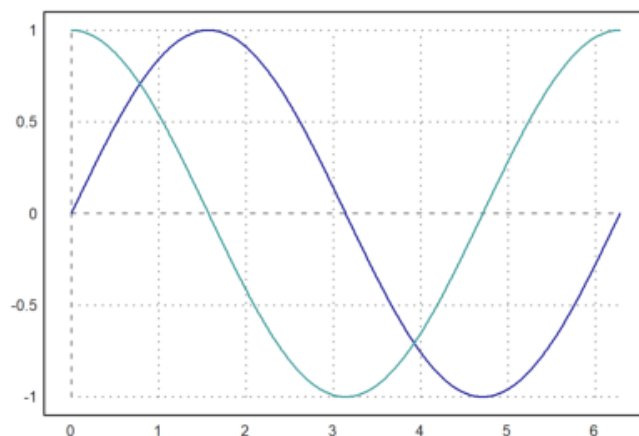
Perlu diperhatikan bahwa parameter warna dapat berupa vektor. Maka, setiap warna digunakan untuk setiap baris dalam matriks tersebut.

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

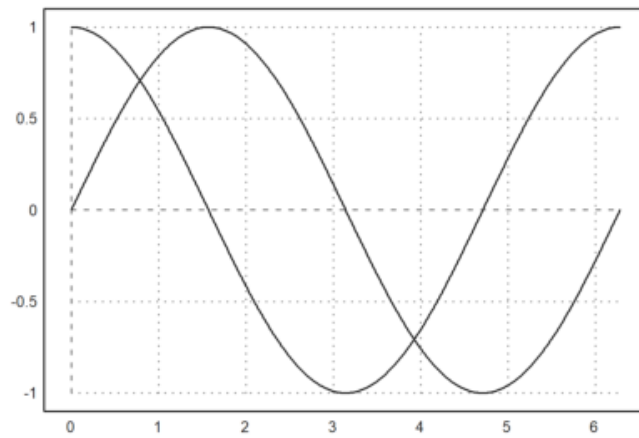


Metode lainnya adalah menggunakan vektor ekspresi (string). Anda dapat menggunakan vektor warna, vektor gaya, dan vektor ketebalan dengan panjang yang sama.

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



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



Anda dapat mendapatkan vektor seperti itu dari Maxima menggunakan `makelist()` dan `mxm2str()`.

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

```

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

```

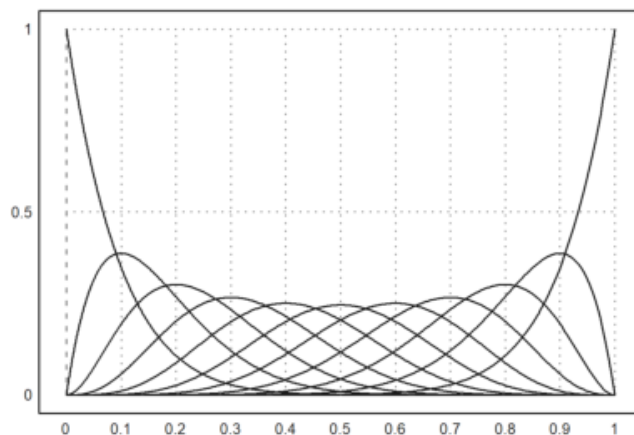
```
>mxm2str(v) // get a vector of strings from the symbolic vector
```

```

(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
```

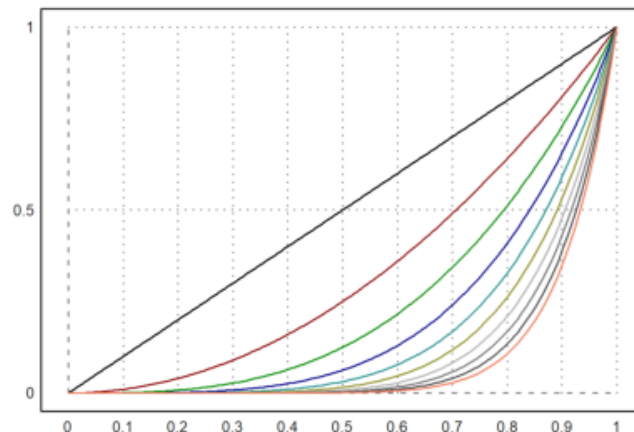


Alternatif lainnya adalah menggunakan bahasa matriks Euler.

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

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

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

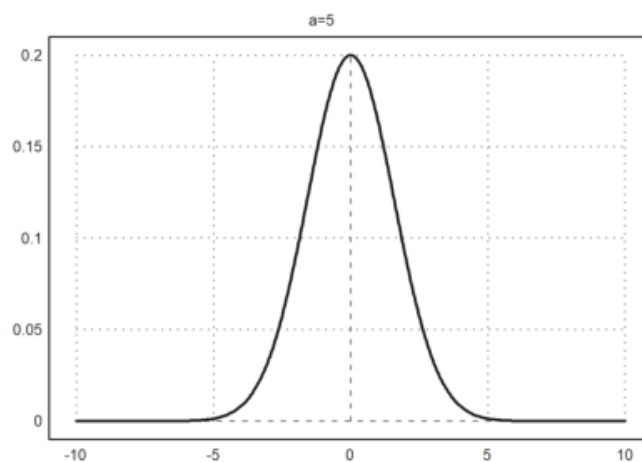


Ekspresi dan fungsi satu baris dapat melihat variabel global.

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

Pastikan untuk menempatkan semua parameter yang ditugaskan di akhir perintah plot2d. Pada contoh ini, kami meneruskan a=5 ke fungsi f, yang kami plot dari -10 hingga 10.

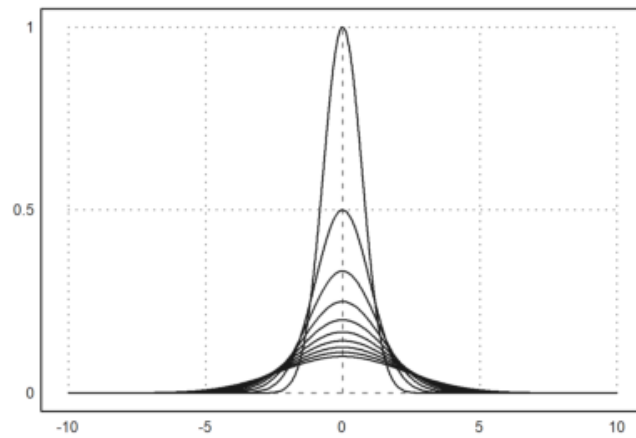
```
>function f(x,a) := 1/a*exp(-x^2/a); ...
plot2d("f",-10,10;5,thickness=2,title="a=5"):
```



Sebagai alternatif, Anda dapat menggunakan koleksi dengan nama fungsi dan semua parameter tambahan. Daftar istimewa ini disebut koleksi panggilan, dan ini adalah cara yang disarankan untuk meneruskan argumen ke fungsi yang sendiri diteruskan sebagai argumen ke fungsi lain.

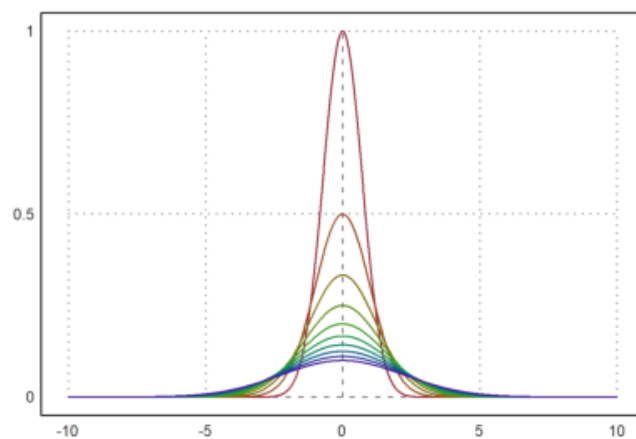
Pada contoh berikut, kami menggunakan perulangan untuk memplot beberapa fungsi (lihat tutorial tentang pemrograman perulangan).

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



Kami dapat mencapai hasil yang sama dengan cara berikut menggunakan bahasa matriks EMT. Setiap baris matriks $f(x, a)$ adalah satu fungsi. Selain itu, kita dapat mengatur warna untuk setiap baris matriks. Double klik pada fungsi `getspectral()` untuk penjelasan lebih lanjut.

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



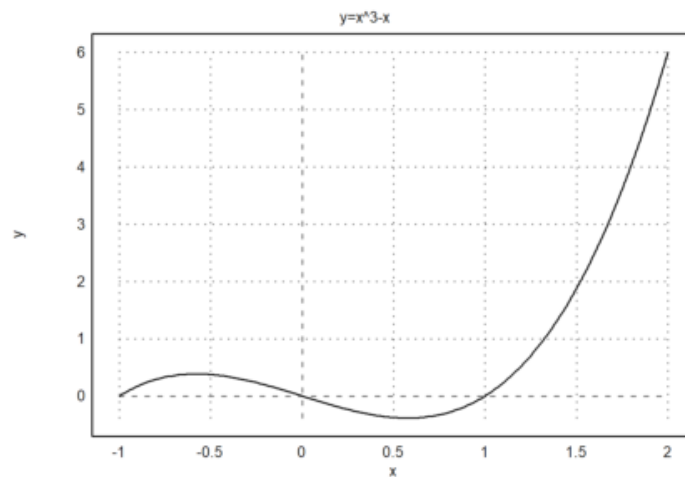
Label Teks

Dekorasi sederhana dapat berupa:

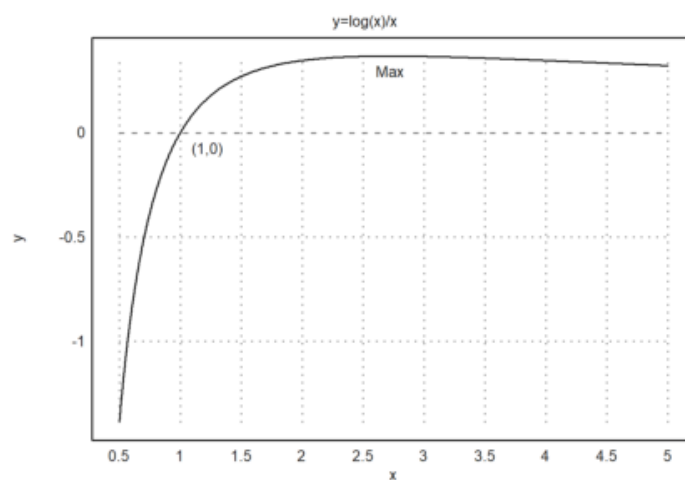
- judul dengan `title="..."`
- label x dan y dengan `xl="..."`, `yl="..."`
- label teks lain dengan `label("...",x,y)`

Perintah `label` akan diplot ke dalam plot saat ini pada koordinat plot (x, y). Ini dapat mengambil argumen posisi.

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

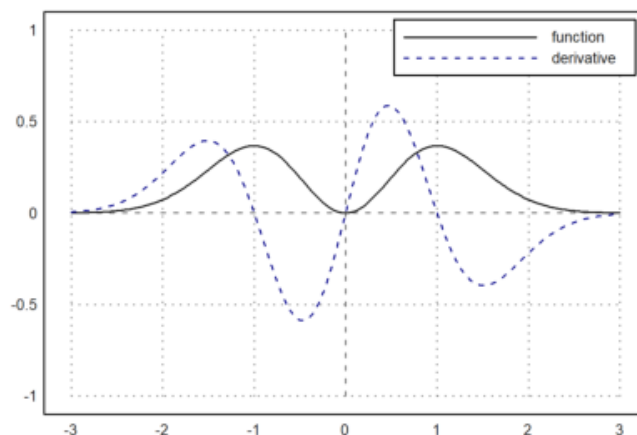


```
>expr := "log(x)/x"; ...
plot2d(expr,0.5,5,title="y="+expr,xl="x",yl="y"); ...
label("(1,0)",1,0); label("Max",E,expr(E),pos="lc"):
```



Terdapat juga fungsi `labelbox()`, yang dapat menampilkan fungsi-fungsi dan teks. Ini menerima vektor-vektor string dan warna-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):
```

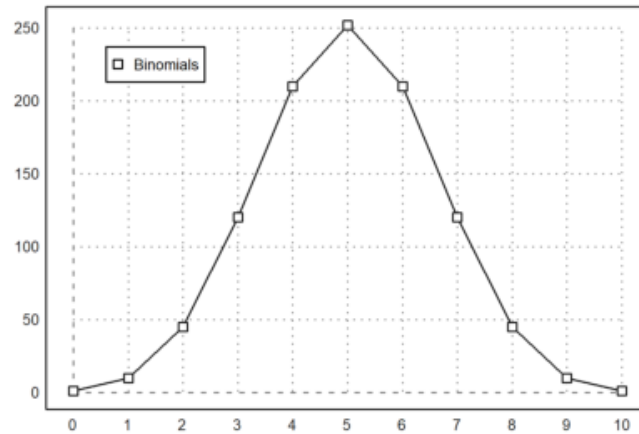


Kotak biasanya di-ancor di sudut kanan atas secara default, tetapi `>left` akan meng-ancor di sudut kiri atas. Anda dapat memindahkannya ke tempat mana saja yang Anda inginkan. Posisi anchor adalah sudut kanan atas dari kotak, dan angka-angka tersebut merupakan pecahan dari ukuran jendela grafis. Lebar kotak adalah otomatis.

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

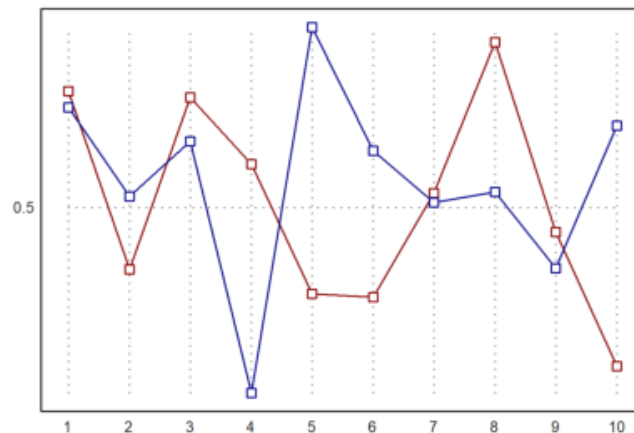
Pada contoh berikut, hanya ada satu fungsi. Oleh karena itu, kami dapat menggunakan string alih-alih vektor string. Kami juga mengatur warna teksnya menjadi hitam untuk contoh ini.

```
>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 dalam `statplot()`. Seperti pada `plot2d()`, warna dapat diatur untuk setiap baris dalam plot. Ada lebih banyak plot khusus untuk tujuan statistik (lihat tutorial tentang statistik).

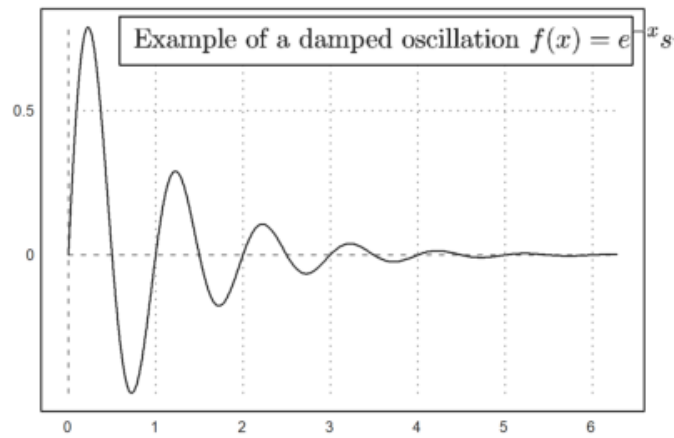
```
>statplot(1:10,random(2,10),color=[red,blue]):
```



Fitur serupa adalah fungsi `textbox()`.

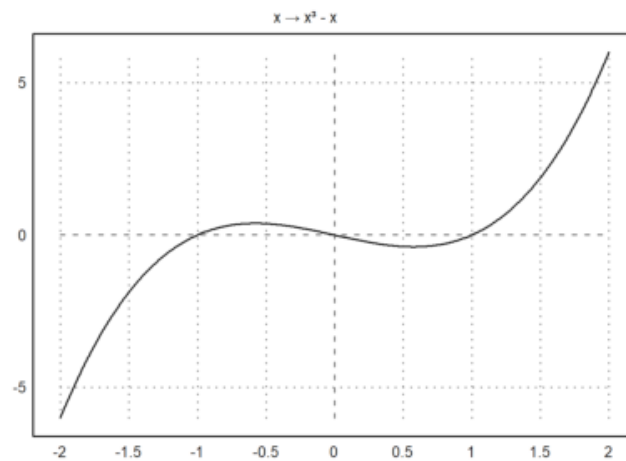
Lebar kotak teks secara default adalah lebar maksimal dari baris-baris teks. Tetapi pengguna juga dapat mengaturnya sendiri.

```
>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):
```



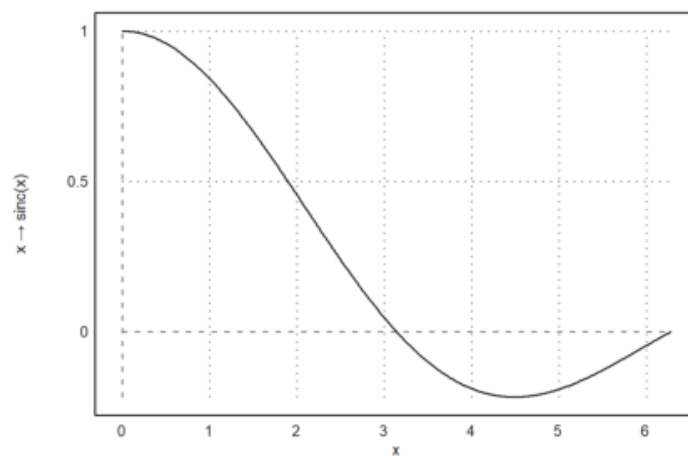
Label teks, judul, kotak label, dan teks lainnya dapat mengandung string Unicode (lihat sintaks EMT untuk informasi lebih lanjut tentang string Unicode).

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



Label pada sumbu x dan y dapat berorientasi vertikal, begitu juga dengan sumbunya.

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



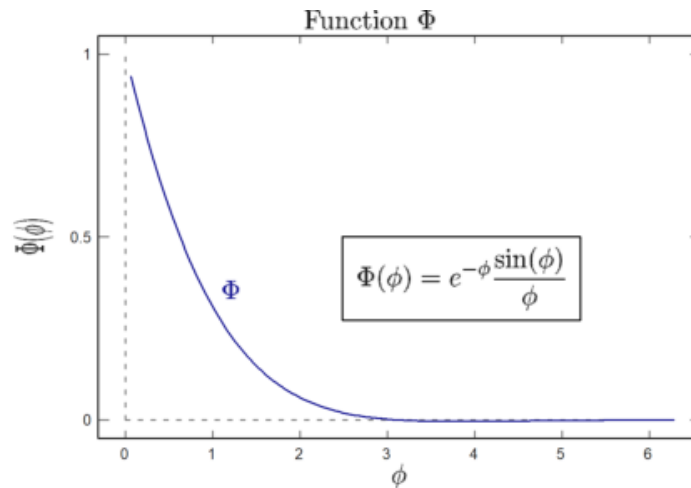
LaTeX

Anda juga dapat memplot formula LaTeX jika Anda telah instal sistem LaTeX. Saya merekomendasikan penggunaan MiKTeX. Jalur ke biner "latex" dan "dvi2png" harus ada dalam jalur sistem, atau Anda harus mengatur LaTeX dalam menu opsi.

Perlu diingat bahwa parsing LaTeX memakan waktu. Jika Anda ingin menggunakan LaTeX dalam plot beranimasi, Anda sebaiknya memanggil `latex()` sebelum perulangan sekali dan menggunakan hasilnya (gambar dalam matriks RGB).

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

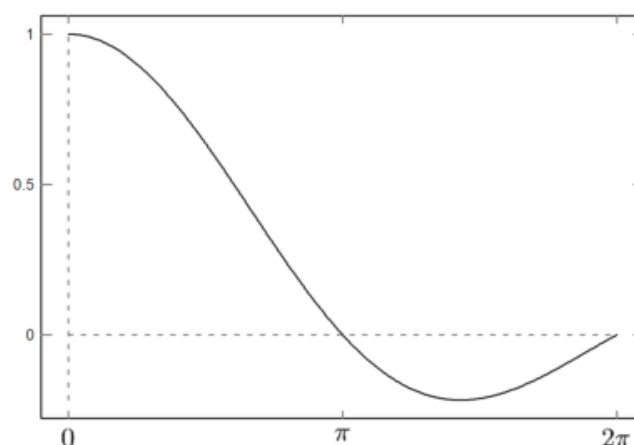
```
>plot2d("exp(-x)*sin(x)/x",a=0,b=2pi,c=0,d=1,grid=6,color=blue, ...
  title=latex("\text{Function \$\Phi\$}"), ...
  xl=latex("\phi"),yl=latex("\Phi(\phi)")); ...
  textbox( ...
    latex("\Phi(\phi) = e^{-\phi} \frac{\sin(\phi)}{\phi}"),x=0.8,y=0.5); ...
  label(latex("\Phi",color=blue),1,0.4):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



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

Cara paling mudah adalah melakukan plot kosong dengan bingkai menggunakan `grid=4`, dan kemudian menambahkan grid dengan `ygrid()` dan `xgrid()`. Pada 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):
```



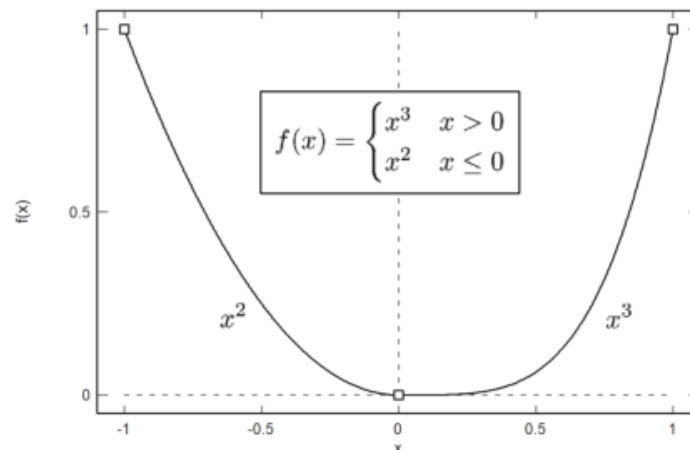
Tentu saja, fungsi-fungsi juga dapat digunakan untuk mengatur label-label pada sumbu x .

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

Parameter "map" membantu dalam menggunakan fungsi untuk vektor. Untuk plot, hal ini mungkin tidak diperlukan. Tetapi untuk menunjukkan bahwa vektorisasi berguna, kita tambahkan beberapa titik kunci ke plot pada $x=-1$, $x=0$, dan $x=1$.

Pada plot berikutnya, kami juga memasukkan beberapa kode LaTeX. Kami menggunakannya untuk dua label dan sebuah kotak teks. Tentu saja, Anda hanya akan dapat menggunakan LaTeX jika Anda telah menginstal LaTeX dengan benar.

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



Interaksi Pengguna

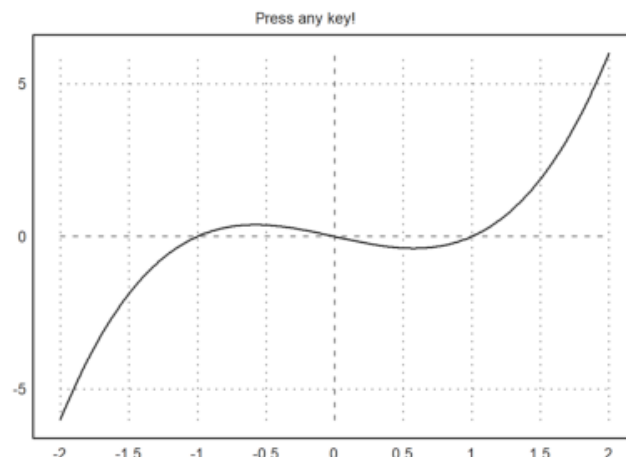
Saat memplot fungsi atau ekspresi, parameter `>user` memungkinkan pengguna untuk memperbesar dan menggeser plot dengan tombol kursor atau mouse. Pengguna dapat:

- memperbesar dengan + atau -
- menggeser plot dengan tombol kursor
- memilih jendela plot dengan mouse
- mereset tampilan dengan spasi
- keluar dengan tombol enter

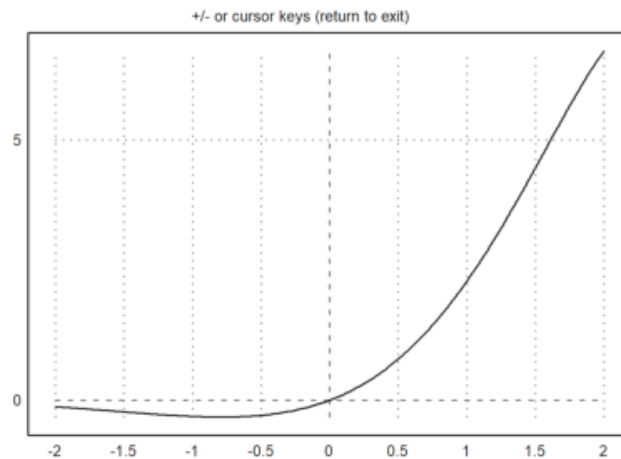
Tombol spasi akan mereset plot ke jendela plot asli.

Saat memplot data, flag `>user` hanya akan menunggu untuk penekanan tombol keyboard.

```
>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 adalah demonstrasi cara interaksi pengguna yang lebih canggih (lihat tutorial tentang pemrograman untuk detailnya).

Fungsi bawaan `mousedrag()` menunggu peristiwa mouse atau keyboard. Ini melaporkan mouse down, mouse moved, atau mouse up, serta penekanan tombol. Fungsi `dragpoints()` menggunakan ini, dan memungkinkan pengguna untuk menarik setiap titik dalam plot.

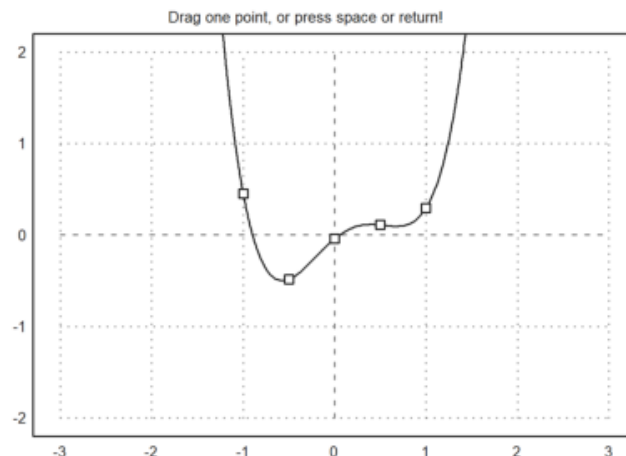
Kami perlu memiliki fungsi plot terlebih dahulu. Sebagai contoh, kami interpolasi dalam 5 titik dengan polinomial. Fungsi ini seharusnya memplot ke dalam area plot tetap.

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

Sekarang kita akan menghasilkan beberapa nilai acak dan biarkan pengguna menarik 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 untuk menyesuaikan parameter-parameter tersebut.

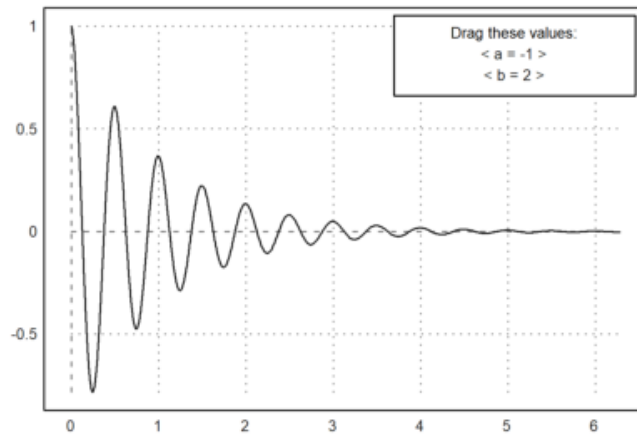
Pertama, kita perlu fungsi `plot`.

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

Kemudian kita memerlukan nama-nama parameter, nilai awal, dan matriks nx2 dari rentang, opsionalnya dengan baris judul.

Ada slider interaktif yang dapat mengatur nilai oleh pengguna. Fungsi dragvalues() menyediakannya.

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

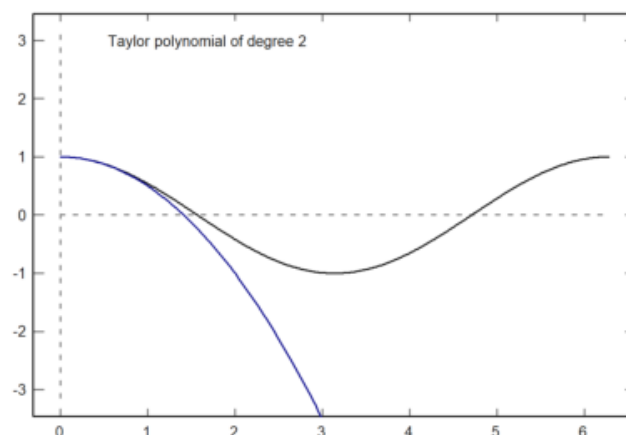


Mungkin juga untuk membatasi nilai-nilai yang ditarik menjadi bilangan bulat. Sebagai contoh, kami menulis fungsi plot yang memplot polinomial Taylor derajat n untuk fungsi kosinus.

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

Sekarang kami memungkinkan derajat n bervariasi dari 0 hingga 20 dalam 20 langkah. Hasil dari dragvalues() digunakan untuk memplot sketsa dengan n ini, dan untuk menyisipkan plot ke dalam notebook.

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



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

```
>function dragtest ...
    plot2d(none,r=1,title="Drag with the mouse, or press any key!");
    start=0;
```

```

repeat
    {flag,m,time}=mousedrag();
    if flag==0 then return; endif;
    if flag==2 then
        hold on; mark(m[1],m[2]); hold off;
    endif;
end
endfunction

```

>dragtest // lihat hasilnya dan cobalah lakukan!

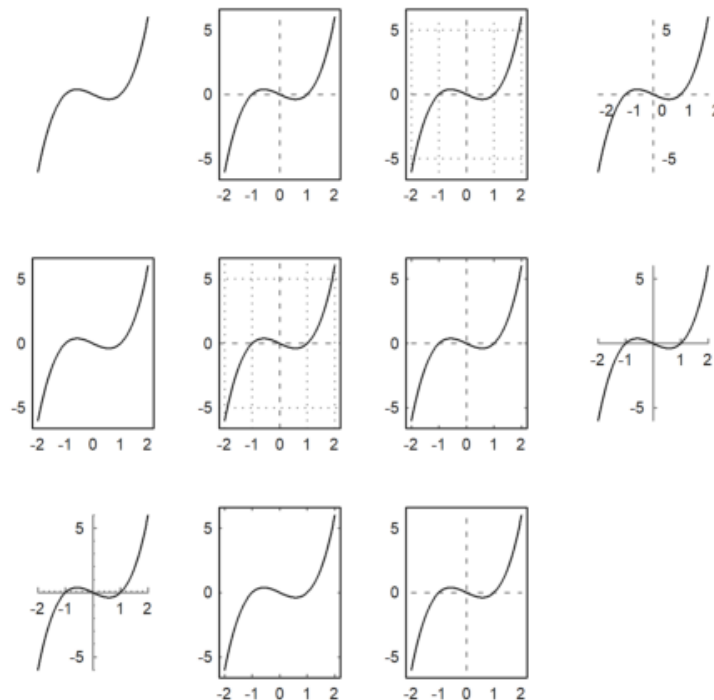
Gaya Plot 2D

Secara default, EMT menghitung tanda sumbu secara otomatis dan menambahkan label pada setiap tanda tersebut. Ini dapat diubah dengan parameter grid. Gaya default sumbu dan labelnya 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):

```



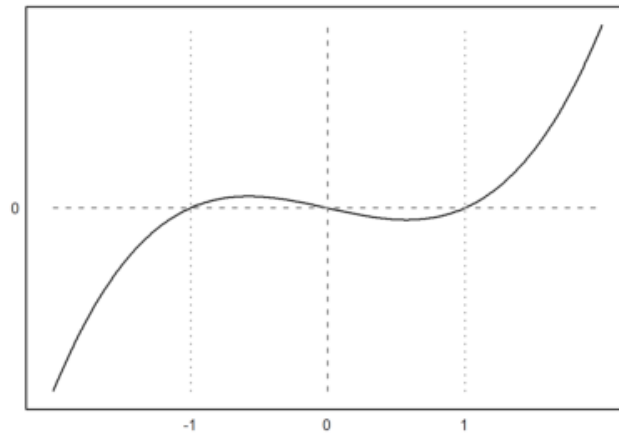
Parameter <frame mematikan bingkai, dan framecolor=blue mengatur bingkai dengan warna biru.

Jika Anda ingin menggunakan tanda-tanda kustom, Anda dapat menggunakan style=0, dan menambahkannya nanti.

```

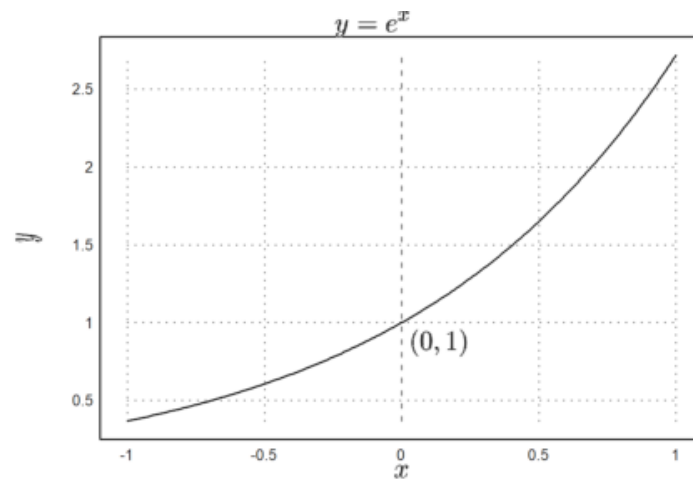
>aspect(1.5);
>plot2d("x^3-x",grid=0); // plot
>frame; xgrid([-1,0,1]); ygrid(0); // add frame and grid

```



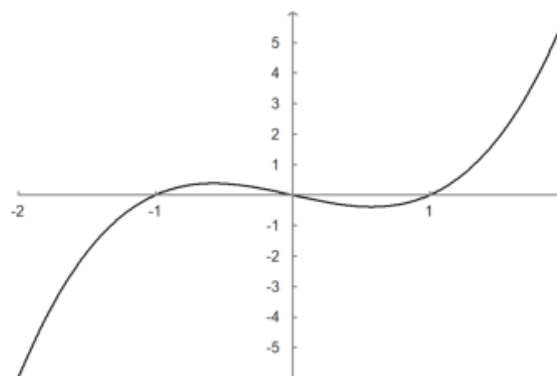
Untuk judul plot dan label sumbu, lihat contoh berikut.

```
>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-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="->");
```

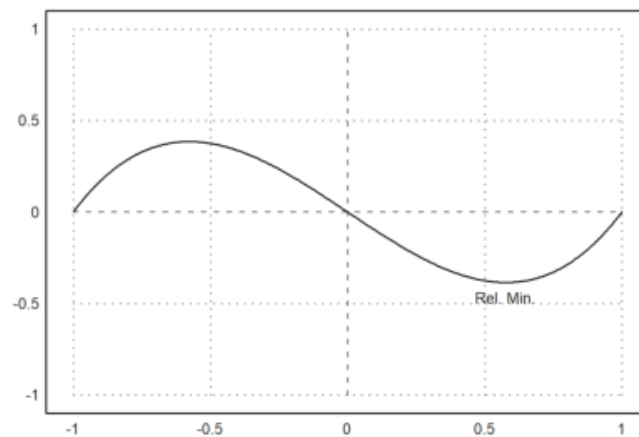


Teks pada plot dapat diatur dengan `label()`. Pada contoh berikut, "lc" berarti lower center. Ini mengatur posisi label relatif terhadap koordinat plot.


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

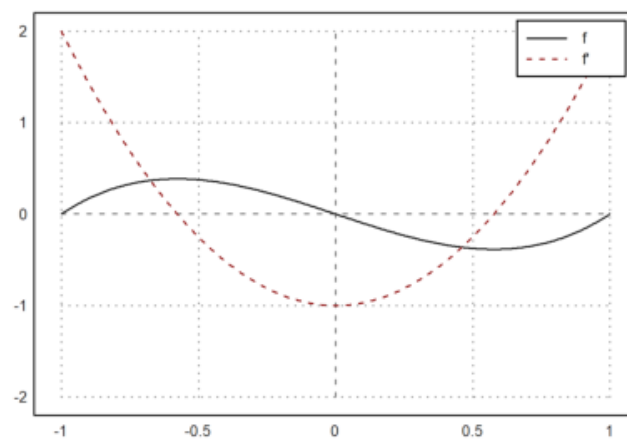
$$x^3 - x$$

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

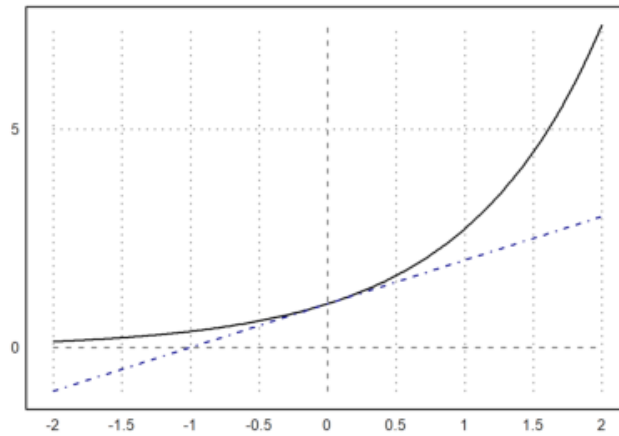


Ada juga kotak teks.

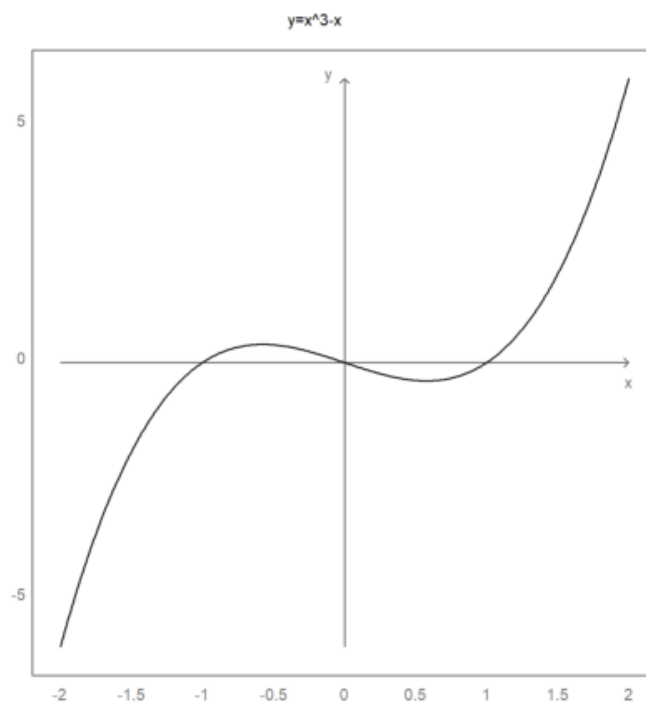
```
>plot2d(&f(x),-1,1,-2,2); // function
>plot2d(&diff(f(x),x),>add,style="--",color=red); // derivative
>labelbox(["f","f'"],["-","--"],[black,red]): // label box
```



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



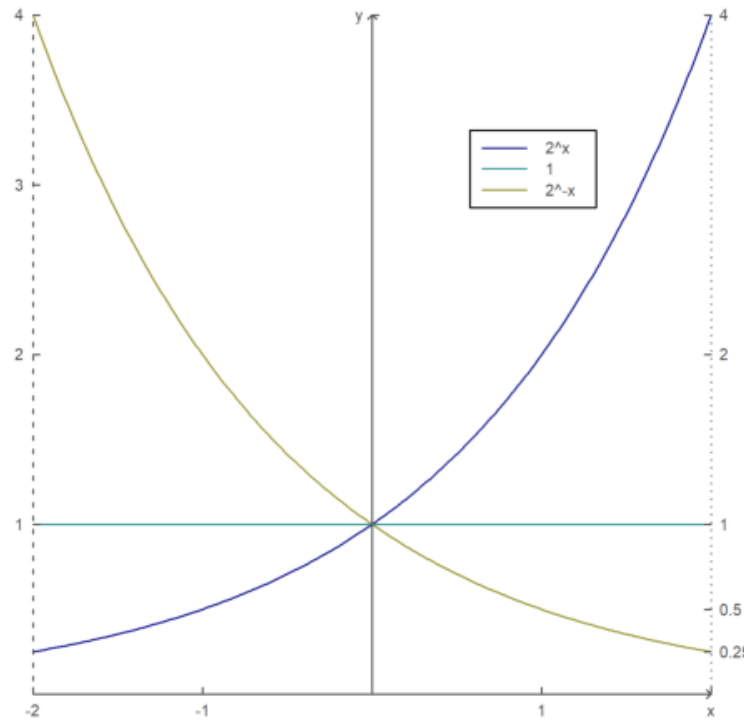
```
>gridstyle("->",color=gray,textcolor=gray,framecolor=gray); ...
plot2d("x^3-x",grid=1); ...
settitle("y=x^3-x",color=black); ...
label("x",2,0,pos="bc",color=gray); ...
label("y",0,6,pos="cl",color=gray); ...
reset():
```



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

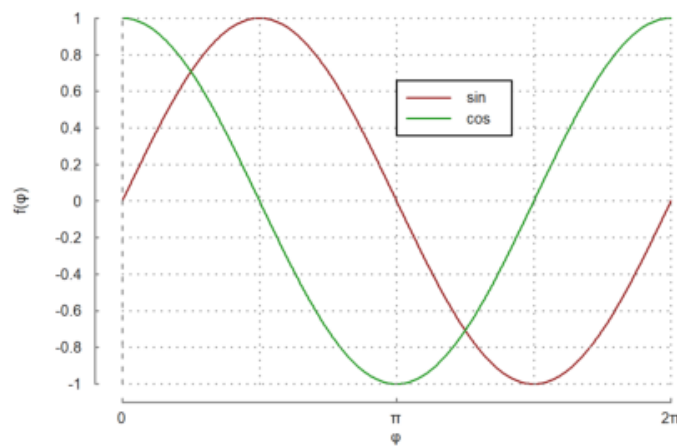
Perintah `fullwindow()` memperluas jendela plot karena kita tidak lagi memerlukan tempat untuk label di luar jendela plot. Gunakan `shrinkwindow()` atau `reset()` untuk mengatur ulang ke pengaturan default.

```
>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-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;)"):
```



Plotting Data 2D

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

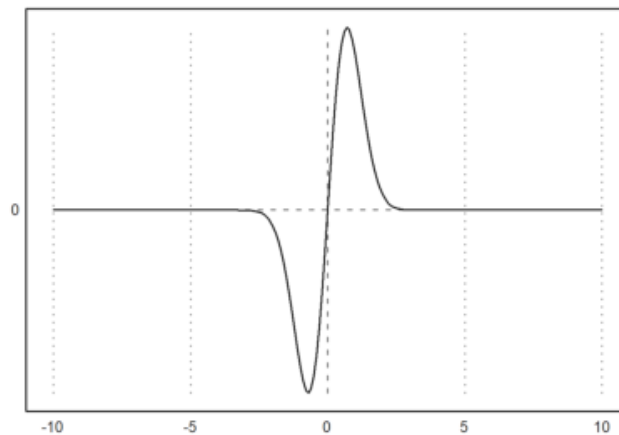
Memplot ekspresi hanyalah singkatan untuk plot data. Untuk plot data, Anda memerlukan satu atau lebih baris nilai-nilai x , dan satu atau lebih baris nilai-nilai y . Dari rentang dan nilai-nilai x , fungsi `plot2d` akan menghitung data yang akan diplot, secara default dengan evaluasi adaptif fungsi tersebut.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

Tetapi Anda juga dapat memasukkan data secara langsung.

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

Berikut contoh dengan satu baris untuk x dan y.

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



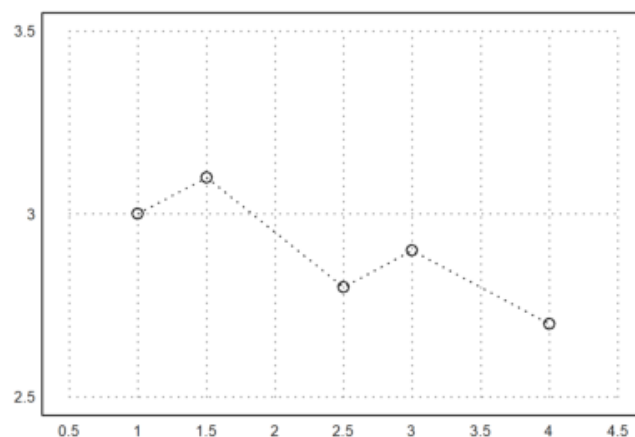
Data juga dapat diplot sebagai titik-titik. Gunakan `points=true` untuk ini. Plot ini bekerja seperti poligon, tetapi hanya menggambar sudut-sudutnya.

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

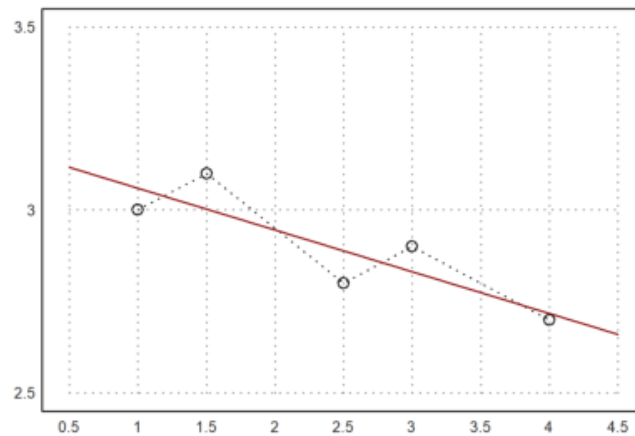
Untuk memplot set titik-titik, gunakan `>points`. Jika warnanya adalah vektor warna, setiap titik akan memiliki warna yang berbeda. Untuk matriks koordinat dan vektor kolom, warna berlaku untuk baris-baris matriks tersebut.

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

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



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



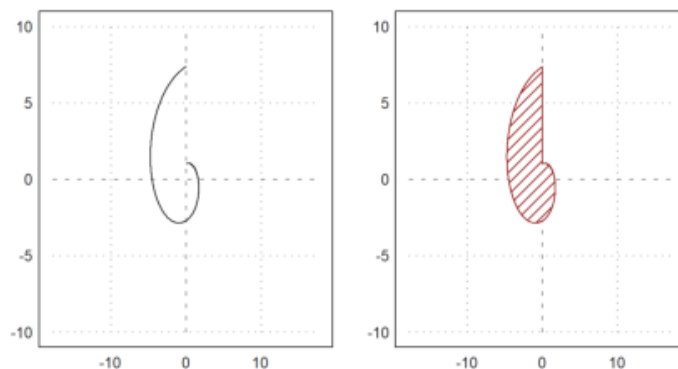
Menggambar Daerah Yang Dibatasi Kurva

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

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

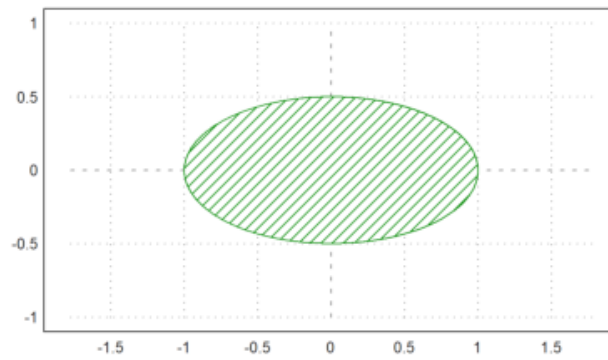
Warna isian ditentukan oleh argumen "`fillcolor`", dan `<outline opsional` menghindari menggambar batas untuk semua gaya kecuali gaya default.

```
>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):
```

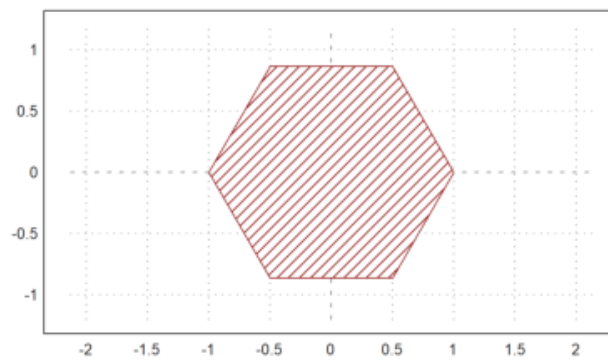


Pada contoh berikut, kami memplot elips terisi dan dua heksagon terisi menggunakan kurva tertutup dengan 6 titik dengan gaya pengisian yang berbeda.

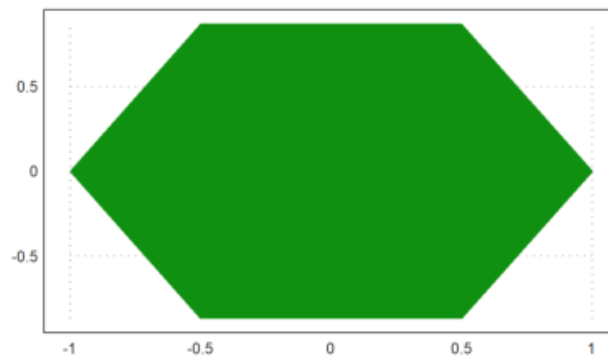
```
>x=linspace(0,2pi,1000); plot2d(sin(x),cos(x)*0.5,r=1,>filled,style="/"):
```



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

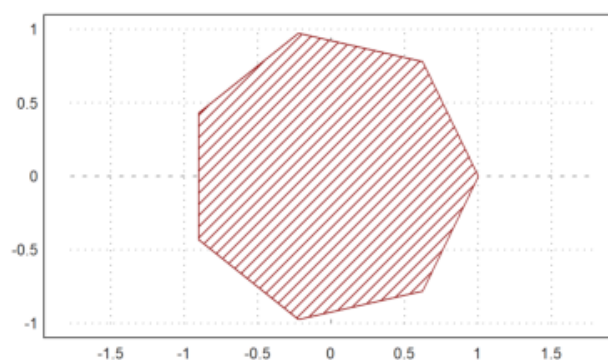


```
>t=linspace(0,2pi,6); plot2d(cos(t),sin(t),>filled,style="#"): 
```



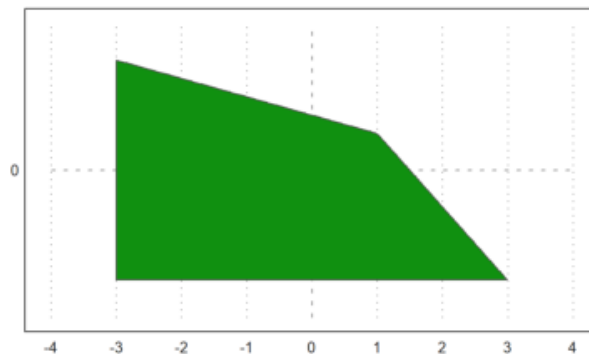
Contoh lainnya adalah sebuah heptagon, yang kita buat dengan 7 titik pada lingkaran satuan.

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



Berikut adalah himpunan nilai maksimal dari empat kondisi linear 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):
```



Titik utama bahasa matriks adalah bahwa itu memungkinkan untuk menghasilkan tabel fungsi dengan mudah.

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

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

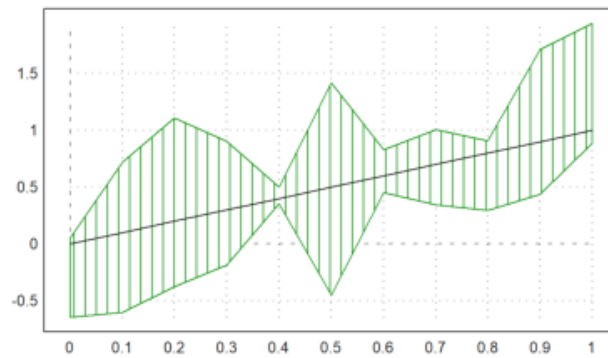
```
>plot2d(x,y,<grid,<frame,>filled):
```



Sebuah vektor interval diplotkan terhadap nilai-nilai x sebagai daerah terisi antara nilai-nilai bawah dan atas dari interval tersebut.

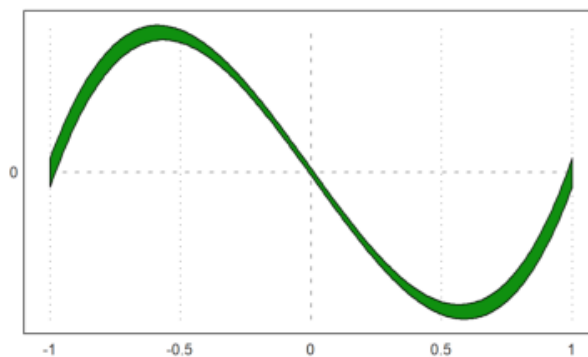
Ini bisa berguna untuk memplot kesalahan perhitungan. Tetapi ini juga dapat digunakan untuk memplot kesalahan statistik.

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



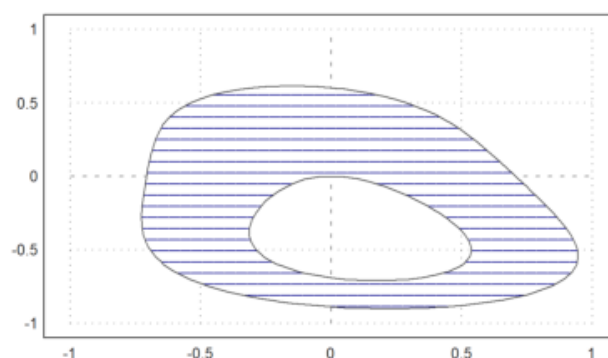
Jika x adalah vektor yang telah diurutkan, dan y adalah vektor interval, maka `plot2d` akan memplot rentang terisi dari interval tersebut di bidang. Gaya pengisian adalah sama dengan gaya poligon.

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



Mungkin untuk mengisi wilayah nilai-nilai untuk fungsi tertentu. Untuk ini, `level` harus menjadi matriks $2 \times n$. Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

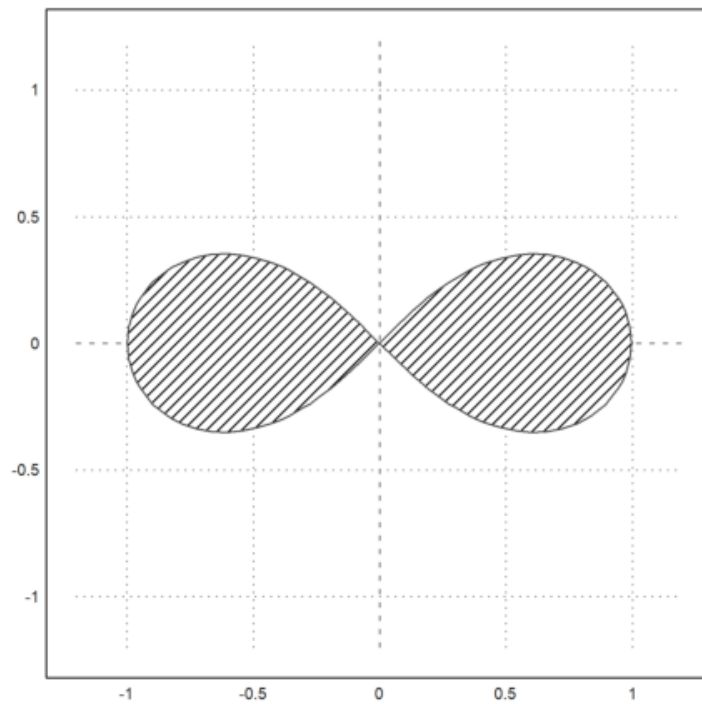
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=[0;1],style="-",color=blue): // 0 <= f(x,y) <= 1
```



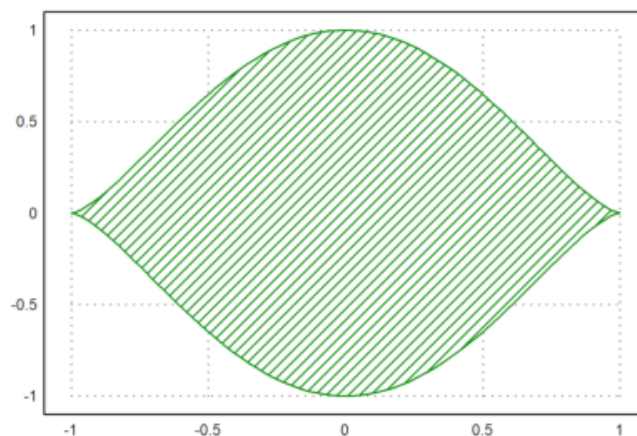
Kami juga dapat mengisi rentang nilai seperti

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

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

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



Grafik Fungsi Parametrik

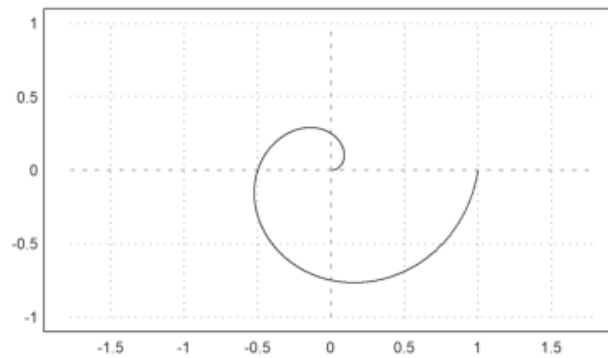
Nilai-nilai x tidak perlu diurutkan. (x, y) hanya menggambarkan sebuah kurva. Jika x diurutkan, kurva tersebut adalah grafik dari suatu fungsi.

Pada contoh berikut, kita memplot spiral

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

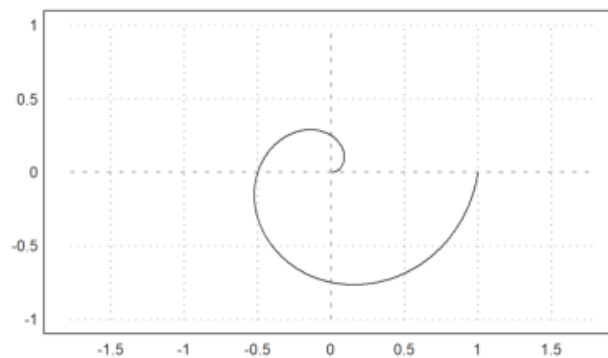
Kita dapat menggunakan banyak titik untuk tampilan yang halus atau menggunakan fungsi `adaptive()` untuk mengevaluasi ekspresi (lihat fungsi `adaptive()` untuk lebih banyak detail).

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

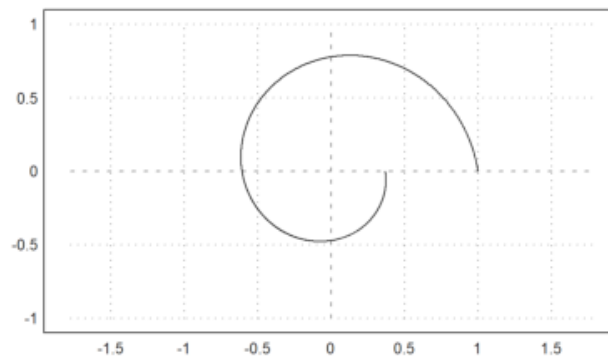


Sebagai alternatif, kita dapat menggunakan dua ekspresi untuk kurva. Berikut memplot kurva yang sama seperti sebelumnya.

```
>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):
```



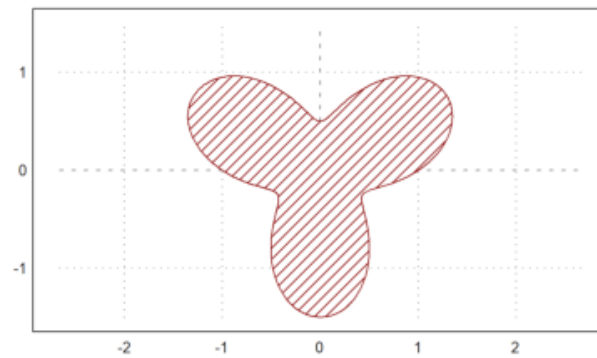
Di contoh berikutnya, kita memplot kurva

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

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):
```



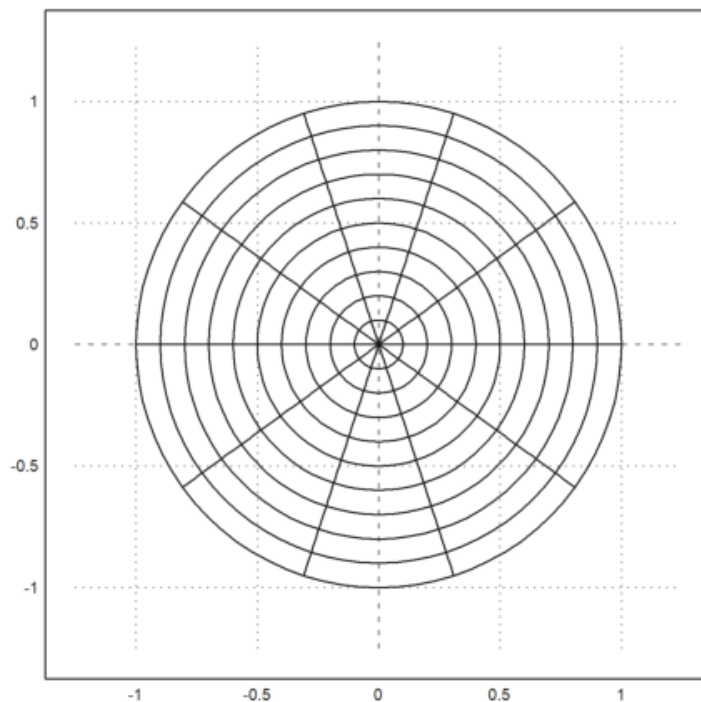
Menggambar Grafik Bilangan Kompleks

Sebuah array bilangan kompleks juga dapat diplotkan. Kemudian titik-titik grid akan dihubungkan. Jika sejumlah garis grid ditentukan (atau vektor 1x2 dari garis grid) dalam argumen `cgrid`, hanya garis grid tersebut yang terlihat.

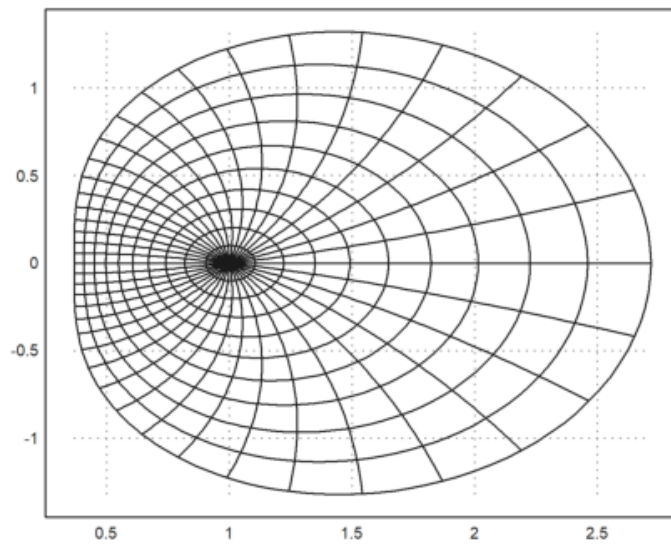
Matriks bilangan kompleks akan otomatis diplot sebagai grid dalam bidang kompleks.

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

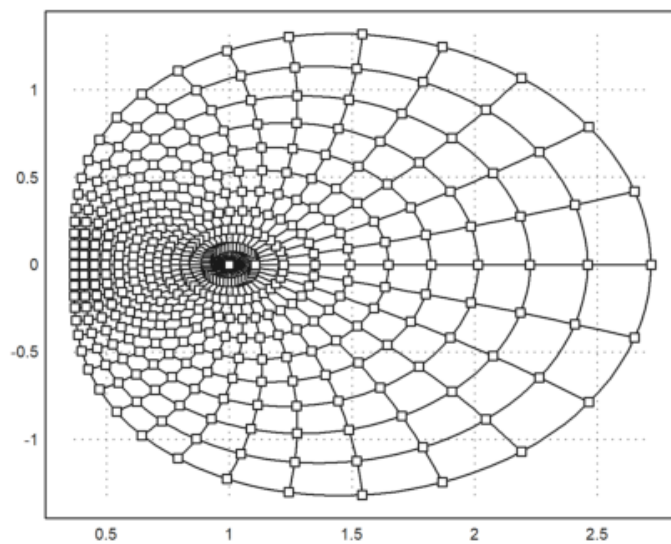
```
>aspect(); r=linspace(0,1,50); a=linspace(0,2pi,80)'; z=r*exp(I*a);...
plot2d(z,a=-1.25,b=1.25,c=-1.25,d=1.25,cgrid=10):
```



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



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

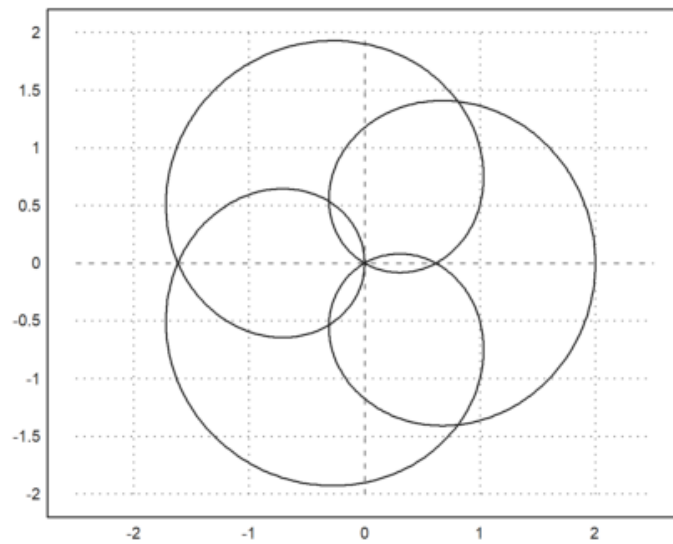


Sebuah vektor bilangan kompleks otomatis diplot sebagai kurva dalam bidang kompleks dengan bagian real dan bagian imajiner.

Pada contoh berikut, 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

Terdapat banyak fungsi yang mengkhususkan diri dalam plot statistik. Salah satu plot yang sering digunakan adalah plot kolom.

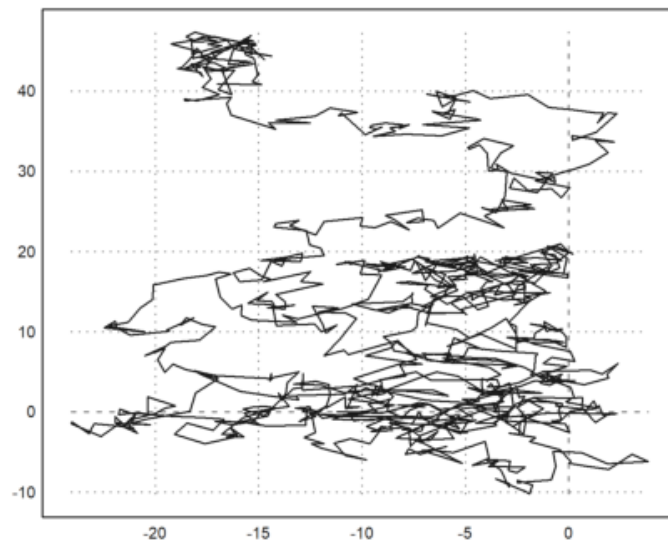
Jumlah kumulatif dari nilai-nilai yang terdistribusi normal 0-1 menghasilkan perjalanan acak.

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

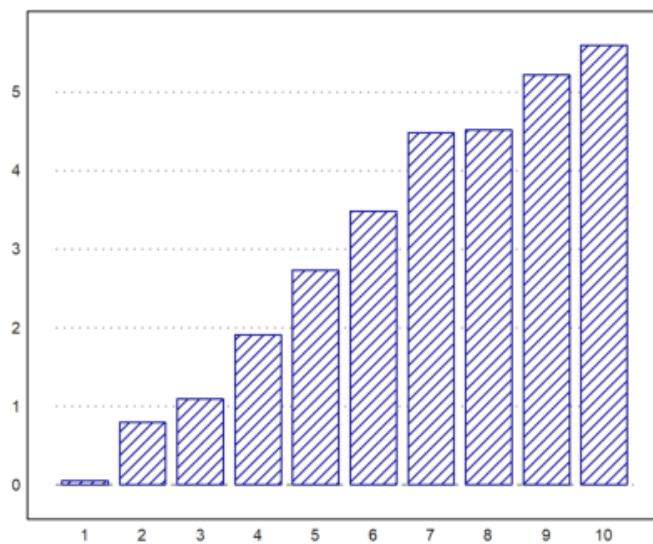


Menggunakan dua baris menampilkan perjalanan dalam dua dimensi.

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

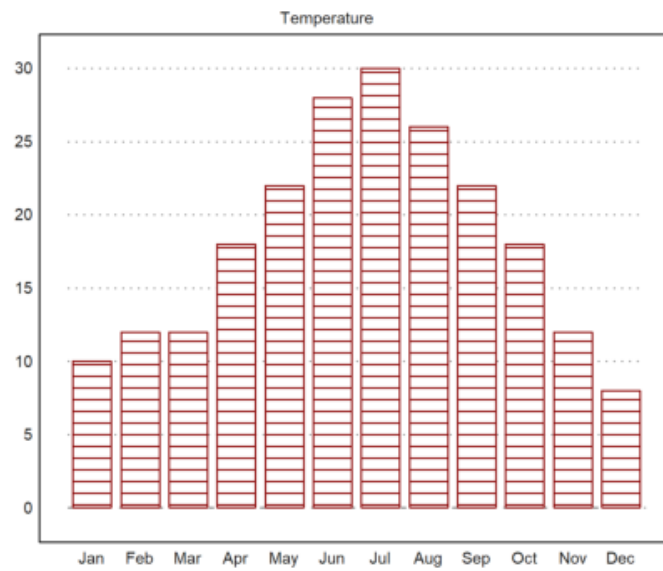


```
>columnsplot(cumsum(random(10)),style="/",color=blue):
```

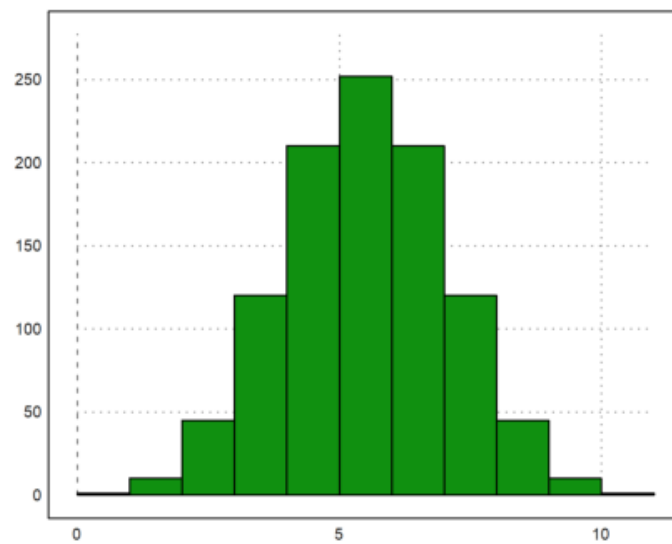


Ini juga dapat menampilkan string sebagai label.

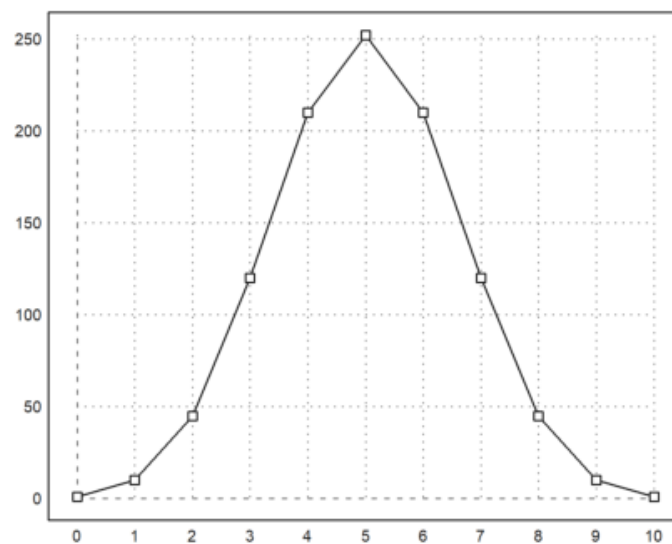
```
>months=["Jan","Feb","Mar","Apr","May","Jun", ...
         "Jul","Aug","Sep","Oct","Nov","Dec"];
>values=[10,12,12,18,22,28,30,26,22,18,12,8];
>columnsplot(values,lab=months,color=red,style="-");
>title("Temperature"):
```



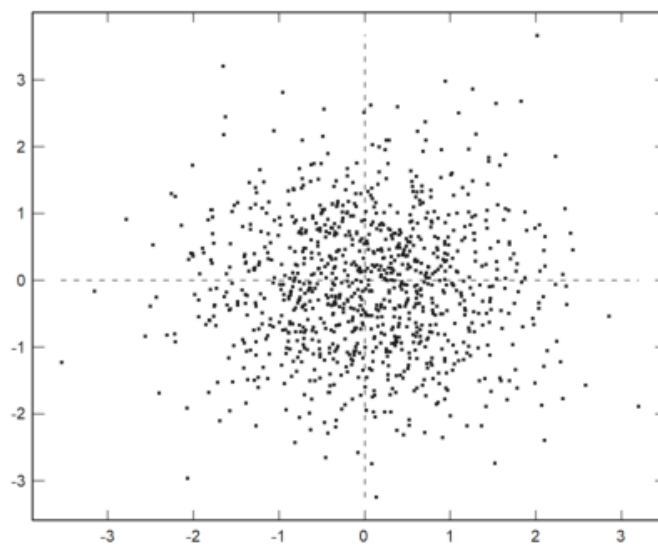
```
>k=0:10;
>plot2d(k,bin(10,k),>bar):
```



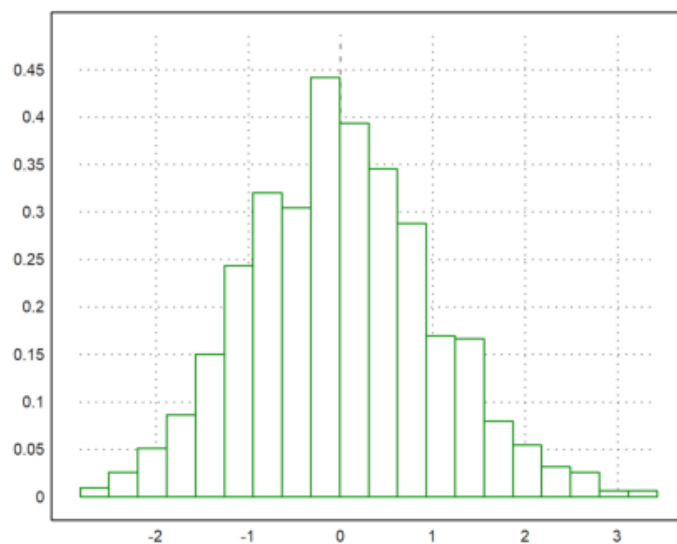
```
>plot2d(k,bin(10,k)); plot2d(k,bin(10,k),>points,>add):
```



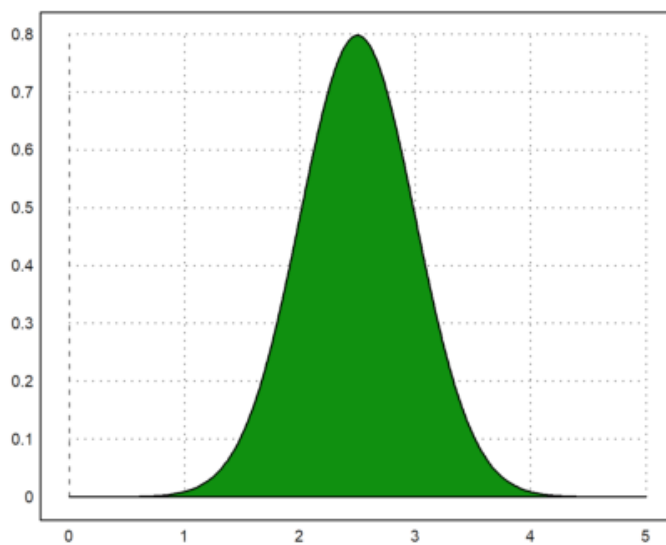
```
>plot2d(normal(1000),normal(1000),>points,grid=6,style=".."):
```



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

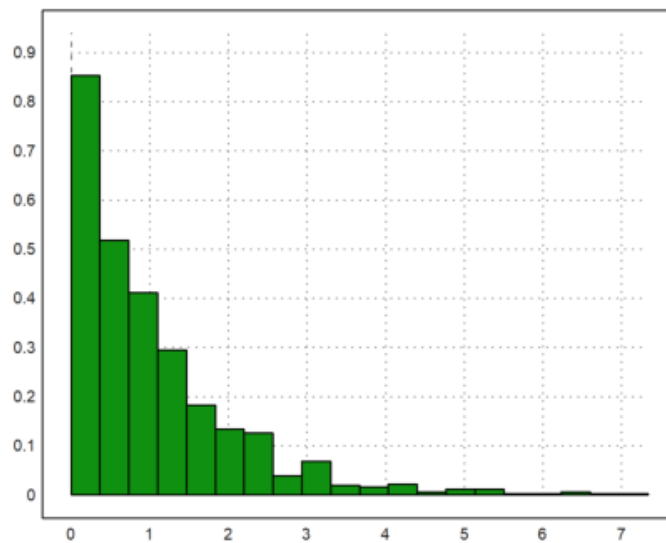


```
>plot2d("qnormal",0,5;2.5,0.5,>filled):
```



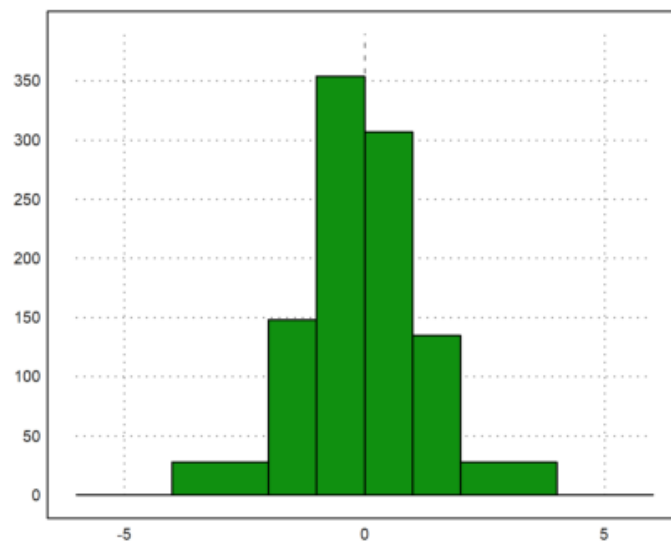
Untuk memplot distribusi statistik eksperimental, Anda dapat menggunakan `distribution=n` dengan `plot2d`.

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



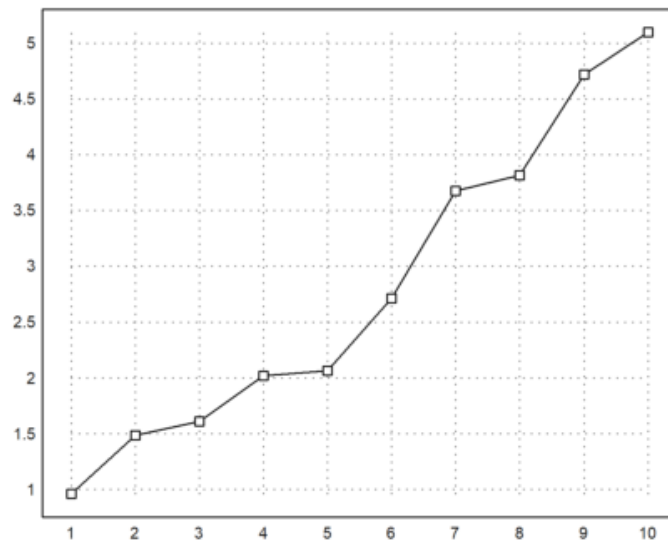
Atau Anda dapat menghitung distribusi dari data dan memplot hasilnya dengan `>bar` di `plot3d`, atau dengan `plot kolom`.

```
>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):
```

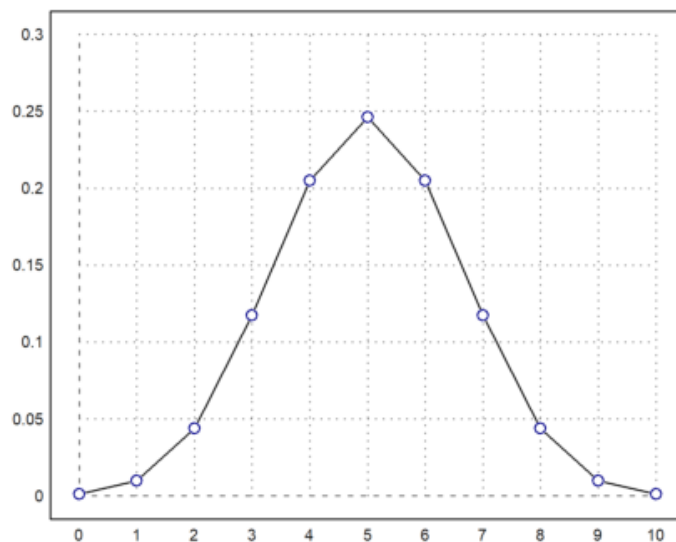


Fungsi `statplot()` mengatur gaya dengan string sederhana.

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



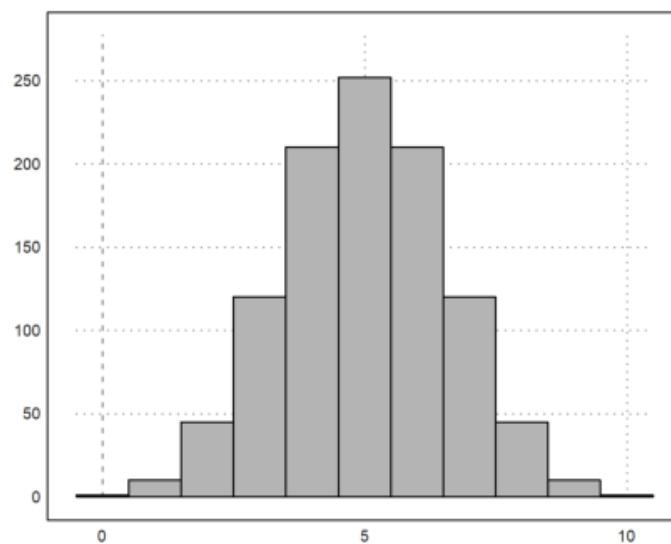
```
>n=10; i=0:n; ...
plot2d(i,bin(n,i)/2^n,a=0,b=10,c=0,d=0.3); ...
plot2d(i,bin(n,i)/2^n,points=true,style="ow",add=true,color=blue):
```



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

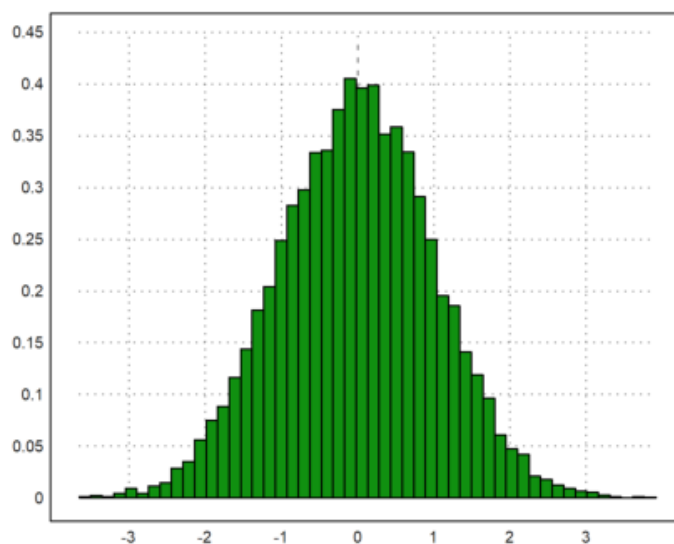
Gaya pengisian dapat digunakan seperti di atas.

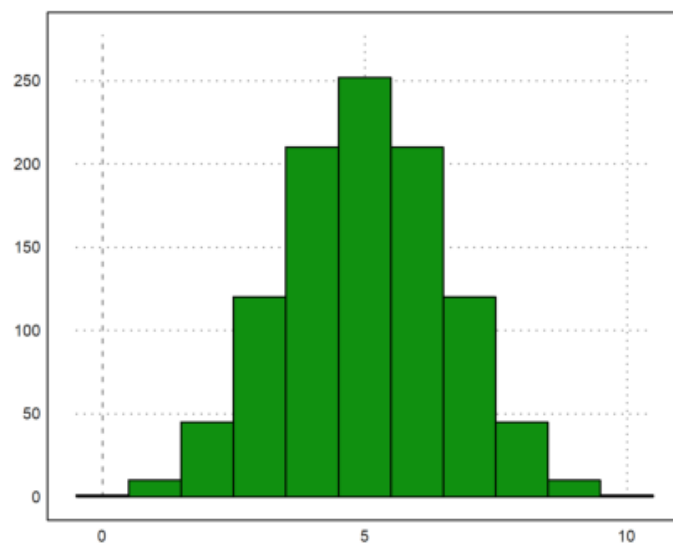
```
>n=10; k=bin(n,0:n); ...
plot2d(-0.5:n+0.5,k,bar=true,fillcolor=lightgray):
```



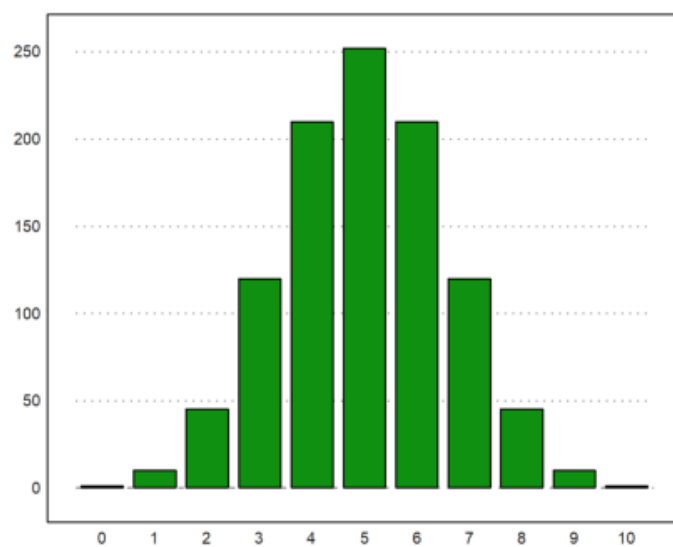
Data untuk plot batang (`bar=1`) dan histogram (`histogram=1`) dapat diberikan secara eksplisit dalam `xv` dan `yv`, atau dapat dihitung dari distribusi empiris dalam `xv` dengan `>distribution` (atau `distribution=n`). Histogram dari nilai-nilai `xv` akan dihitung secara otomatis dengan `>histogram`.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

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

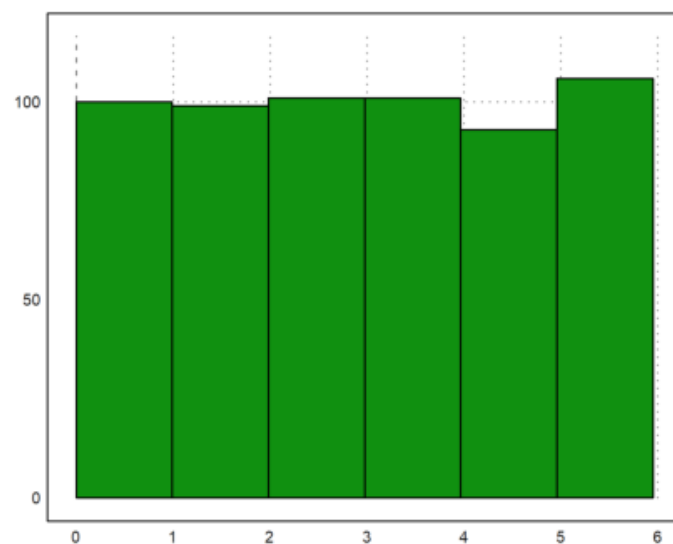




```
>columnplot(m,k):
```

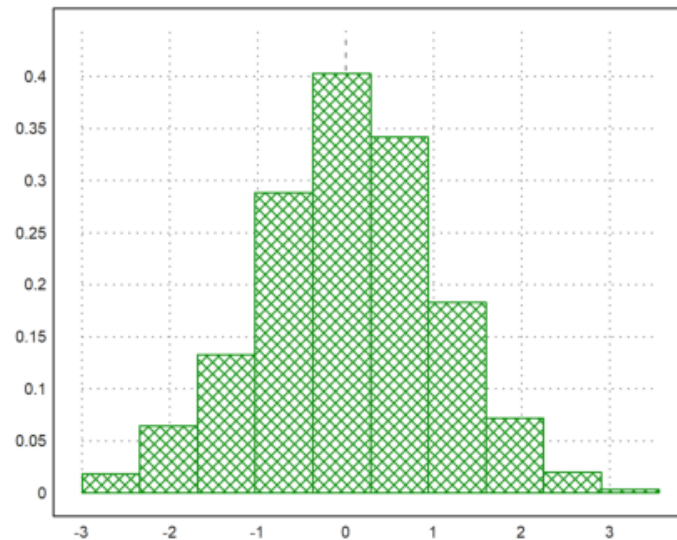


```
>plot2d(random(600)*6,histogram=6):
```



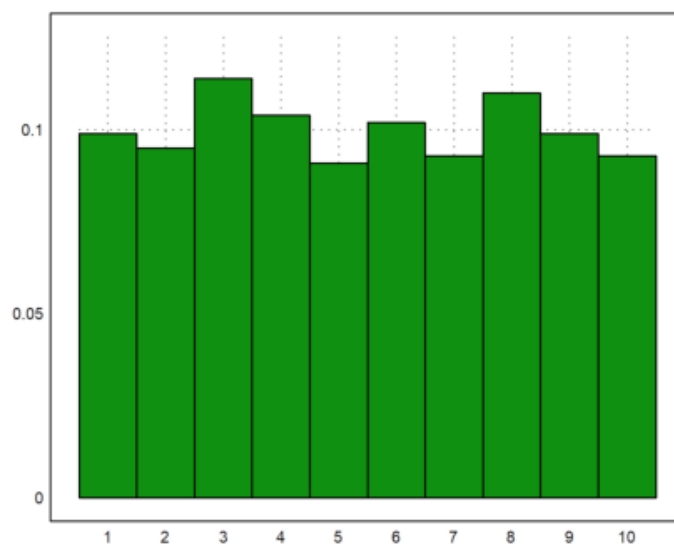
Untuk distribusi, ada parameter `distribution=n`, yang menghitung nilai secara otomatis dan mencetak distribusi relatif dengan `n` sub-interval.

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



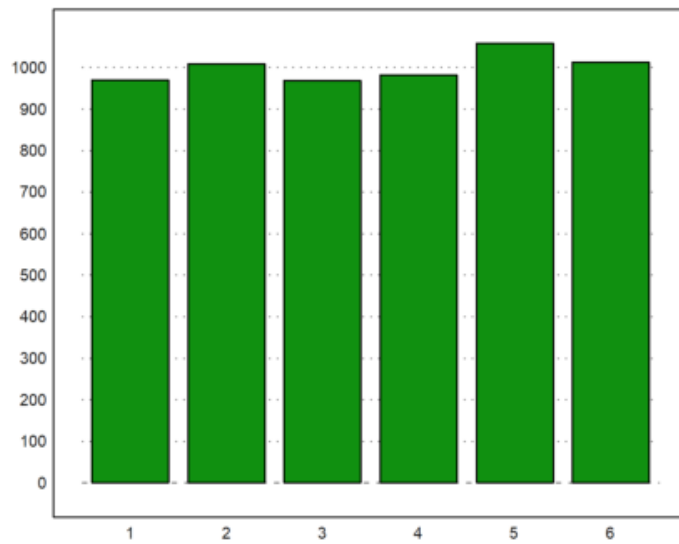
Dengan parameter `even=true`, ini akan menggunakan interval bilangan bulat.

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

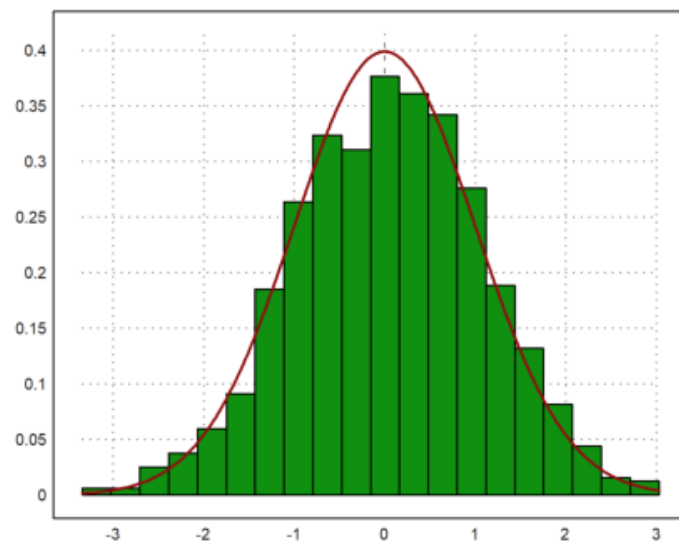


Perlu diingat bahwa ada banyak plot statistik yang mungkin berguna. Lihatlah tutorial tentang statistik untuk informasi lebih lanjut.

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

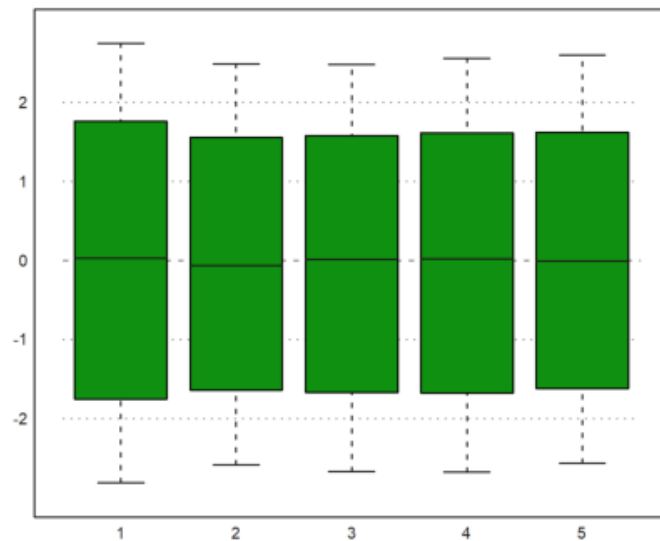


```
>plot2d(normal(1,1000),>distribution); ...
  plot2d("qnormal(x)",color=red,thickness=2,>add):
```



Ada juga banyak plot khusus untuk statistik. Boxplot menampilkan kuartil dari distribusi ini dan banyak outlier. Menurut definisi, outlier dalam boxplot adalah data yang melebihi 1.5 kali jangkauan tengah 50% dari plot.

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



Fungsi-fungsi Implisit

Plot implisit menampilkan garis tingkat yang memecahkan $f(x,y)=level$, di mana "level" dapat menjadi nilai tunggal atau vektor nilai. Jika `level="auto"`, akan ada `nc` garis tingkat, yang akan menyebar antara minimum dan maksimum fungsi dengan merata. Warna yang lebih gelap atau lebih terang dapat ditambahkan dengan `>hue` untuk menunjukkan nilai fungsi. Untuk fungsi implisit, `xv` harus menjadi fungsi atau ekspresi dari parameter `x` dan `y`, atau, sebagai alternatif, `xv` dapat menjadi matriks nilai-nilai.

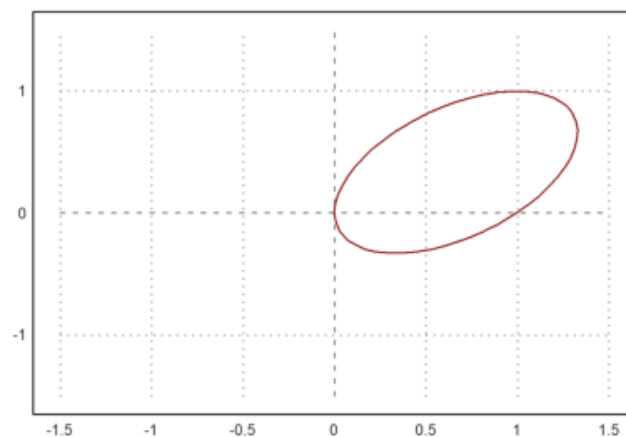
Euler dapat menandai garis tingkat

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

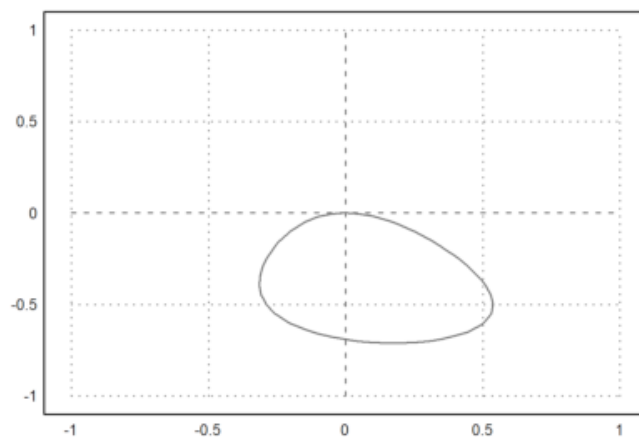
dari setiap fungsi.

Untuk menggambar set $f(x,y)=c$ 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 menjadi vektor garis tingkat. Selain itu, skema warna dapat digambar di latar belakang untuk menunjukkan nilai fungsi untuk setiap titik dalam plot. Parameter "`n`" menentukan keluwesan plot.

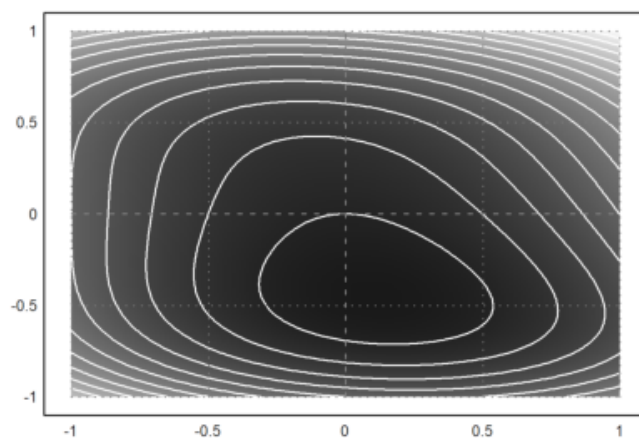
```
>aspect(1.5);
>plot2d("x^2+y^2-x*y-x",r=1.5,level=0,contourcolor=red):
```



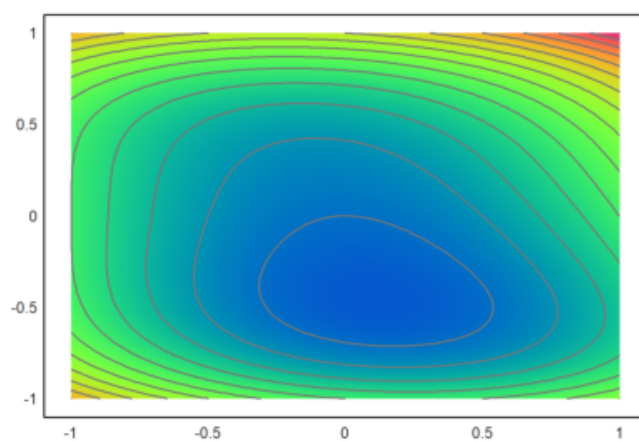
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=0): // Solutions of f(x,y)=0
```



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

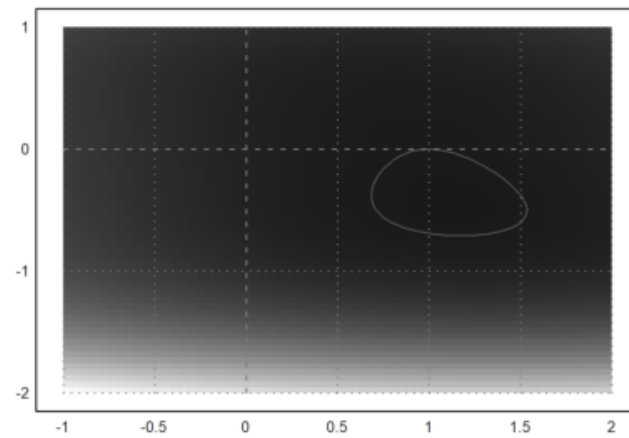


```
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=200,grid=4): // nicer
```

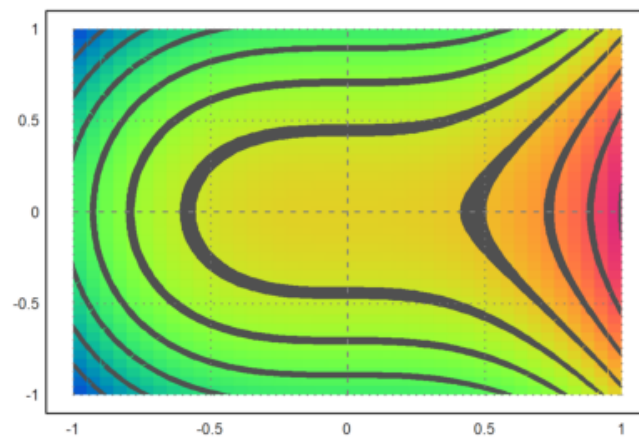


Ini juga berfungsi untuk plot data. Tetapi Anda harus menentukan rentang untuk label sumbu.

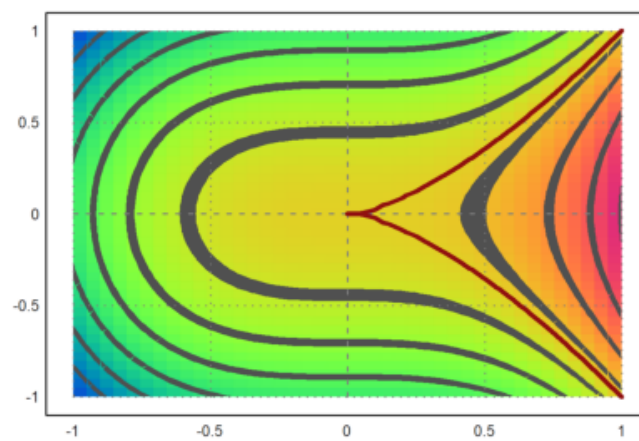
```
>x=-2:0.05:1; y=x'; z=expr(x,y);
>plot2d(z,level=0,a=-1,b=2,c=-2,d=1,>hue):
```

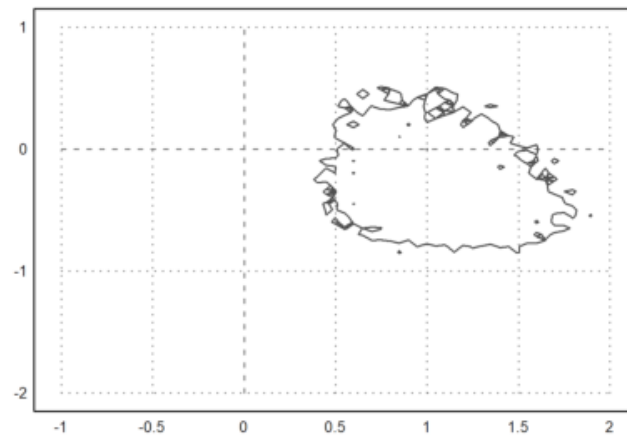
```
>plot2d("x^3-y^2",>contour,>hue,>spectral):
```



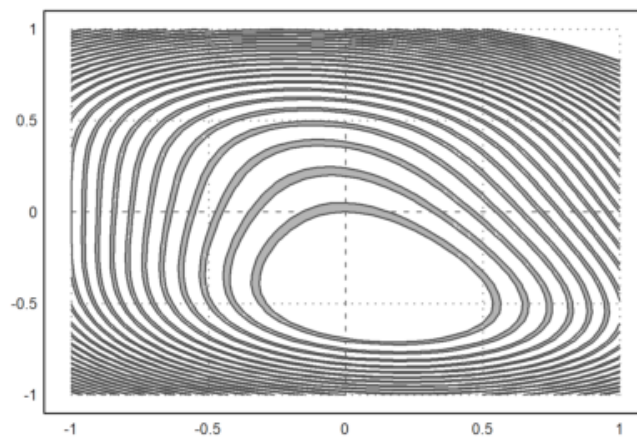
```
>plot2d("x^3-y^2",level=0,contourwidth=3,>add,contourcolor=red):
```



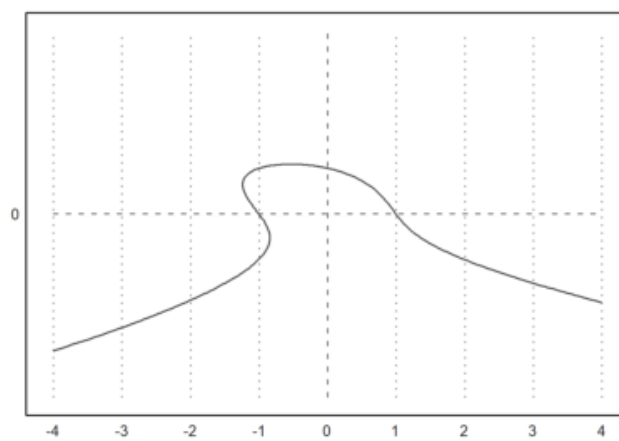
```
>z=z+normal(size(z))*0.2;
>plot2d(z,level=0.5,a=-1,b=2,c=-2,d=1):
```



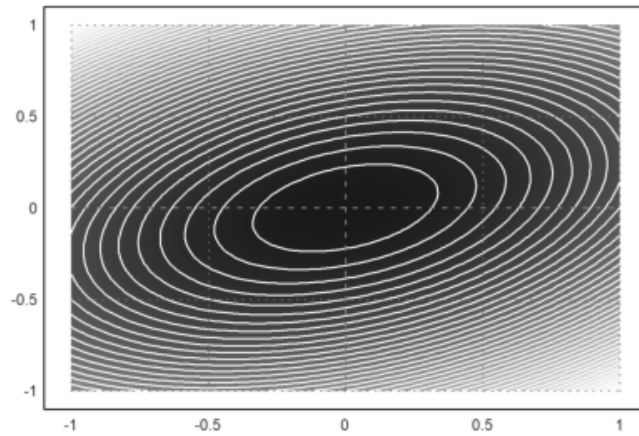
```
>plot2d(expr,level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
```



```
>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):
```



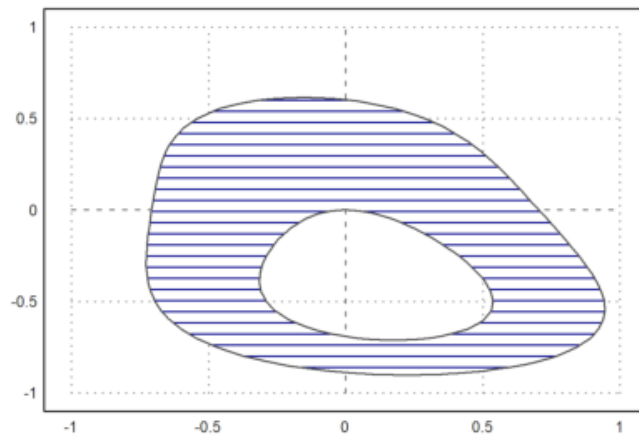
Anda juga dapat mengisi set

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

dengan rentang level.

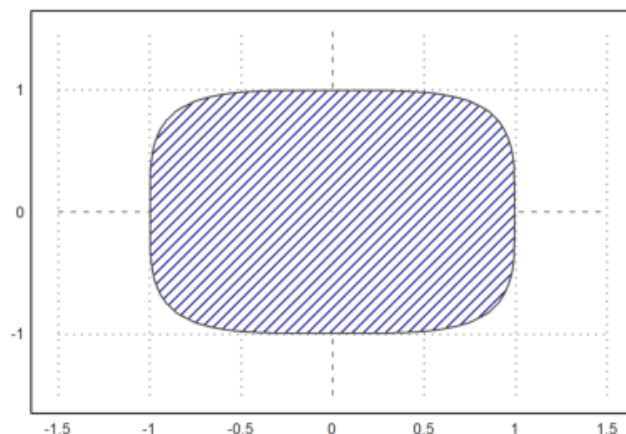
Anda dapat mengisi wilayah nilai-nilai untuk fungsi tertentu. Untuk ini, level harus berupa matriks 2xn. Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

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

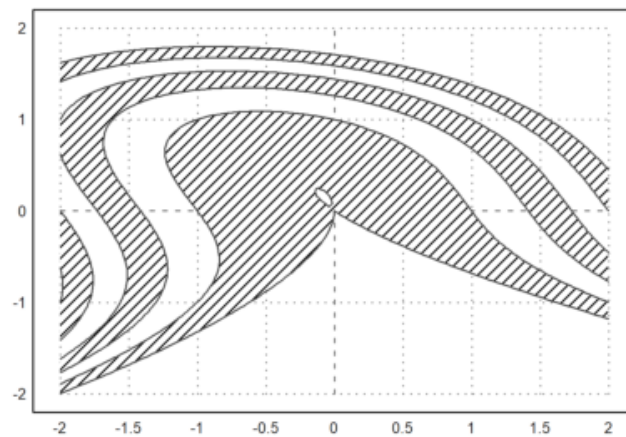


Plot implisit juga dapat menampilkan rentang level. Maka itu, level harus berupa matriks 2xn dari interval level, di mana baris pertama berisi awal dan baris kedua berisi akhir dari setiap interval. Sebagai alternatif, vektor baris sederhana dapat digunakan untuk level, dan parameter dl memperluas nilai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

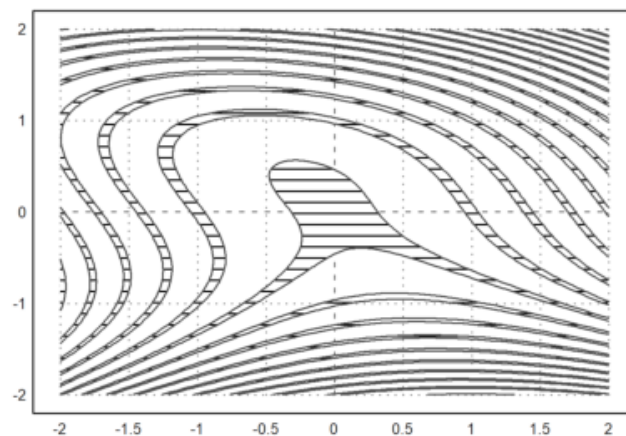
```
>plot2d("x^4+y^4",r=1.5,level=[0;1],color=blue,style="/"):
```



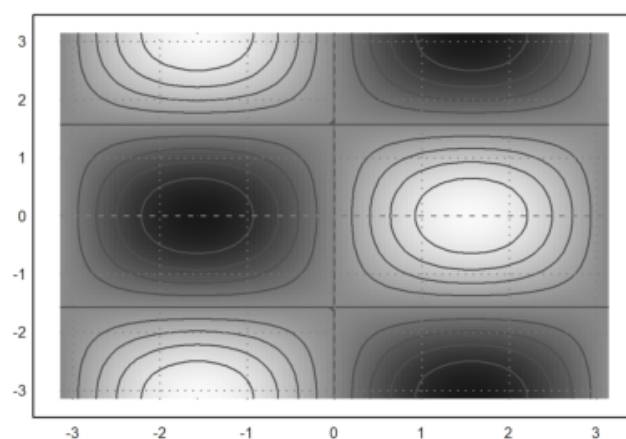
```
>plot2d("x^2+y^3+x*y",level=[0,2,4;1,3,5],style="/",r=2,n=100):
```



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



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

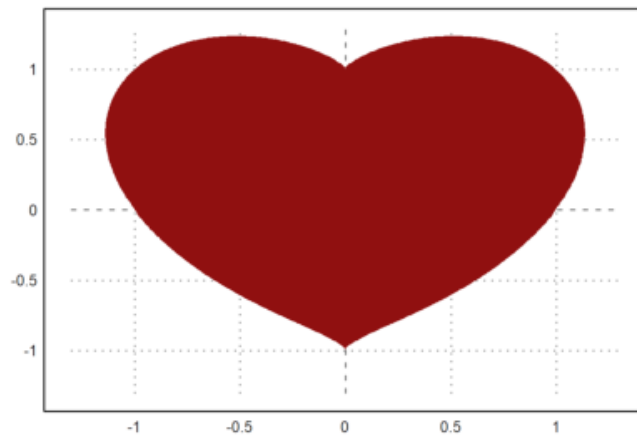


Juga mungkin untuk menandai sebuah wilayah

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

Ini dilakukan dengan menambahkan sebuah level dengan dua baris.

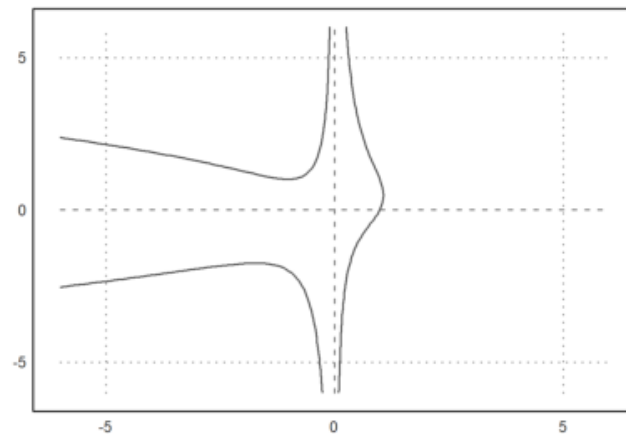
```
>plot2d("(x^2+y^2-1)^3-x^2*y^3",r=1.3, ...  
style="#",color=red,<outline, ...  
level=[-2;0],n=100):
```



Mungkin untuk menentukan level tertentu. Misalnya, kita dapat menggambar 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):
```

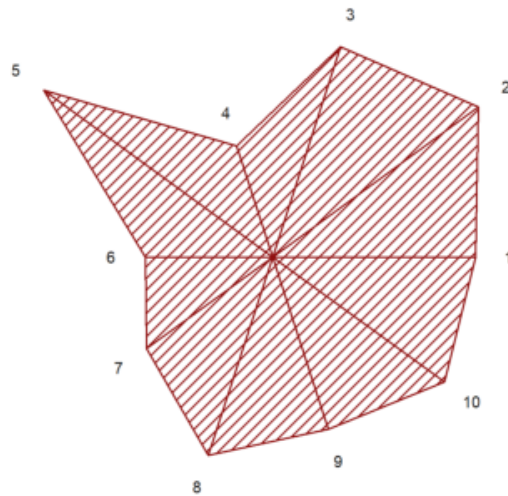


```
>function starplot1 (v, style="/", color=green, lab=none) ...
  if !holding() then clg; endif;
  w=window(); window(0,0,1024,1024);
  h=holding(1);
  r=max(abs(v))*1.2;
  setplot(-r,r,-r,r);
  n=cols(v); t=linspace(0,2pi,n);
  v=v|v[1]; c=v*cos(t); s=v*sin(t);
  cl=barcolor(color); st=barstyle(style);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

Tidak ada grid atau tanda sumbu di sini. Selain itu, kita menggunakan jendela penuh untuk plot ini.

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

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



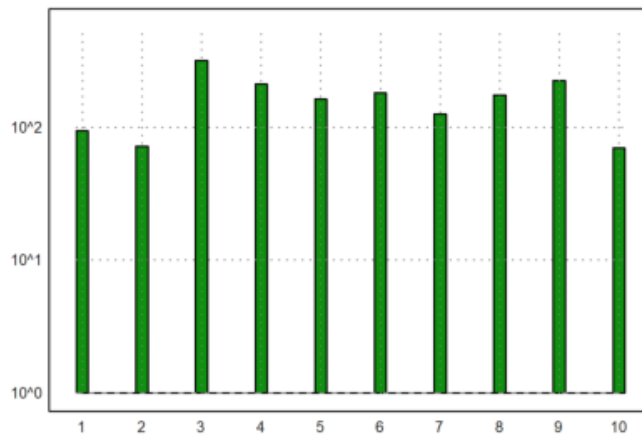
Terkadang, Anda mungkin ingin menggambar sesuatu yang tidak dapat dilakukan oleh plot2d, tetapi hampir bisa.

Dalam fungsi berikut, kita melakukan plot impulse logaritmik. plot2d dapat melakukan plot logaritmik, tetapi tidak untuk batang impulse.

```
>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 didistribusikan secara eksponensial.

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



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

Fungsi `wait(0)` memaksa plot muncul di jendela grafis. Sebaliknya, pembaruan terjadi dalam interval waktu yang jarang.

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

Tekan tombol apa saja untuk menghentikan animasi ini.

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

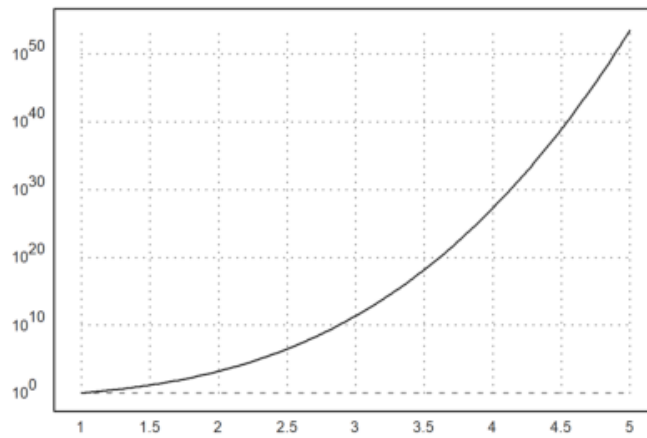
Logarithmic Plots

EMT menggunakan parameter "logplot" untuk skala logaritmik.

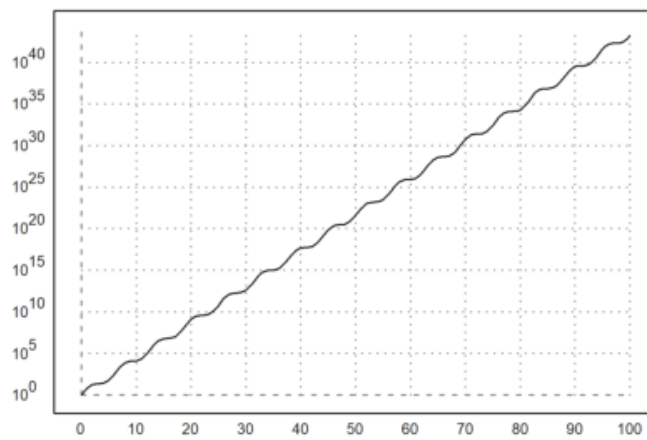
Plot logaritmik dapat digambar baik dengan menggunakan skala logaritmik di y dengan `logplot=1`, atau menggunakan skala logaritmik di x dan y dengan `logplot=2`, atau di x dengan `logplot=3`.

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

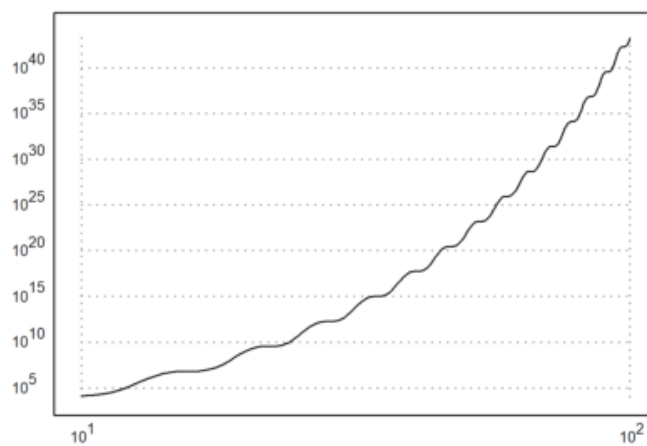
```
>plot2d("exp(x^3-x)*x^2",1,5,logplot=1):
```



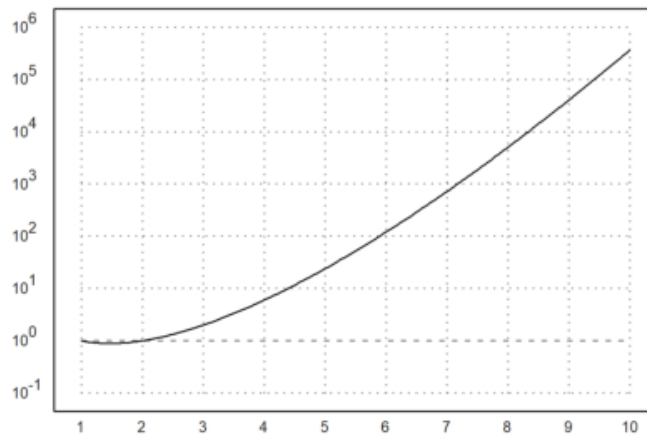
```
>plot2d("exp(x+sin(x))",0,100,logplot=1):
```



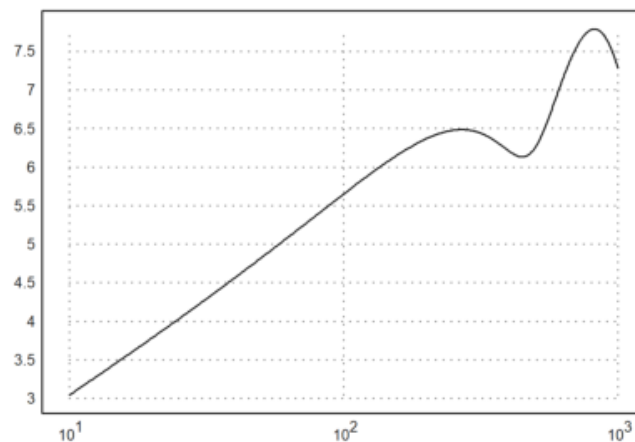
```
>plot2d("exp(x+sin(x))",10,100,logplot=2):
```



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

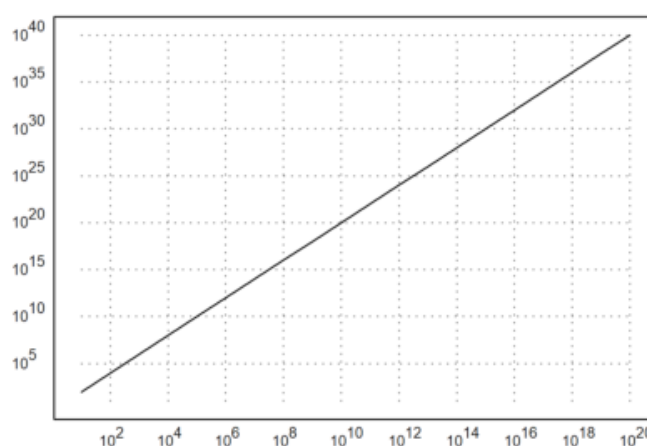



```
>plot2d("log(x*(2+sin(x/100)))",10,1000,logplot=3):
```



Ini juga berfungsi dengan plot data.

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

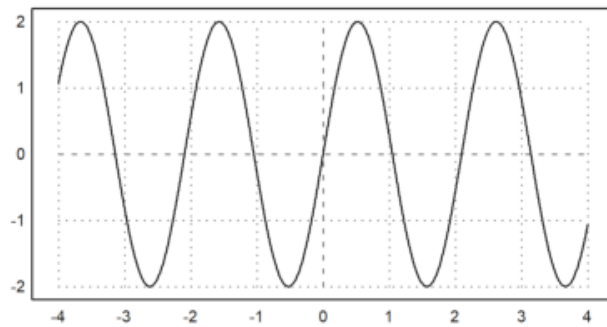


Soal-Soal

1. Gambarkan grafik fungsi

$$2\sin(3x)$$

```
>aspect(2); plot2d("2*sin(3*x)", -4, 4):
```

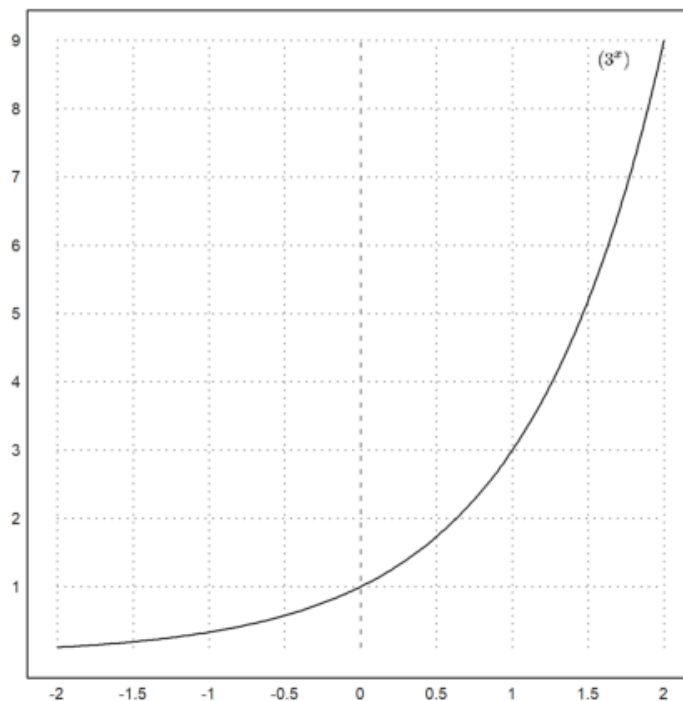


di penyelesaian ini, digunakan rasio 2:1 dan gMBr grafik dimulai dari -4 sampai 4

2. Gambarkan grafik fungsi eksponen

$$f(x) = 3^x$$

```
>plot2d("3^x"); label(latex("(3^x)"),1.5,9,color=black):
```

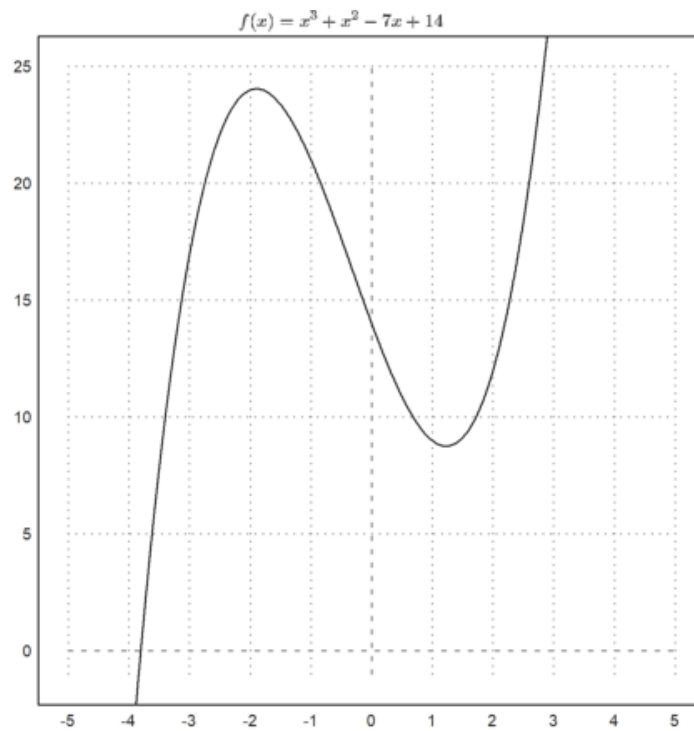


Pada grafik diatas ditambahkan label dengan format latex untuk fungsi 3^x

3. Lukislah grafik polinomial

$$f(x) = x^3 + x^2 - 7x + 14$$

```
>plot2d("x^3+x^2-7*x+14",-5,5,-1,25,grid=2);
>settitle(latex("f(x)=x^3+x^2-7x+14"),color=black):
```



Pada grafik diatas, ditambahkan judul dengan settitle dan latex

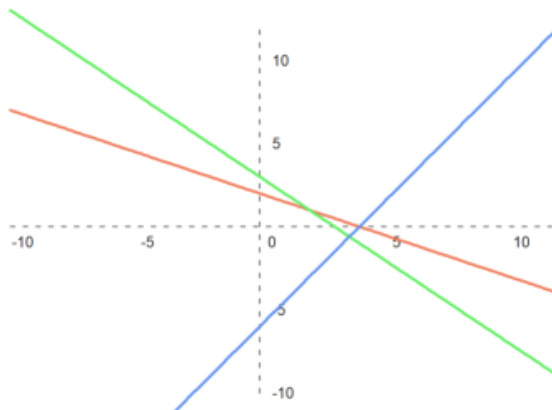
3. Gambarkan kurva yang memuat :

$$x + 2y = 4$$

$$x + y = 3$$

$$3x - 2y = 12$$

```
>plot2d(["(4-x)/2","(3-x)","(3*x-12)/2"] ...
,-10,12,-10,12,color=10:12,thickness=2,grid=3):
```



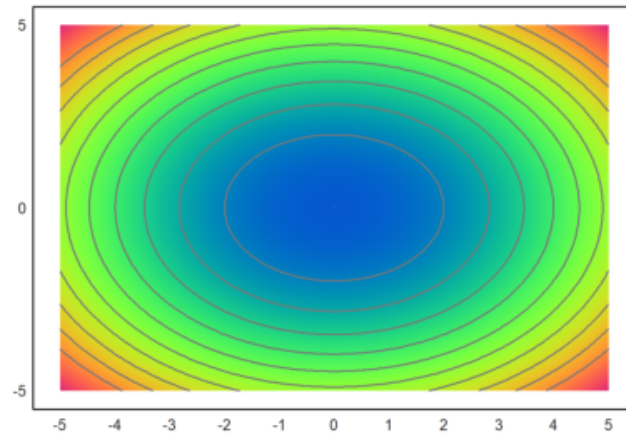
Disini diatur untuk tampilan grafiknya menggunakan grid=3 dimana akan menampilkan sumbu x-y dengan label di dalamnya.

4. Buatlah sketsa kurva

$$f(x, y) = 1/2(x^2 + y^2) \text{ dengan level bilangan genap dari 0 sampai 20}$$

```
>aspect(1.5);
>expr := "1/2*(x^2 + y^2)"; // mendefinisikan f(x,y)

>plot2d(expr,level=0:2:20,>hue,>spectral,n=200,grid=4,r=5):
```

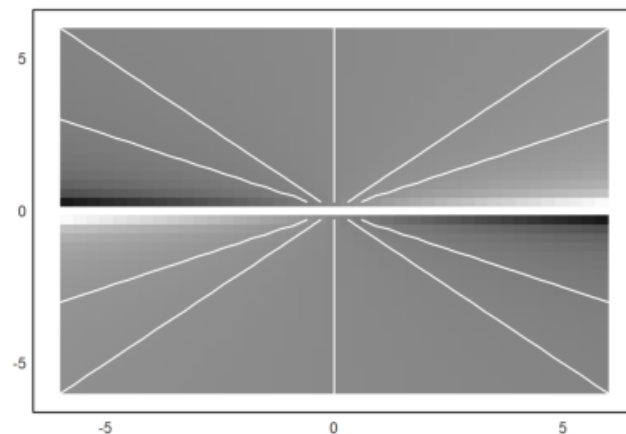


Untuk menggambarkan grafik diatas, fungsi disimpan pada variabel ekspresi kemudian plot grafik dengan level yang diinginkan dan menggunakan grid=4 supaya lebih jelas.

5. Buatlah sketsa kurva

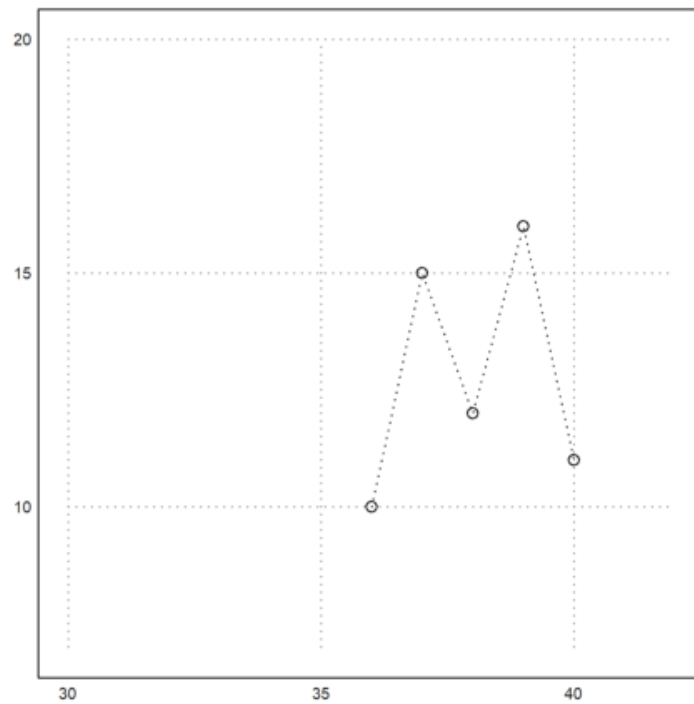
$$f(x, y) = x/y \text{ dengan level } -2, -1, 0, 1, 2$$

```
>aspect(1.5);
>z := "x/y";
>plot2d(z, level=-2:1:2, >hue, contourcolor=white, grid=4, r=6):
```



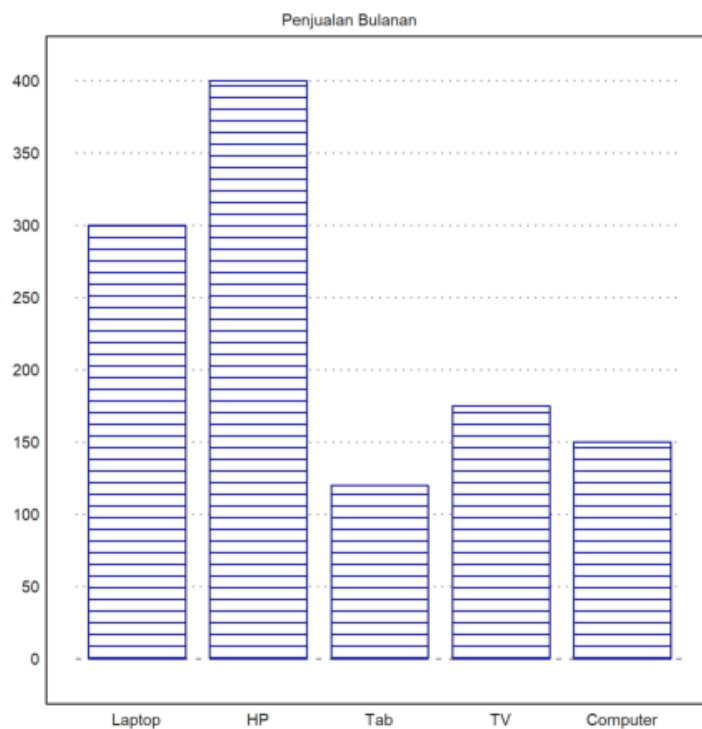
6. Terdapat data ukuran sepatu karyawan suatu supermarket. karyawan yang memiliki ukuran sepatu 36 ada 10 orang, 37 ada 15 orang, 38 ada 12 orang, 39 ada 16 orang dan 40 ada 11 orang. Buatlah grafik untuk data tersebut.

```
>xdata=[36,37,38,39,40]; ydata=[10,15,12,16,11]; // data
>plot2d(xdata,ydata,a=30,b=42,c=7,d=20,style="."); // lines
>plot2d(xdata,ydata,>points,>add,style="o"): // add points
```



7. Sebuah pabrik memproduksi Laptop, HP, Tablet, TV, serta Komputer. Setiap bulannya penjualan setiap produk berbeda. Laptop terjual 300 pcs, HP terjual 400 pcs, Tablet terjual 120 pcs, dan Komputer terjual 150 pcs setiap bulannya. Gambarkan diagram batang untuk penjualan bulanan produk pabrik itu.

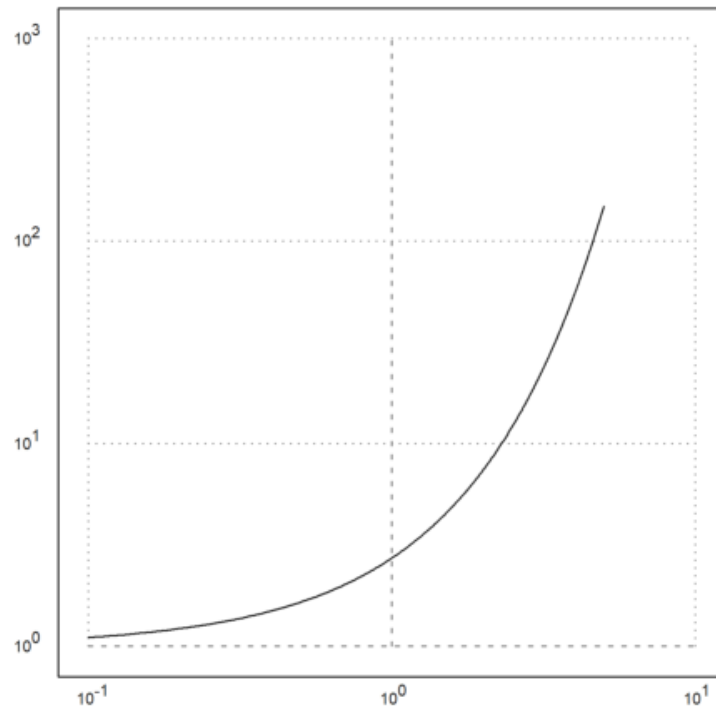
```
>product=["Laptop", "HP", "Tab", "TV", "Computer"];
>values=[300,400,120,175,150];
>columnsplot(values,lab=product,color=blue,style="--");
>title("Penjualan Bulanan"):
```



8. Gambarkan grafik fungsi

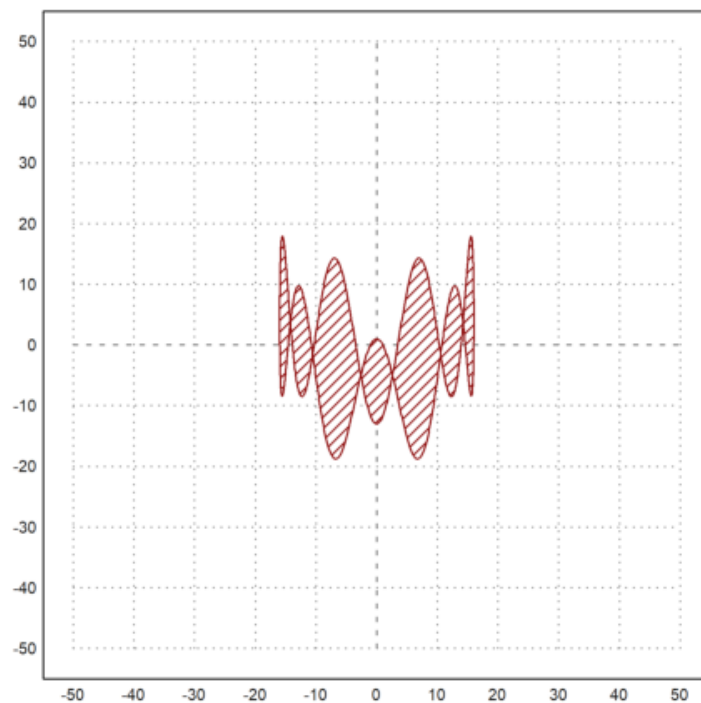
$y = e^x$ dalam rentang dari 0,1 sampai 5

```
>x=(0.1:0.01:5); y=exp(x);
>plot2d(x,y,logplot=2):
```



9. Gambar daerah yang dibatasi kurva

```
>t=linspace(0,2pi,1000); x=16*sin(t); y=5*cos(t)-12*cos(7*t)-5*cos(2*t)-cos(8*t); ...
>plot2d(x,y,>filled,fillcolor=red,style="/",r=50):
```



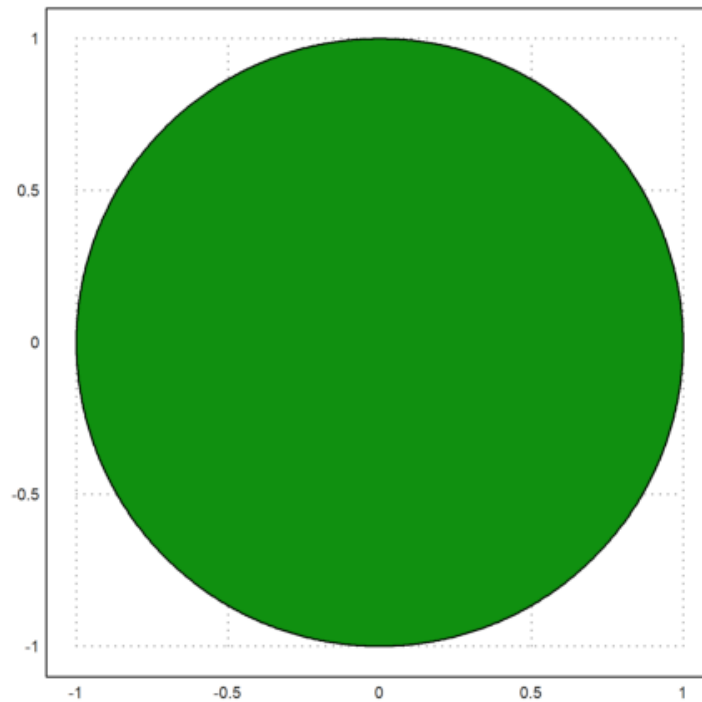
10.

$$x(t) = \cos(t) \text{ dan } y(t) = \sin(t)$$

dengan rentang t dari 0 hingga 2π

```
>t=linspace(0,2*pi,1000); ...
```

```
plot2d(cos(t),sin(t),r=1,>filled,fillcolor=green,style=".") :
```



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 "")

x1, y1 : 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 x1="string" or y1="string". Other labels can be added with the functions label() or labelbox(). The title can be a unicode string or an image of a Latex formula.

cgrid :

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

Overview

The function can plot

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

If a function or expression for xv is given, `plot2d()` will compute values in the given range using the function or expression. The expression must be an expression in the variable x . The range must be defined in the parameters a and b unless the default range $[-2,2]$ should be used. The y -range will be computed automatically, unless c and d are specified, or a radius r , which yields the range $[-r,r]$ for x and y . For plots of functions, `plot2d` will use an adaptive evaluation of the function by default. To speed up the plot for complicated functions, switch this off with `<adaptive`, and optionally decrease the number of intervals n . Moreover, `plot2d()` will by default use mapping. I.e., it will compute the plot element for element. If your expression or your functions can handle a vector x , you can switch that off with `<maps` for faster evaluation.

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