

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

Dalam contoh ini, kita hanya menggambar beberapa garis acak dengan berbagai warna. Untuk rincian tentang fungsi-fungsi ini, pelajari fungsi inti EMT.

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



```
>reset;
```

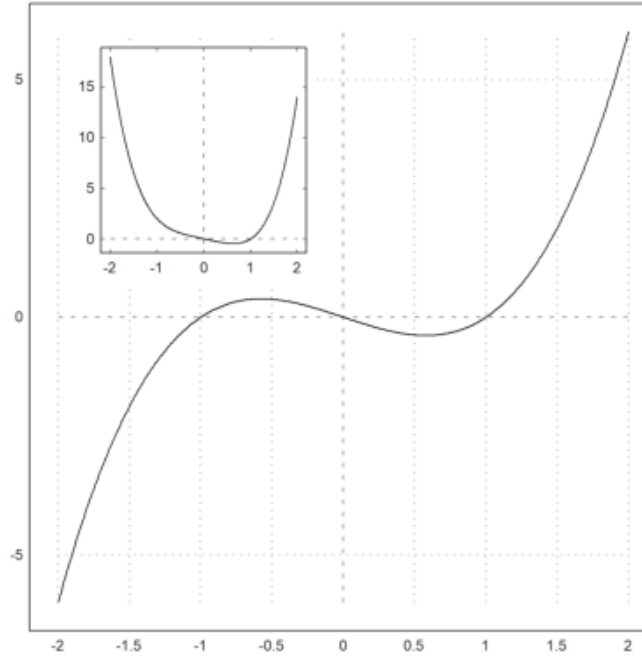
Perlu untuk menahan grafik, karena perintah `plot()` akan menghapus jendela plot.

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

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

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

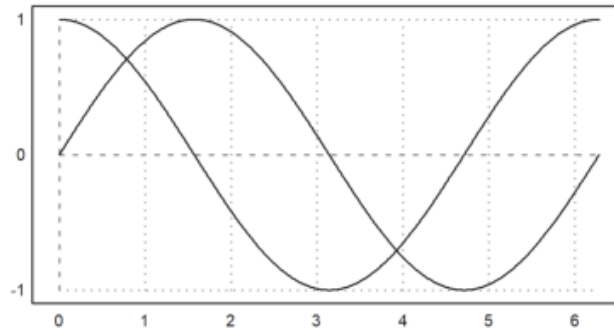
Plot dengan beberapa gambar dicapai dengan cara yang sama. Ada fasilitas fungsi `figure()` untuk ini.

Plot Aspect (Aspek Plot)

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

Tapi Anda juga bisa mengubahnya untuk satu plot. Untuk ini, ukuran area plot saat ini telah diubah, dan jendela diatur agar label memiliki ruang yang cukup.

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



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

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

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

Anda dapat membuat plot di Maxima menggunakan Gnuplot atau di Python menggunakan Math Plot Lib.

Euler dapat membuat plot 2D dari

- ekspresi
- fungsi, variabel, atau kurva terparameter,
- vektor nilai x-y,
- awan titik di bidang,(((clouds of points in the plane)))
- kurva implisit dengan level atau daerah level.
- Fungsi kompleks

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

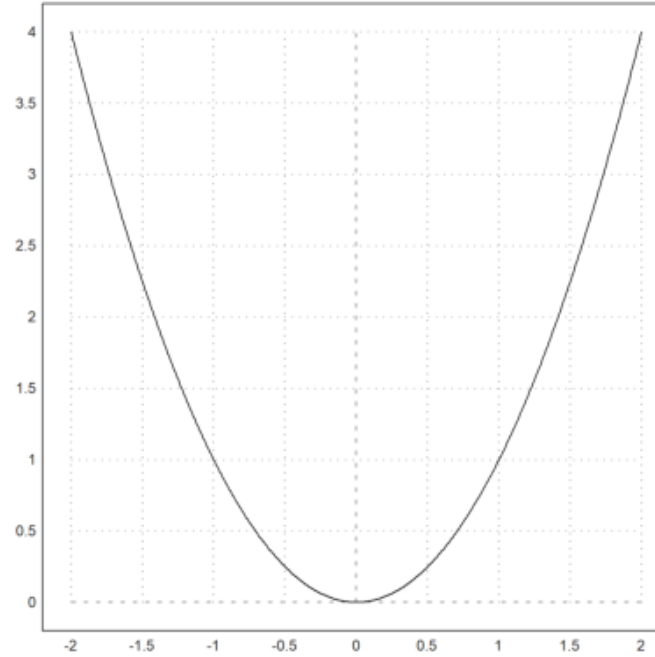
Plot Ekspresi atau Variabel

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

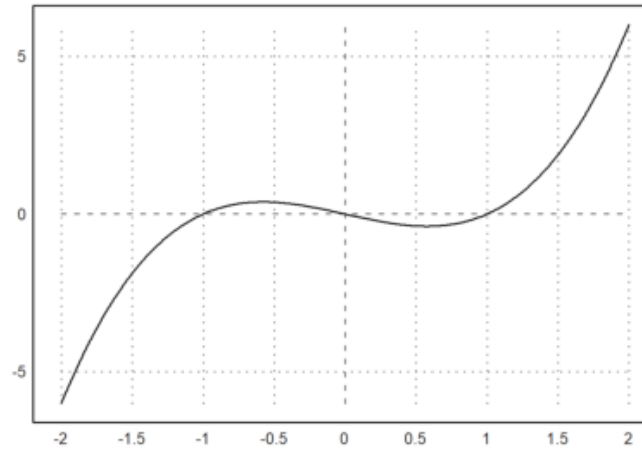
Berikut adalah contoh yang paling dasar, yang menggunakan rentang default dan menetapkan rentang y yang tepat untuk menyesuaikan plot fungsi tersebut.

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

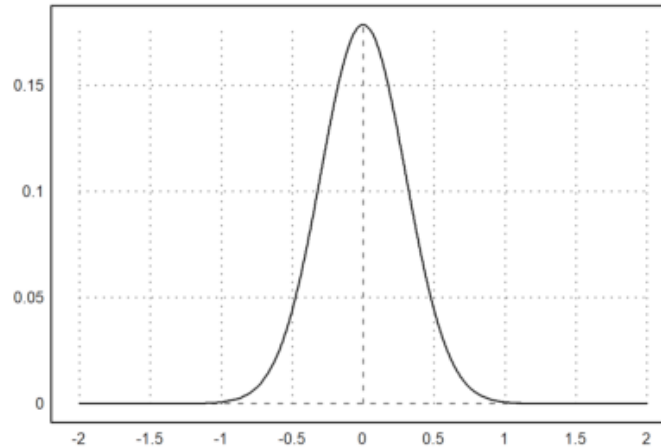
```
>plot2d("x^2"):
```

```
>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
```

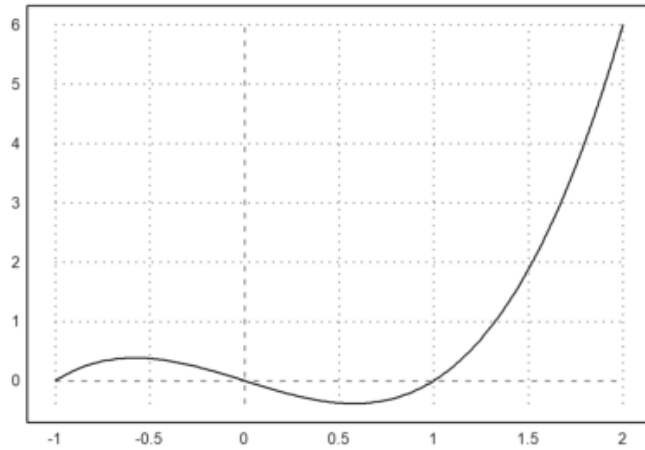


Dari beberapa contoh sebelumnya Anda dapat melihat bahwa aslinya gambar plot menggunakan sumbu X dengan rentang nilai dari -2 sampai dengan 2. Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan nilai-nilai batas X (dan Y) di belakang ekspresi yang digambar.

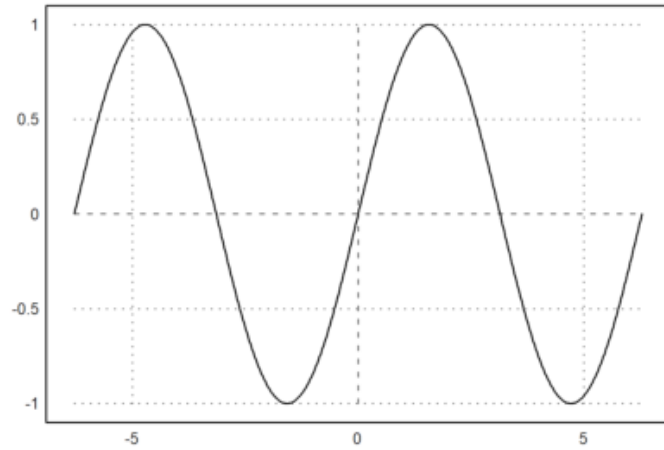
Rentang plot ditetapkan dengan parameter yang ditentukan sebagai berikut.

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

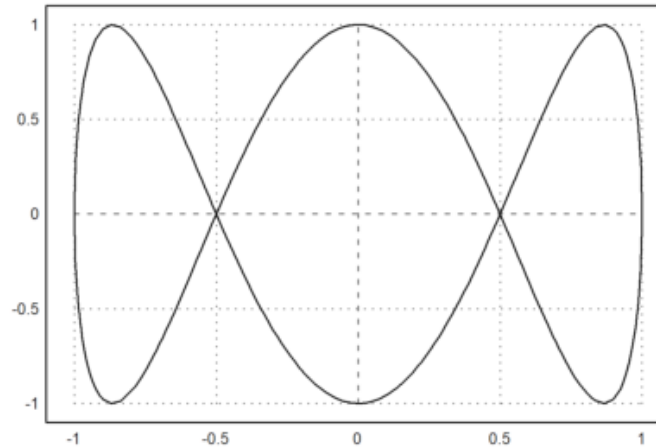
```
>plot2d("x^3-x",-1,2):
```



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



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



((xmin=0 dan xmax=2*pi menentukan rentang variabel x dari 0 hingga 2pi(sekitar 6,28) yang merupakan satu periode penuh untuk fungsi trigonometri)))

Alternatif untuk titik dua adalah perintah insing(lines), yang menyisipkan plot yang menempati sejumlah baris teks yang ditentukan.

Dalam opsi, plot dapat diatur untuk muncul.

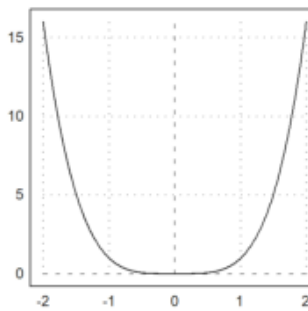
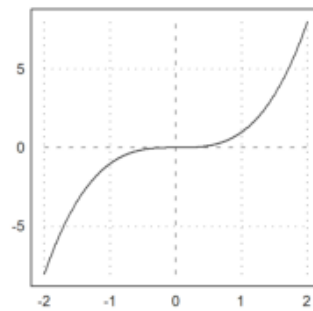
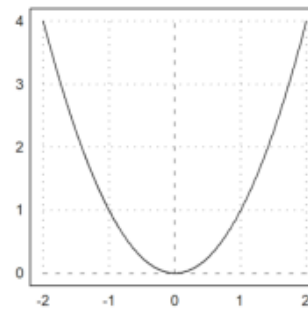
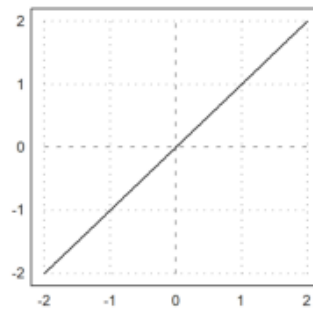
- di jendela terpisah yang dapat diubah ukurannya,
- di jendela notebook.

Lebih banyak gaya dapat dicapai dengan perintah plot tertentu.

Dalam beberapa kasus, tekan tombol tabulasi untuk melihat plot, jika itu tersembunyi.

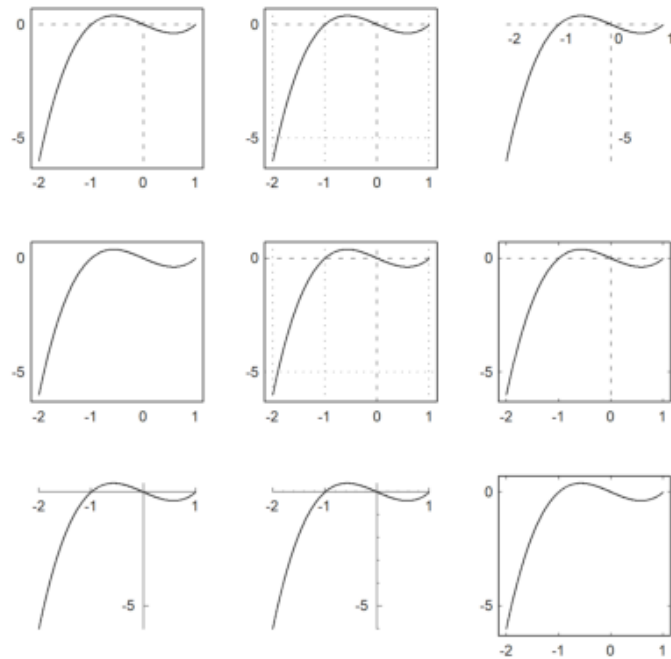
Untuk membagi jendela menjadi beberapa plot, gunakan perintah figure(). Dalam contoh ini, kita memplot x^1 hingga x^4 ke dalam 4 bagian jendela. figure(0) mengatur ulang jendela default.

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



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

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

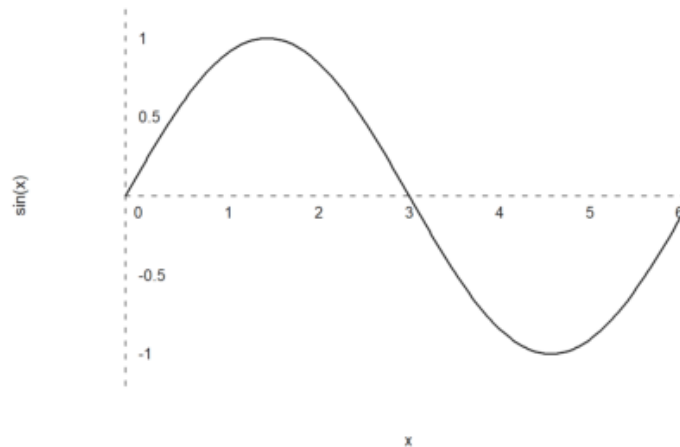


Jika argumen untuk `plot2d()` adalah sebuah ekspresi diikuti oleh empat angka, angka-angka tersebut adalah rentang `x` dan `y` untuk plot.

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

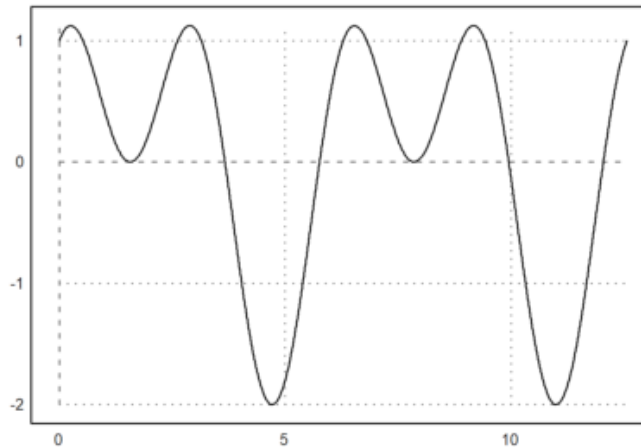
Dalam contoh berikut, kita mengubah gaya grid, menambahkan label, dan menggunakan label vertikal untuk sumbu `y`.

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



(((rasio besar x dan y nya 1:1.5, dengan batas sumbu x 0 sampai 2pi dan sumbu y -1.2 sampai 1.2, serta menggunakan tipe grid ketiga, yaitu hanya ada garis sumbu tanpa grid dan bingkai)))

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



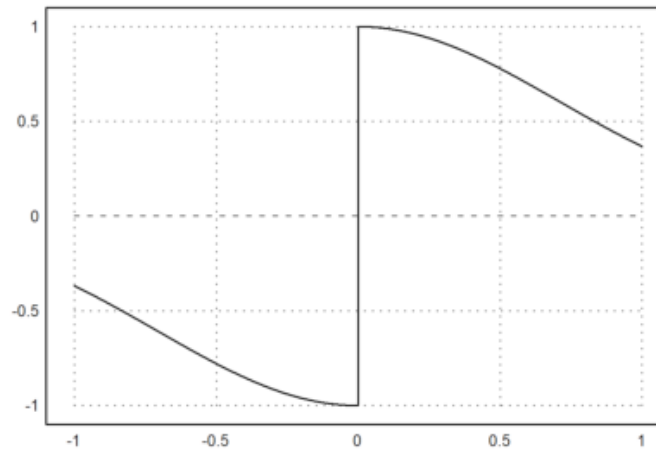
(((Plot dibuat dengan interval sumbu x dari 0 sampai 4 pi, sedangkan interval sumbu y tidak diatur, sehingga mengikuti defaultnya yang memenuhi nilai y max dan y min nya)))

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

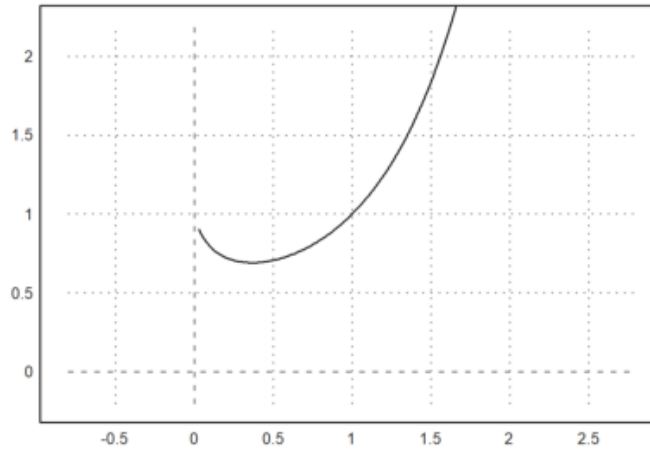
Anda bisa dengan mudah menandai gambar apa pun dan menyalinnya ke clipboard dengan Ctrl-C. Tentu saja, Anda juga dapat mengeksport grafik saat ini dengan fungsi-fungsi di menu File.

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

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

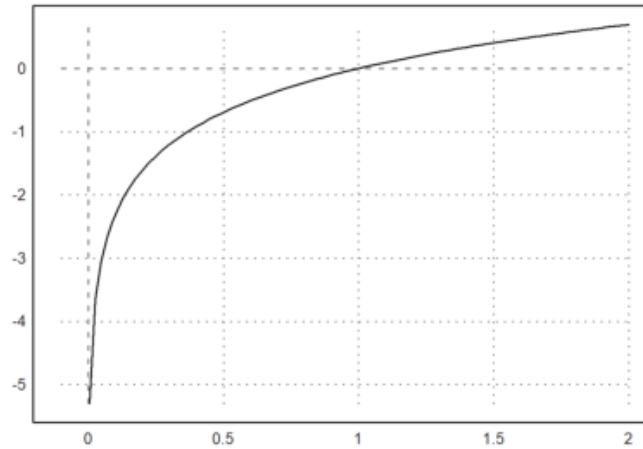


```
>plot2d("x^x",r=1.2,cx=1,cy=1):
```



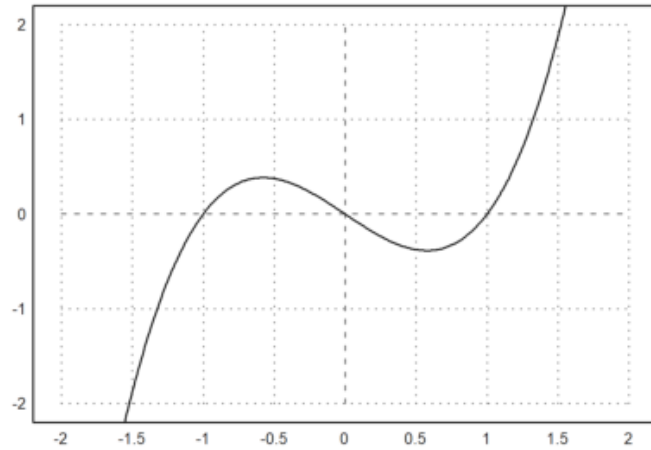
Perhatikan bahwa x^x tidak terdefinisi untuk $x \leq 0$. Fungsi `plot2d` menangkap kesalahan ini, dan mulai menggambar segera setelah fungsi didefinisikan. Ini berlaku untuk semua fungsi yang mengembalikan NAN di luar rentang definisinya.

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

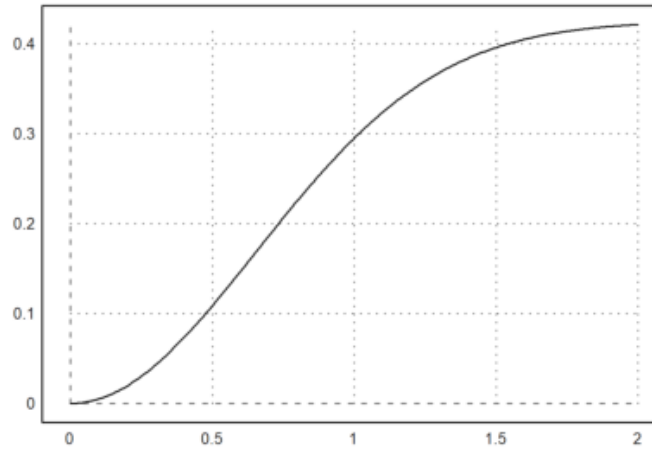


Parameter `square=true` (atau `>square`) secara otomatis memilih rentang y sehingga hasilnya adalah jendela plot berbentuk persegi. Perhatikan bahwa secara default, Euler menggunakan ruang persegi di dalam jendela plot.

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

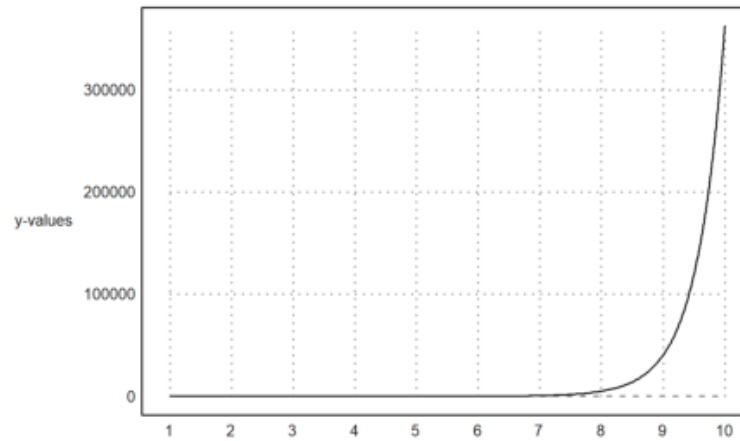


```
>plot2d(''integrate("sin(x)*exp(-x^2)",0,x)'',0,2): // plot integral
```



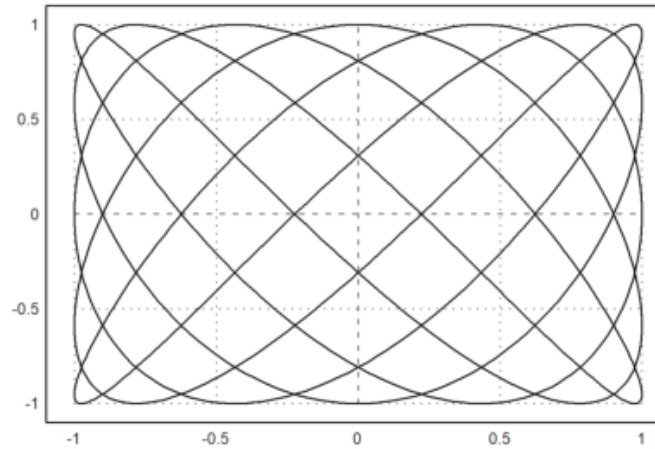
Jika Anda memerlukan lebih banyak ruang untuk label sumbu-y, panggil `shrinkwindow()` dengan parameter yang lebih kecil, atau atur nilai positif untuk "smaller" di `plot2d()`.

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

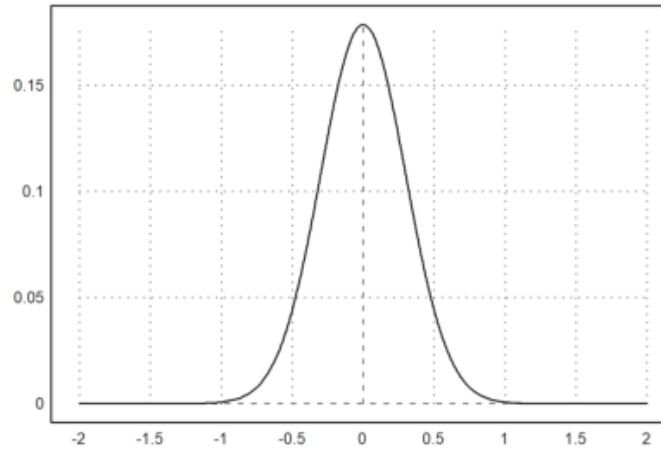


Ekspresi simbolik juga dapat digunakan, selama mereka disimpan sebagai ekspresi string sederhana.

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

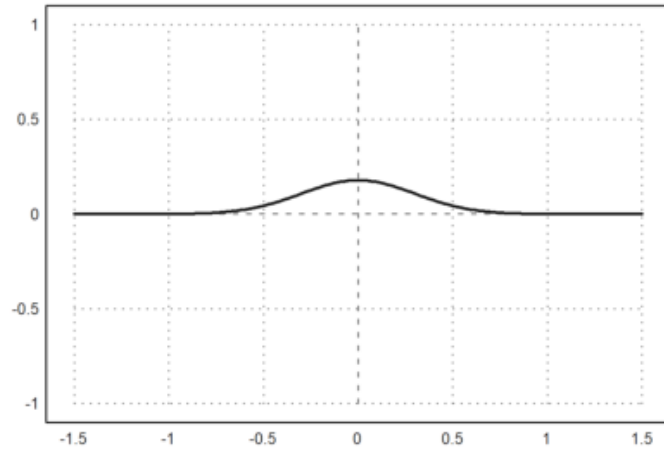



```
>a:=5.6; expr &= exp(-a*x^2)/a; // definisikan ekspresi  
>plot2d(expr,-2,2): // plot dari -2 sampai 2
```



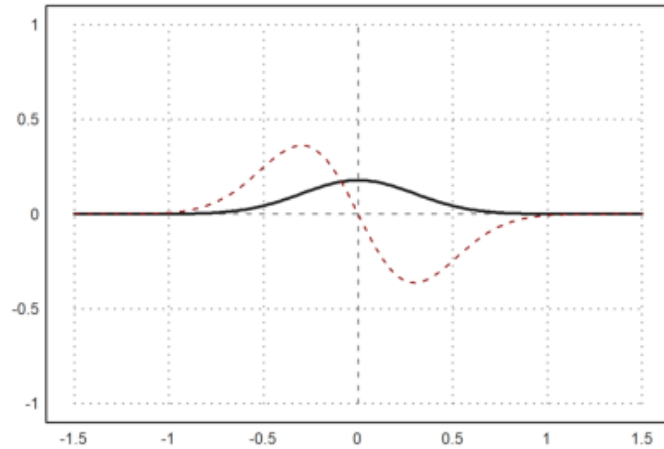
((((membuat plot dari fungsi yang sebelumnya telah didefinisikan, dengan rentang x dari -1 hingga 2, sedangkan y tidak diatur, sehingga mengikuti default yang mengakomodir tampilan plot secara sempurna)))

```
>plot2d(expr,r=1,thickness=2): // plot pada sebuah kotak sekitar (0,0)
```



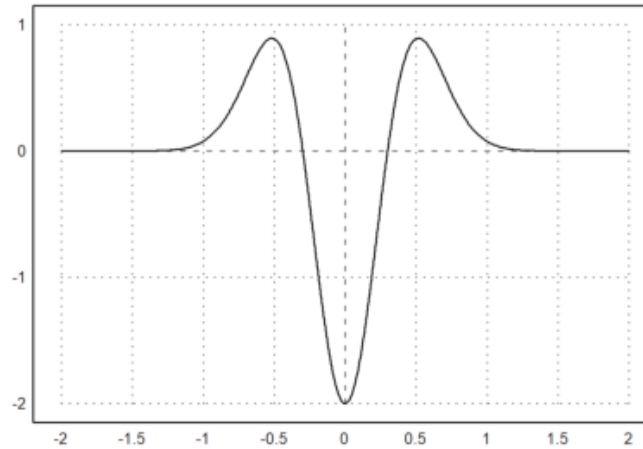
(((membuat garis lebih tebal dengan menambahkan fungsi thickness=2, default nya adalah 1)))

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



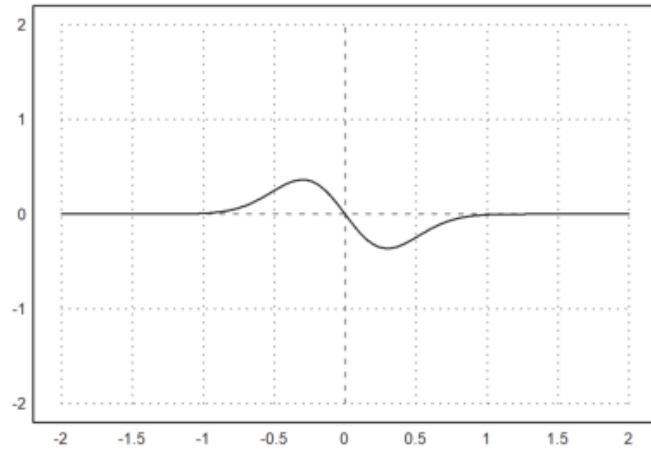
(((menambahkan plot lain, dengan style yang berbeda(-) dan warna diatur menjadi merah, defaultnya hitam)))

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

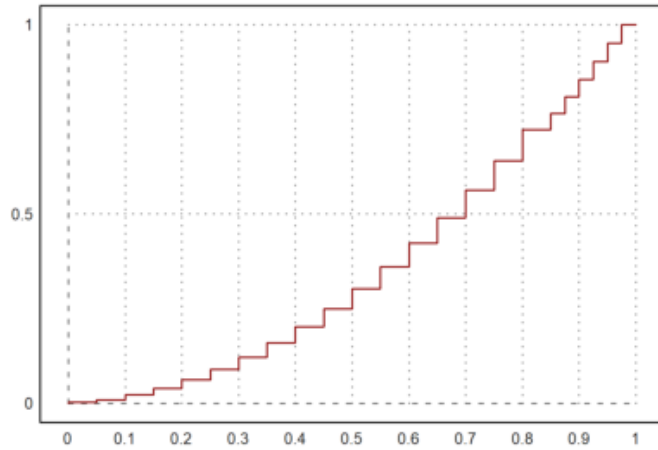


(((pada plot ini semua intervalnya diatur, rentang x di (a,b) dan rentang y di (c,d).)))

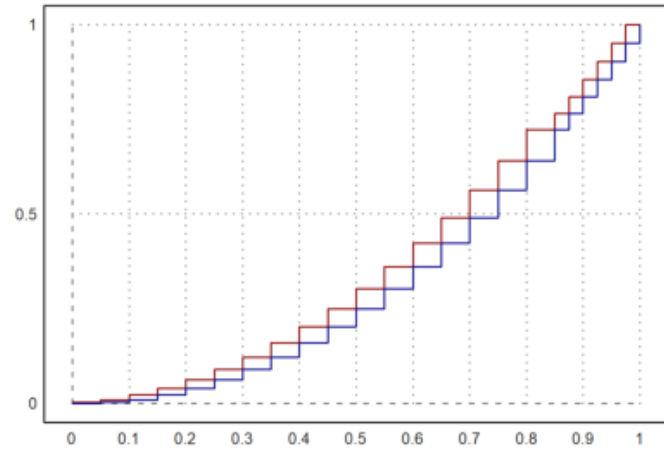
```
>plot2d(&diff(expr,x),a=-2,b=2,>square): // membuat plot persegi
```



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



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

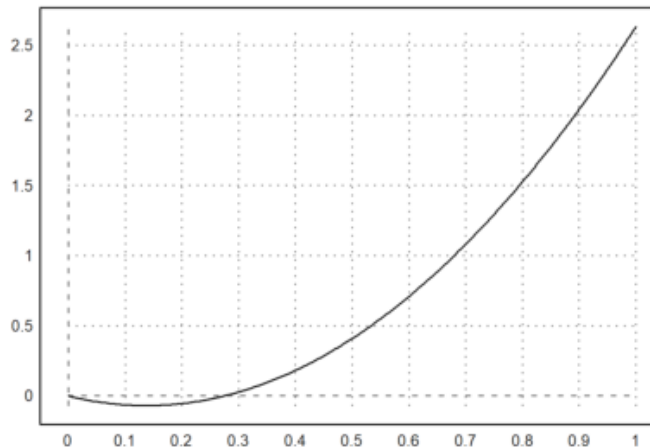


Fungsi-Fungsi dengan Satu Parameter

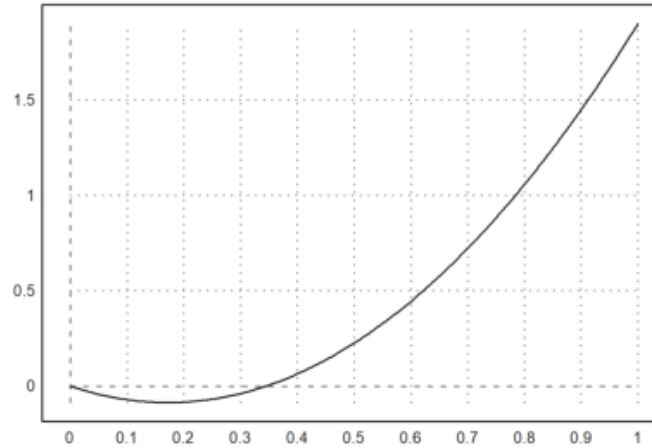
Fungsi pemetaan yang paling penting untuk plot planar adalah `plot2d()`. Fungsi tersebut diimplementasikan dalam bahasa Euler di file "plot.e", yang dimuat pada awal program.

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

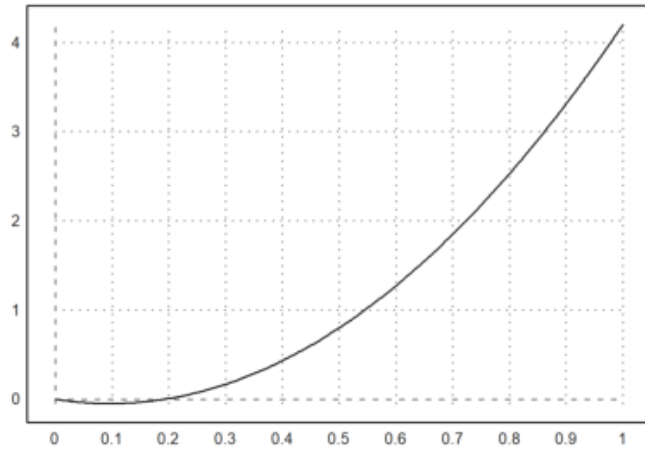
```
>function f(x,a) := x^2/a+a*x^2-x; // definisikan fungsi a  
>a=0.3; plot2d("f",0,1;a): // plot dengan a=0.3
```



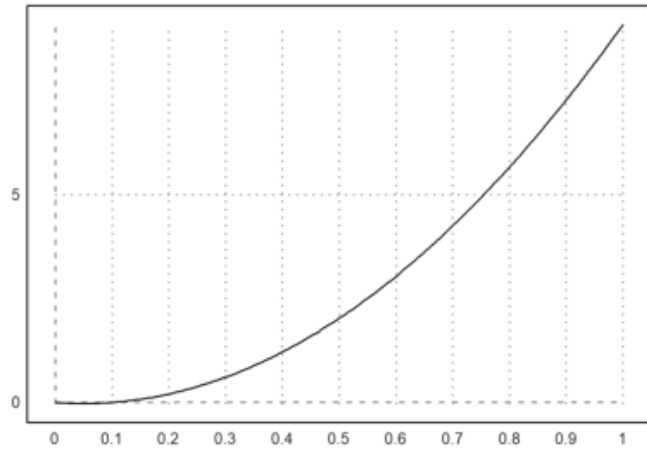
```
>plot2d("f",0,1;0.4): // plot dengan a=0.4
```



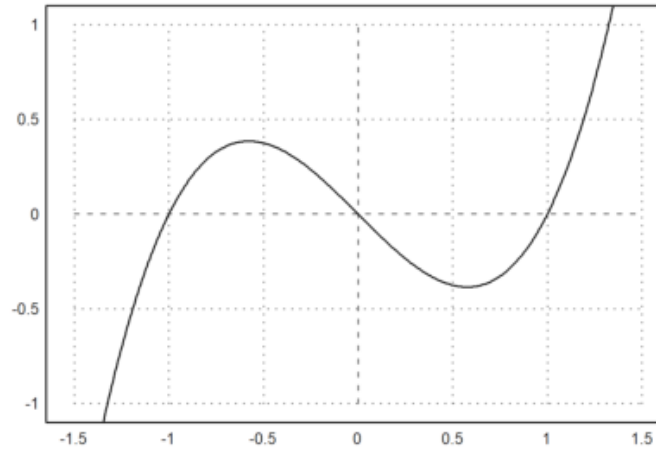
```
>plot2d({"f",0.2}),0,1): // plot dengan a=0.2
```



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



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



Berikut adalah ringkasan dari fungsi-fungsi yang diterima.

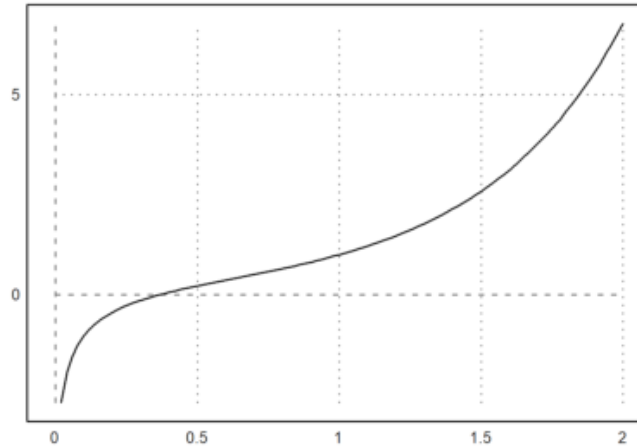
- ekspresi atau ekspresi simbolik dalam x
- fungsi atau fungsi simbolik dengan nama "f"
- fungsi simbolik hanya dengan nama f

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

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

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

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

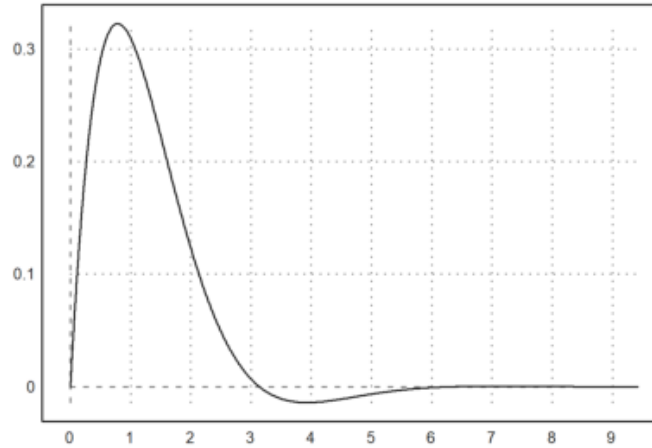


Tentu saja, untuk ekspresi atau ekspresi simbolik, nama variabel sudah cukup untuk memplotnya.

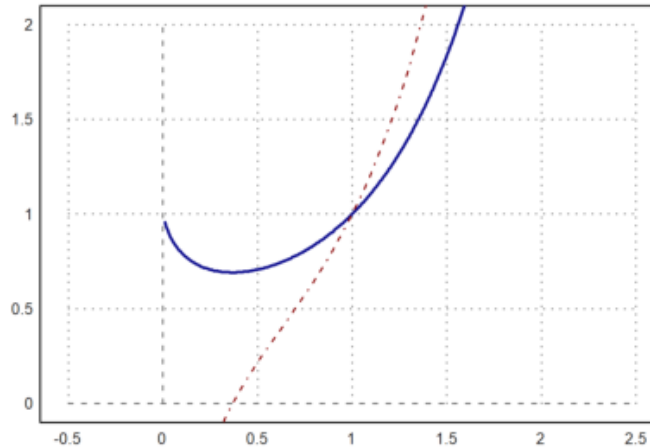
```
>expr &= sin(x)*exp(-x)
```

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

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



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



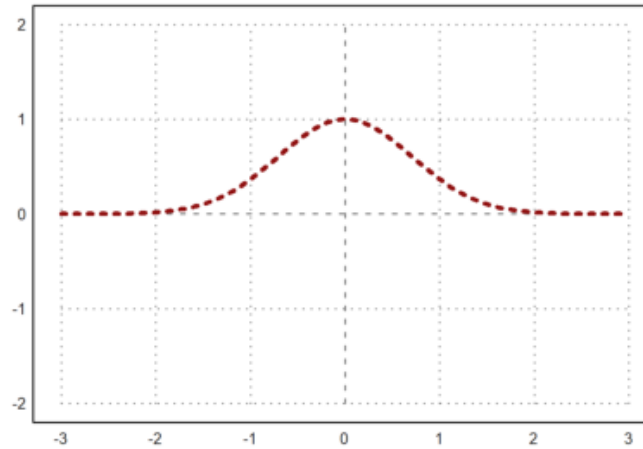
Untuk gaya garis, ada berbagai pilihan.

- gaya="...". Pilih dari "-", "_", "-.", ".", "-.-", "-.-".
- warna: Lihat di bawah untuk warna-warna.
- ketebalan: Default adalah 1.

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

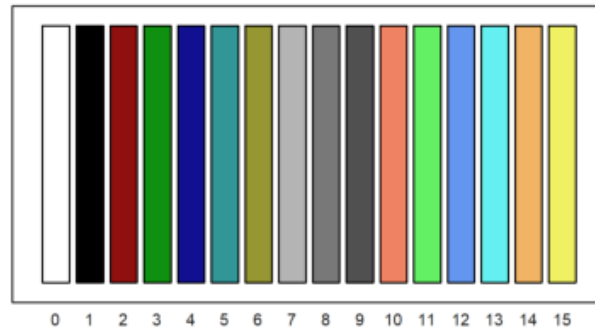
- 0..15: indeks warna default.
- konstanta warna: putih, hitam, merah, hijau, biru, cyan, zaitun, abu-abu terang, abu-abu, abu-abu gelap, oranye, hijau terang, turquoise, biru terang, oranye terang, kuning
- rgb(red,green,blue): parameter adalah bilangan riil dalam $[0,1]$.

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

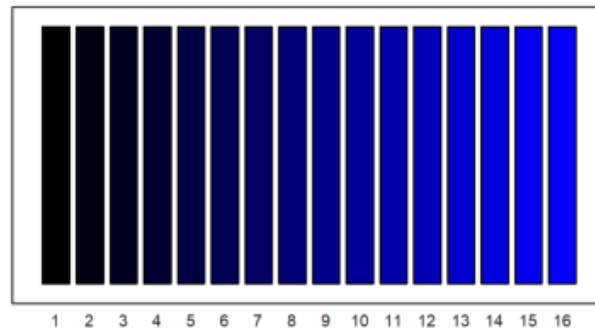
Berikut adalah tampilan dari warna-warna yang telah ditentukan sebelumnya pada EMT.

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



Tetapi Anda dapat menggunakan warna apapun.

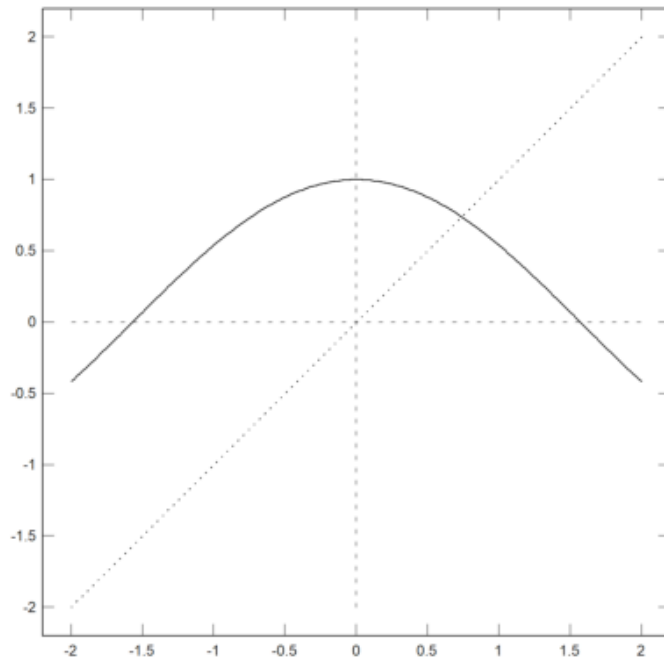
```
>columnsplot(ones(1,16),grid=0,color=rgb(0,0,linspace(0,1,15))):
```



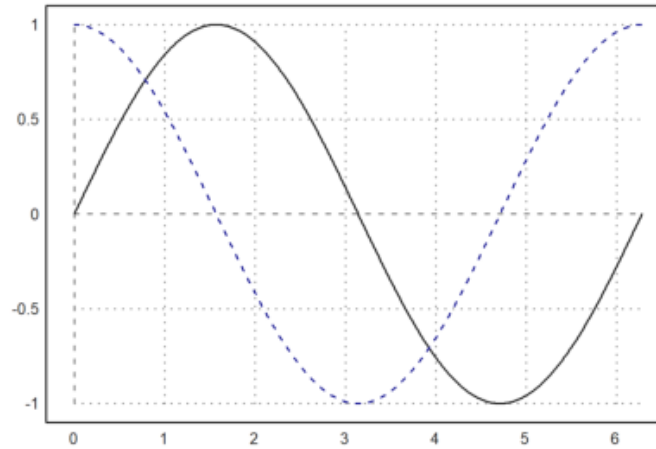
Menggambar Beberapa Kurva pada Bidang Koordinat yang Sama

Memplot lebih dari satu fungsi (fungsi ganda) dalam satu jendela dapat dilakukan dengan berbagai cara. Salah satu metode adalah dengan menggunakan `>add` untuk beberapa panggilan ke `plot2d`, kecuali panggilan yang pertama. Kami sudah menggunakan fitur ini dalam contoh-contoh di atas.

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

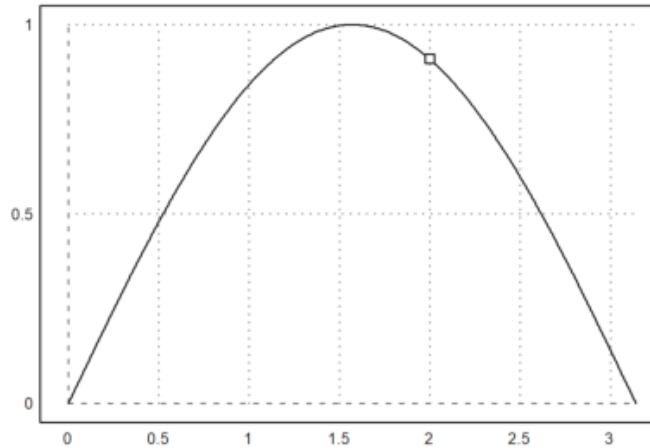


```
>aspect(1.5); plot2d("sin(x)",0,2pi); plot2d("cos(x)",color=blue,style="--",>add):
```



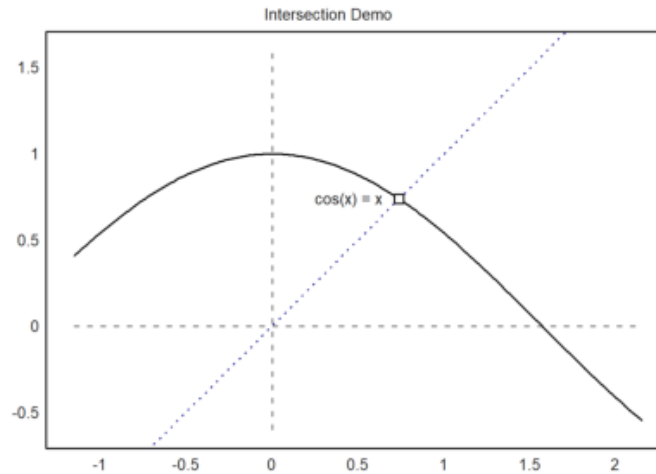
Salah satu kegunaan `>add` adalah untuk menambahkan titik pada kurva.

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



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

```
>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. Kita menghitung ekspansi ini menggunakan maxima melalui ekspresi simbolik.

Plot ini dilakukan dengan perintah multi-baris berikut dengan tiga panggilan ke `plot2d()`. Panggilan kedua dan ketiga menggunakan `flat >add`, yang membuat plot menggunakan rentang plot sebelumnya.

Kami juga menambahkan kotak label yang menjelaskan fungsi-fungsi tersebut.

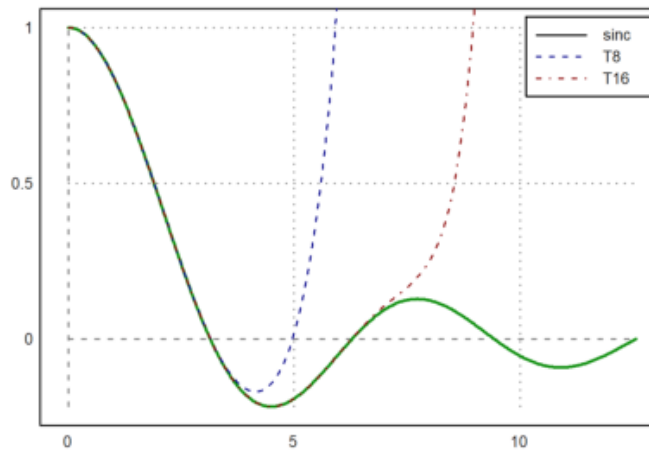
```
>$taylor(sin(x)/x,x,0,4)
```

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

```

>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]):

```



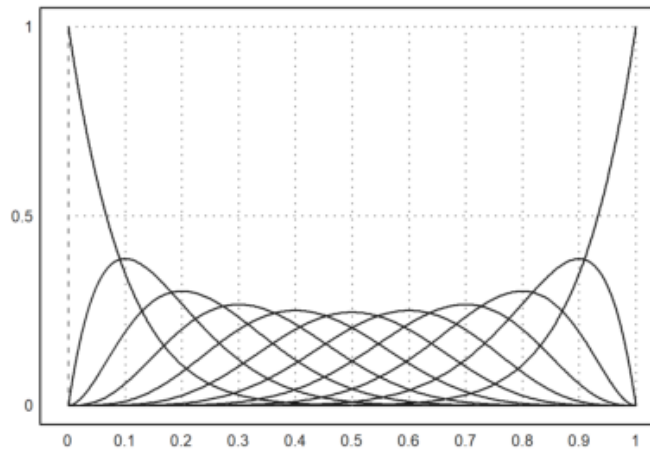
Pada contoh berikut, kami menghasilkan Polinomial Bernstein.

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


```

>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;

```



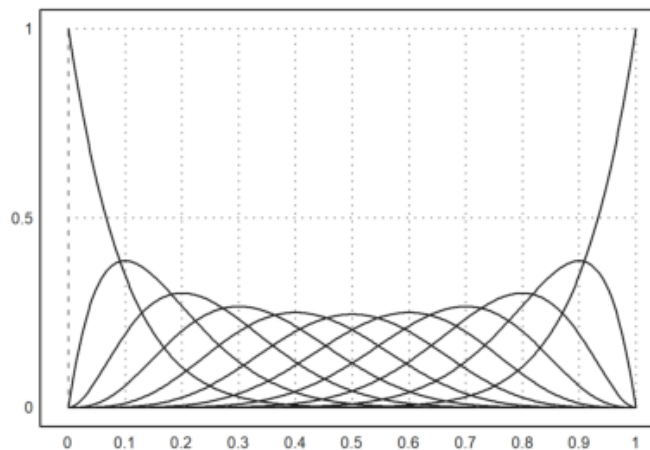
Metode kedua adalah dengan menggunakan sepasang matriks nilai x dan matriks nilai y dengan ukuran yang sama.

Kami menghasilkan matriks nilai dengan satu Polinomial Bernstein di setiap baris. Untuk ini, kami cukup menggunakan vektor kolom dari i. Lihatlah pengantar tentang bahasa matriks untuk mempelajari lebih lanjut tentang detailnya.

```

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

```

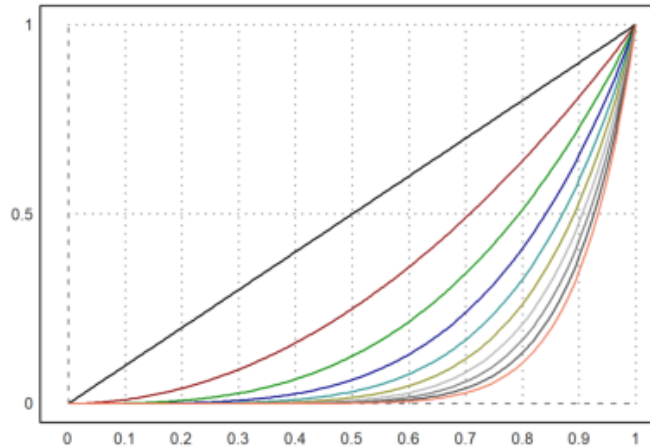


Perhatikan bahwa parameter warna dapat berupa vektor. Kemudian setiap warna digunakan untuk setiap baris pada matriks.

```

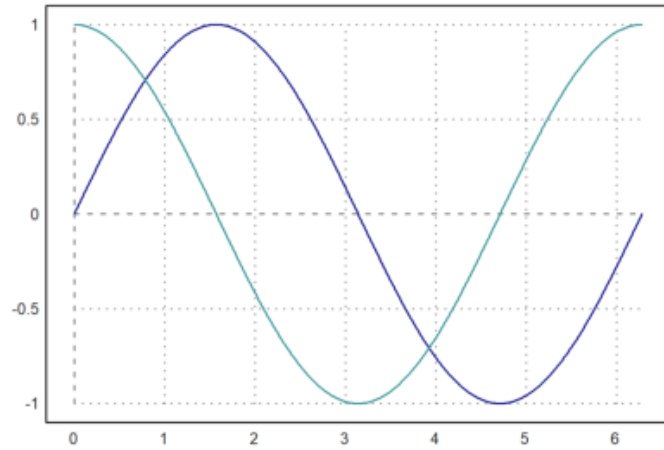
>x=linspace(0,1,200); y=x^(1:10)'; plot2d(x,y,color=1:10):

```

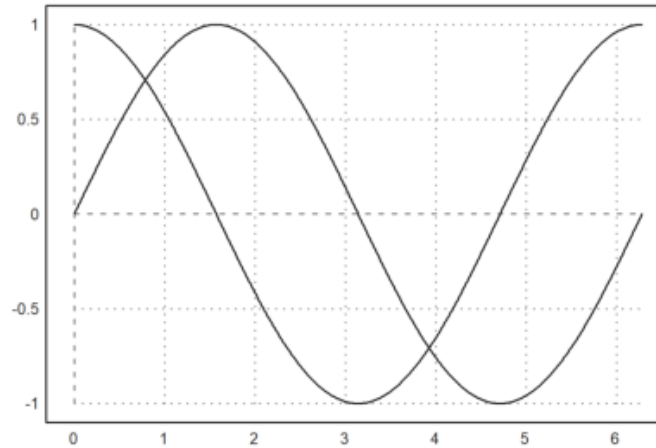


Metode lain yaitu menggunakan sebuah vektor ekspresi (string). Anda dapat menggunakan sebuah susunan warna, susunan gaya, dan susunan ketebalan yang memiliki panjang yang sama.

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



```
>plot2d(["sin(x)","cos(x)"],0,2pi): // plot ekspresi vektor
```



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

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

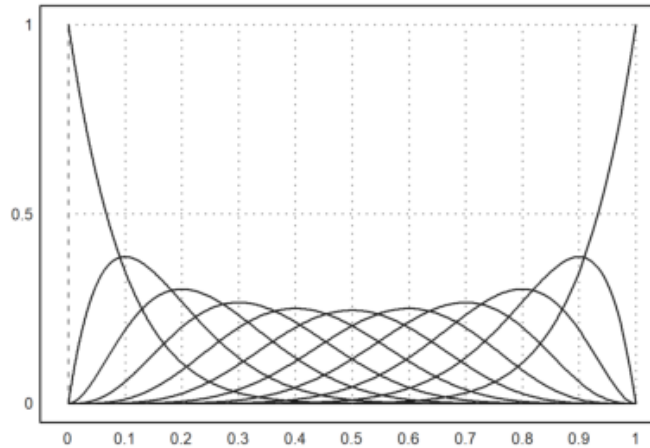
```

      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) // memperoleh vektor string dari vektor simbolik
```

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

```
>plot2d(mxm2str(v),0,1): // memplot fungsi
```

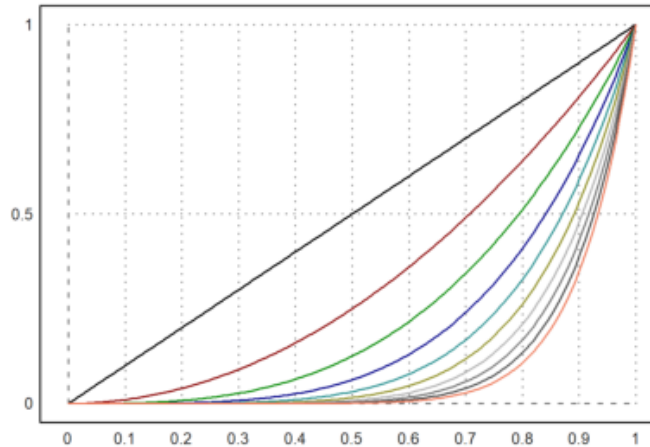


Alternatif lainnya yaitu menggunakan bahasa matriks Euler.

Jika sebuah ekspresi menghasilkan sebuah matriks fungsi, dengan satu fungsi setiap barisnya, semua vektor ini akan diplot menjadi satu plot.

Untuk ini, gunakan vektor parameter berbentuk vektor kolom. Jika sebuah susunan warna ditambahkan, ia akan digunakan untuk setiap baris plotnya.

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

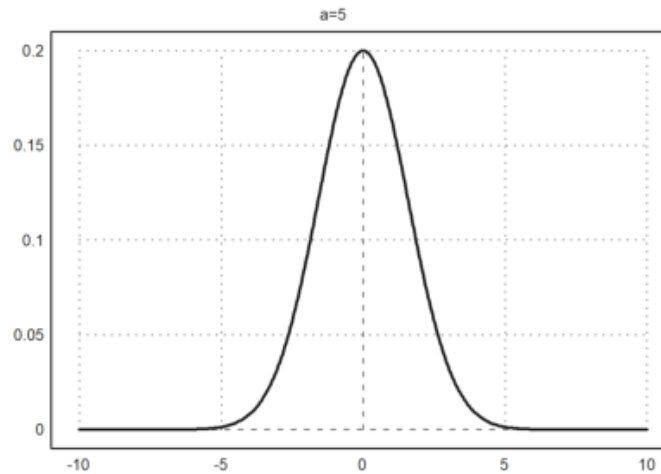


Ekspresi dan fungsi satu-baris dapat mengakses variabel global.

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

Pastikan untuk menempatkan semua parameter yang ditetapkan di akhir perintah plot2d. Pada contoh ini, kita memasukkan $a=5$ ke dalam fungsi f , yang kita plot dari -10 sampai 10.

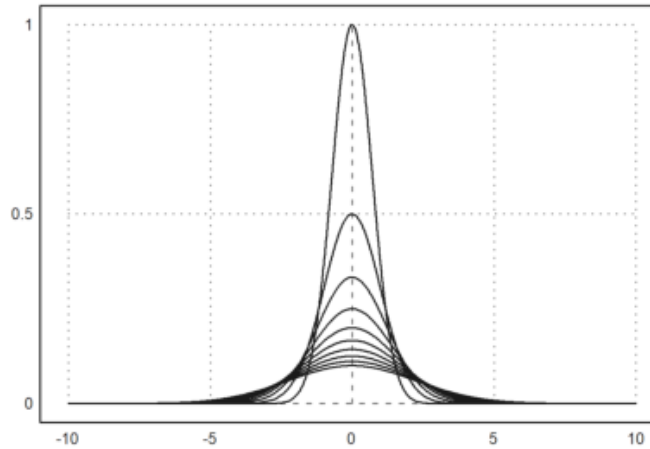
```
>function f(x,a) := 1/a*exp(-x^2/a); ...
>plot2d("f",-10,10;5,thickness=2,title="a=5"):
```

Alternatifnya, gunakan koleksi dengan nama fungsi dan semua parameter tambahan. Daftar khusus ini dinamakan koleksi panggilan (call collections), dan ini merupakan cara yang dipilih untuk mengirimkan argumen ke fungsi yang telah dikirimkan sendiri oleh fungsi tersebut ke fungsi yang lain.

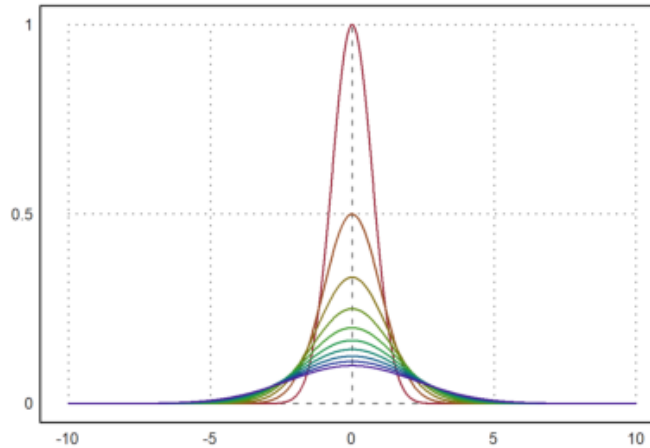
Pada contoh berikut, kita menggunakan sebuah loop untuk memplot beberapa fungsi (lihatlah tutorial tentang pemrograman loop)

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



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

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



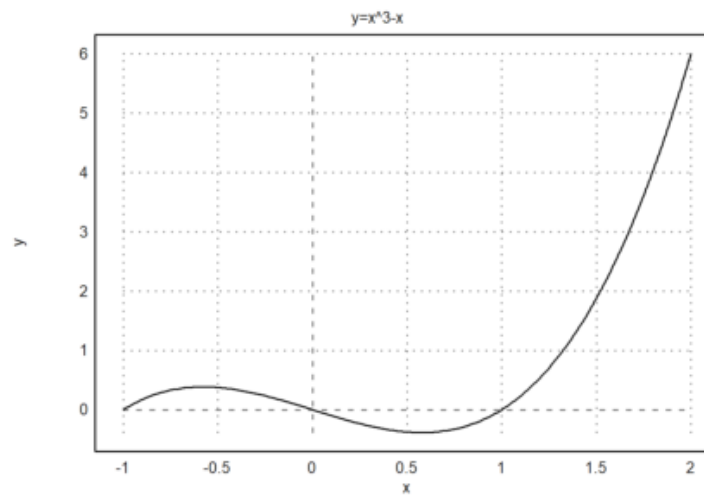
Label Teks

Dekorasi sederhana dapat meliputi :

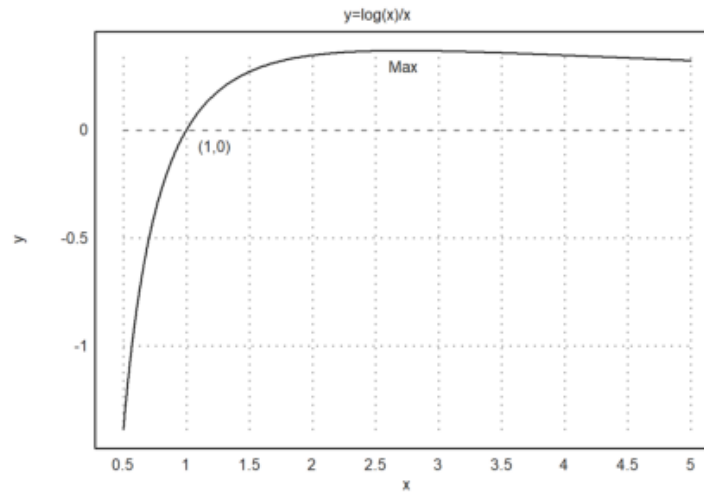
- Judul dengan `title="..."`
- Label sumbu x dan y dengan `xl="..."`, `yl="..."`
- label teks yang lain dengan `label("...",x,y)`

Perintah label akan menempatkan teks ke dalam plot yang saat ini digunakan pada koordinat plot (x,y). Ini dapat menerima argumrn posisi.

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

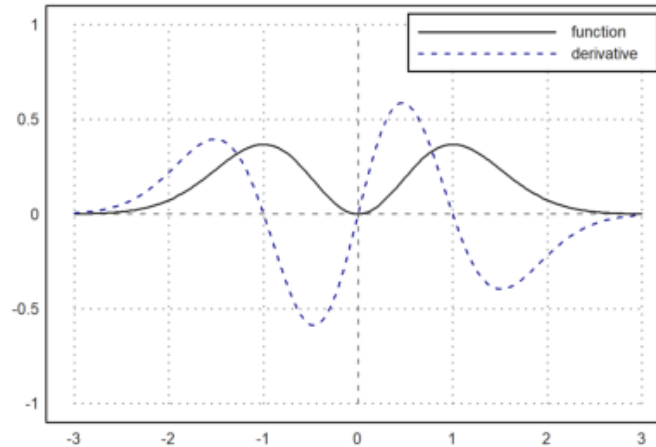


```
>expr := "log(x)/x"; ...  
> plot2d(expr,0.5,5,title="y="+expr,xl="x",yl="y"); ...  
> label("(1,0)",1,0); label("Max",E,expr(E),pos="lc"):
```



Ada juga fungsi `labelbox()`, yang dapat menampilkan fungsi beserta teks. Fungsi tersebut menerima vektor string dan warna, satu item untuk masing-masing 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):
```

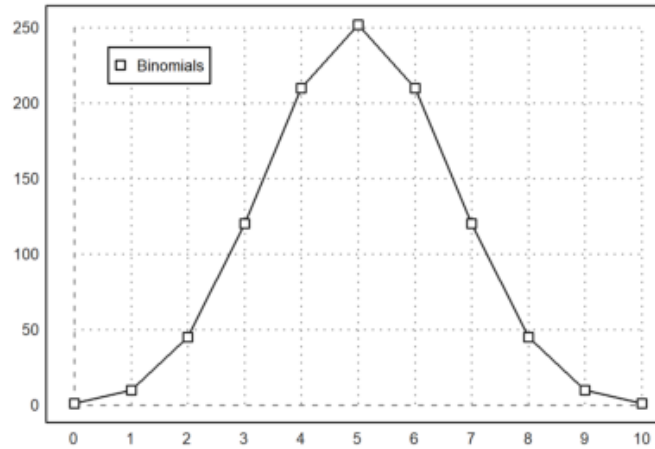


Kotak label secara default diletakkan di sebelah kanan atas, tetapi `>left` meletakkannya di sebelah kiri atas. Anda bisa memindahkannya kemanapun yang Anda suka. Posisi peletakannya di ujung kanan atas kotak, dan angka-angkanya adalah fraksi dari jendela grafik. Lebar kotaknya bersifat otomatis.

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

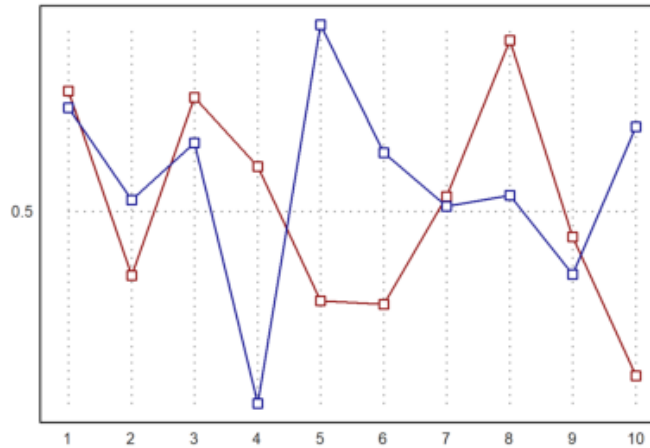
Pada contoh berikut, hanya ada satu fungsi. Jadi kita dapat menggunakan string daripada vektor string. Kita mengatur warna teks menjadi hitam untuk contoh ini.

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



Gaya plot ini juga tersedia pada `statplot()`. Seperti pada `plot2d()`, warnanya dapat diatur untuk setiap baris plot. Ada banyak lagi plot khusus untuk tujuan statistik (lihatlah tutorial tentang statistik).

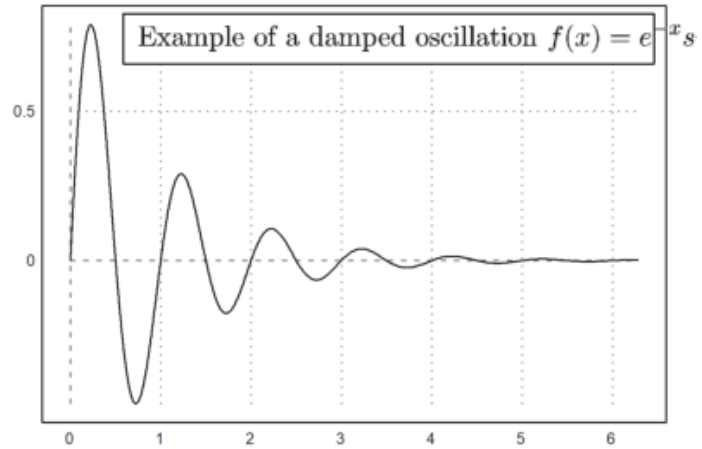
```
>statplot(1:10,random(2,10),color=[red,blue]):
```



Fitur serupa adalah fungsi `textbox()`.

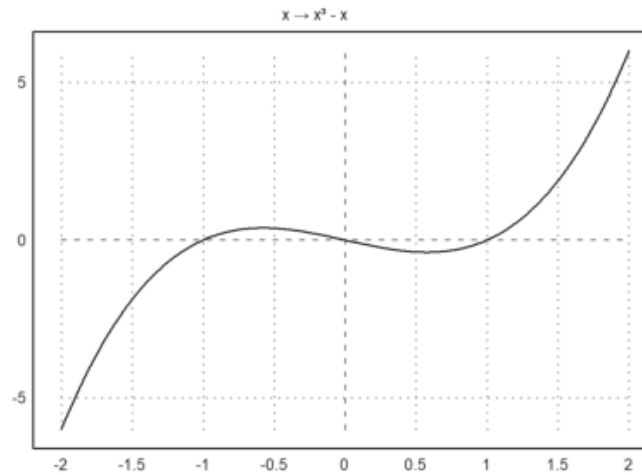
Lebar nya secara default mengikuti lebar maksimal dari baris teks. Tetapi dapat diatur oleh pengguna juga.

```
>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):
```

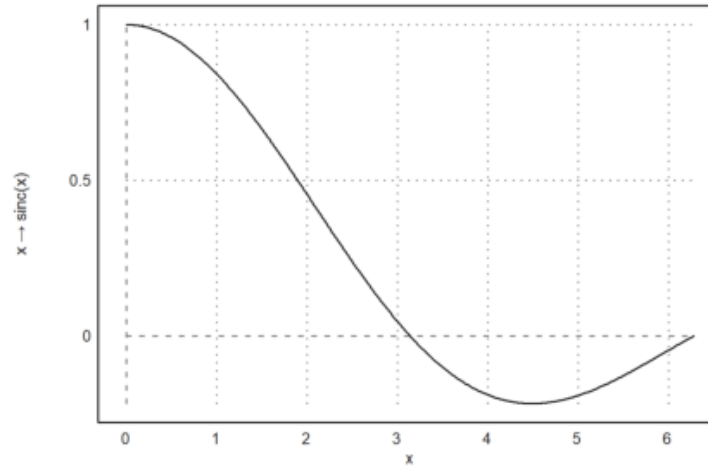
Label teks, judul, kotak label, dan teks lain dapat berisi string Unicode (lihatlah sintaks EMT tentang string Unicode lebih lanjut)

```
>plot2d("x^3-x",title=u"x \rarr; x3 - x"):
```



Label pada sumbu-x dan sumbu-y dapat berbentuk vertikal, sesuai 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 menginstall sitem LaTeX. Saya menyarankan MikTeX. Path ke binary "latex" dan "dvipng" harus ada ddalam path sistem, atau Anda harus mengatur LaTeX pada menu Option.

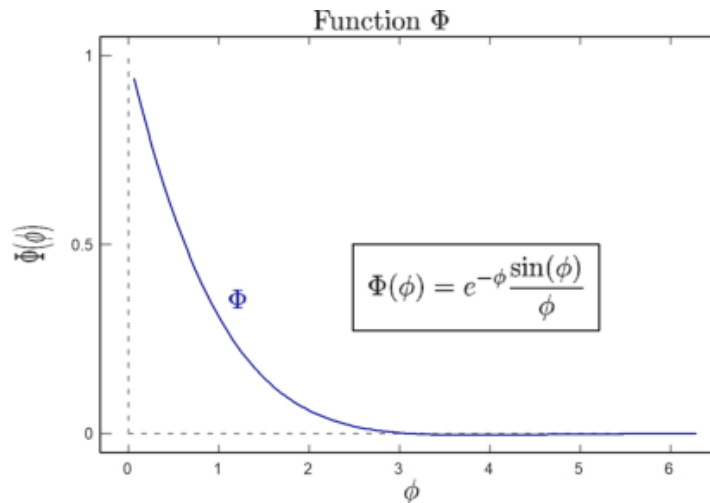
Perhatikan bahwa penguraian LaTeX itu lambat. Jika Anda ingin menggunakan LaTeX pada plot animasi, Anda harus memanggil `latex()` sebelum loop sekali dan menggunakan hasilnya (gambar dalam matriks RGB).

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

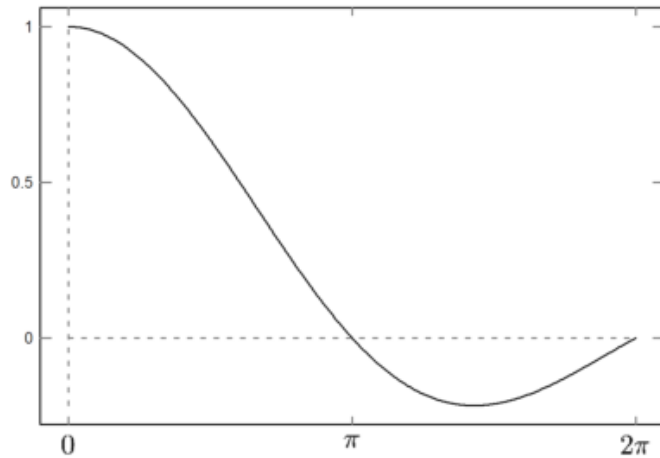
```



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

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

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



Tentunya, beberapa fungsi juga dapat digunakan.

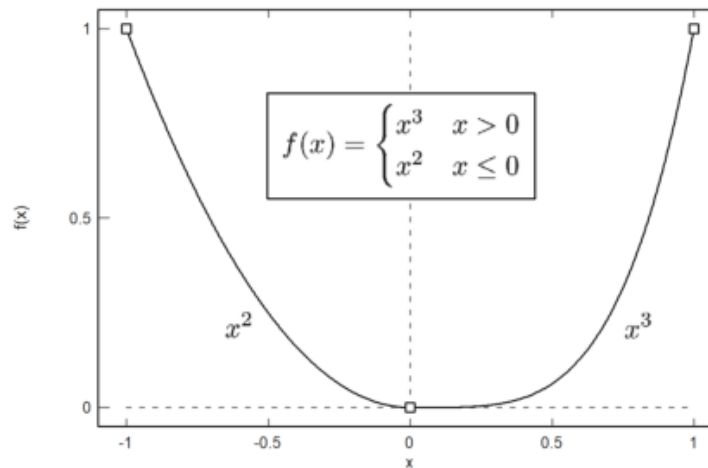
```
>function map f(x) ...
```

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

Parameter "map" membantu untuk menggunakan fungsi untuk vektor. Untuk plot, parameter tersebut tidak diperlukan. Tetapi berguna untuk mendemonstrasikan vektorisasi, kita menambahkan beberapa titik kunci untuk plot pada $x=-1$, $x=0$, dan $x=1$.

Pada plot berikut, kita juga memasukkan beberapa kode LaTeX. Kita menggunakannya untuk dua label dan sebuah kotak teks. Tentunya Anda hanya bisa menggunakan LaTeX jika Anda telah menginstall LaTeX dengan tepat.

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



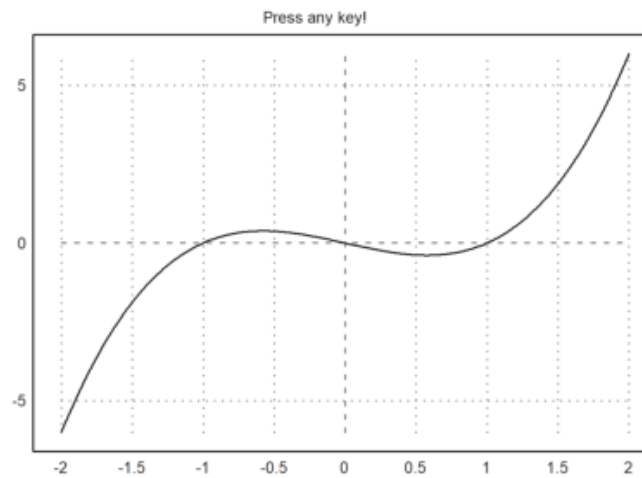
Ketika memplot suatu fungsi atau ekspresi, parameter `>user` mengizinkan pengguna untuk memperbesar dan menggeser plot dengan tombol kursor atau mouse. Pengguna dapat:

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

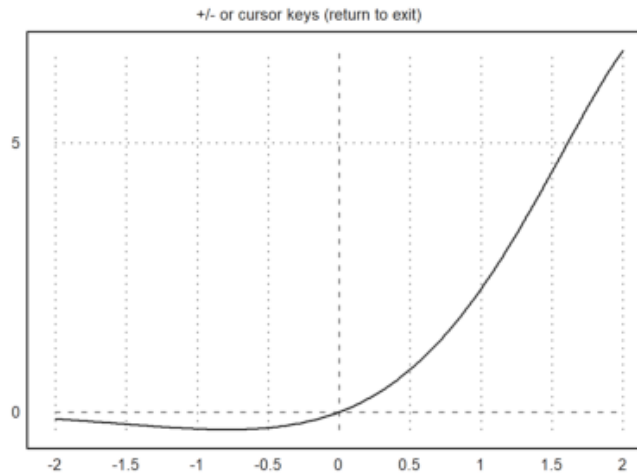
Tombol spasi akan mengatur ulang plot ke jendela plot awal.

Saat memplot data, flag `>user` akan menunggu hingga pengguna menekan tombol.

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



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

Demonstrasi berikut merupakan cara interaksi pengguna yang lebih canggih (lihatlah tutorial tentang pemrograman untuk detailnya).

Fungsi bawaan `mousedrag()` menunggu kejadian mouse atau keyboard. Fungsi tersebut melaporkan aksi mouse seperti tekan, gerakan mouse, atau lepas, dan penekanan tombol. Fungsi `dragpoints()` memanfaatkan ini, dan memungkinkan pengguna untuk menggeser titik manapun pada plot.

Kita perlu fungsi plot terlebih dahulu. Sebagai contoh, Kita melakukan interpolasi pada 5 titik dengan sebuah polinomial. Fungsinya harus memplot ke dalam area plot yang tetap.

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

```
    d=interp(xp,yp);
    plot2d("interpval(xp,d,x)";d,xp,r=2);
    plot2d(xp,yp,>points,>add);
    if select>0 then
```

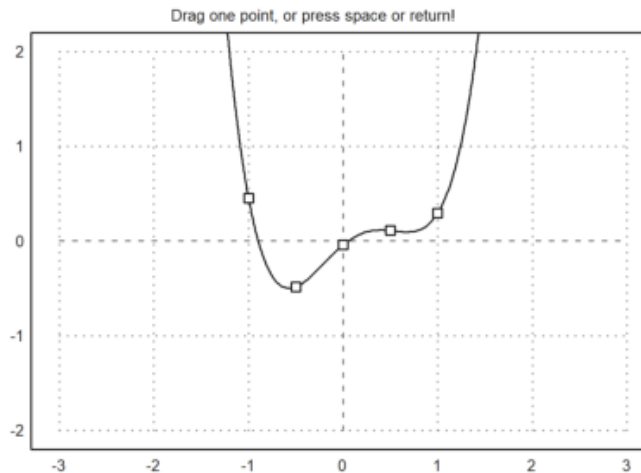
```

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

```

Perhatikan parameter titik koma dalam plot2d (d dan xp), yang diteruskan untuk evaluasi fungsi interp(). Tanpa ini, kita harus menulis fungsi plotinterp() terlebih dahulu, mengakses nilai-nilai secara global. Sekarang kita akan menghasilkan nilai acak, dan membiarkan pengguna menyeret titik-titik tersebut.

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



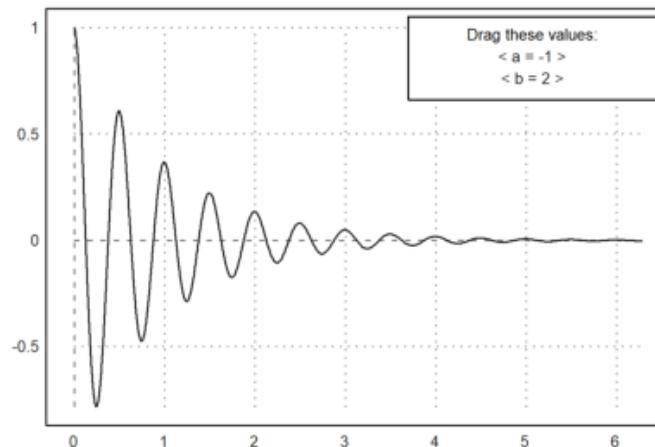
Ada juga sebuah fungsi yang memplot fungsi lain tergantung pada vektor parameter, dan memungkinkan pengguna untuk mengatur parameter tersebut.

Pertama, kita perlu fungsi plotnya.

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

Selanjutnya, kita perlu nama untuk parameter-parameter, nilai awal, matriks nx2 untuk rentang nilai, serta opsi untuk menambahkan judul. Terdapat slider interaktif yang memungkinkan pengguna mengatur nilai. Fungsi `dragvalues()` menyediakan fitur ini.

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



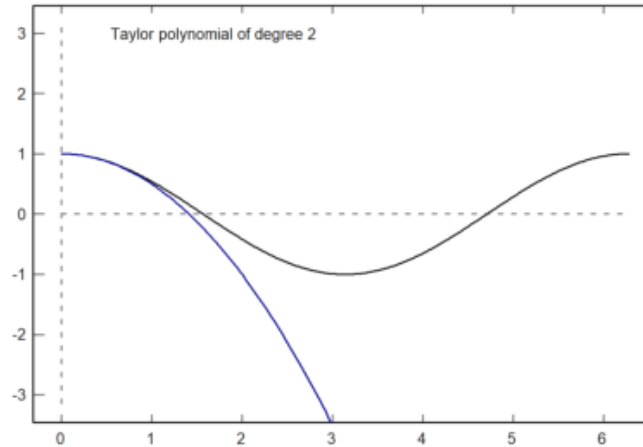
Kita bisa membatasi nilai yang diseret hanya pada bilangan bulat. Sebagai contoh, kita akan menulis fungsi plot yang memplot polinomial Taylor derajat n untuk fungsi kosinus.

```
>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 kita mengizinkan derajat n bervariasi dari 0 hingga 20 dengan 20 langkah. Hasil dari `drag-values()` digunakan untuk memplot sketsa dengan n tersebut dan memasukkan plot ke dalam notebook.

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



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

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

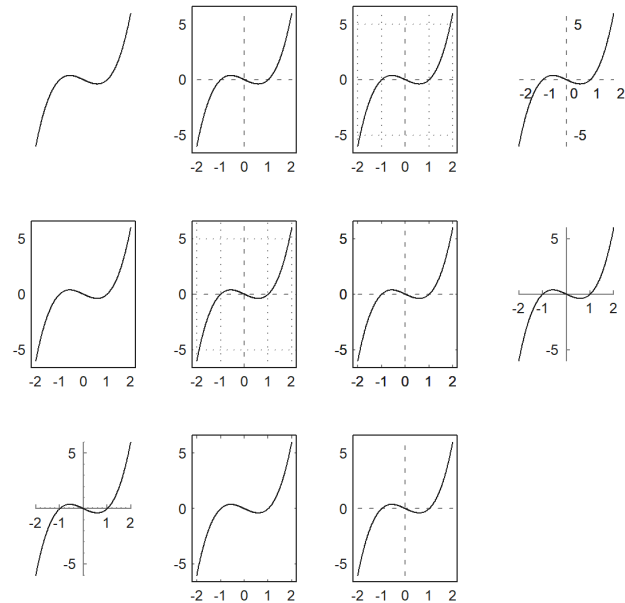
```
    plot2d(none,r=1,title="Drag with the mouse, or press any key!");
    start=0;
    repeat
        {flag,m,time}=mousedrag();
        if flag==0 then return; endif;
        if flag==2 then
            hold on; mark(m[1],m[2]); hold off;
        endif;
    end
endfunction
```

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

Gaya Plot 2d

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

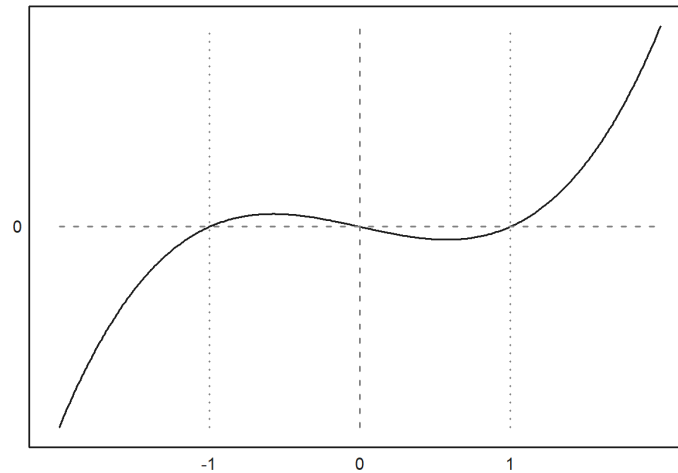
```
>aspect();
>figure(3,4); ...
> figure(1); plot2d("x^3-x",grid=0); ... // tanpa grid, bingkai, atau sumbu
> figure(2); plot2d("x^3-x",grid=1); ... // sumbu x-y
> figure(3); plot2d("x^3-x",grid=2); ... // tanda sumbu default
> figure(4); plot2d("x^3-x",grid=3); ... // sumbu x-y dengan label di dalamnya
> figure(5); plot2d("x^3-x",grid=4); ... // tanpa tanda sumbu, hanya label
> figure(6); plot2d("x^3-x",grid=5); ... // default, tanpa margin
> figure(7); plot2d("x^3-x",grid=6); ... // sumbu saja
> figure(8); plot2d("x^3-x",grid=7); ... // sumbu saja, tanda sumbu pada sumbu
> figure(9); plot2d("x^3-x",grid=8); ... // sumbu saja, tanda sumbu lebih halus sepanjang sumbu
> figure(10); plot2d("x^3-x",grid=9); ... // default,tanda sumbu kecil di dalamnya
> figure(11); plot2d("x^3-x",grid=10); ...// tanpa tanda sumbu, sumbu saja
> figure(0):
```



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

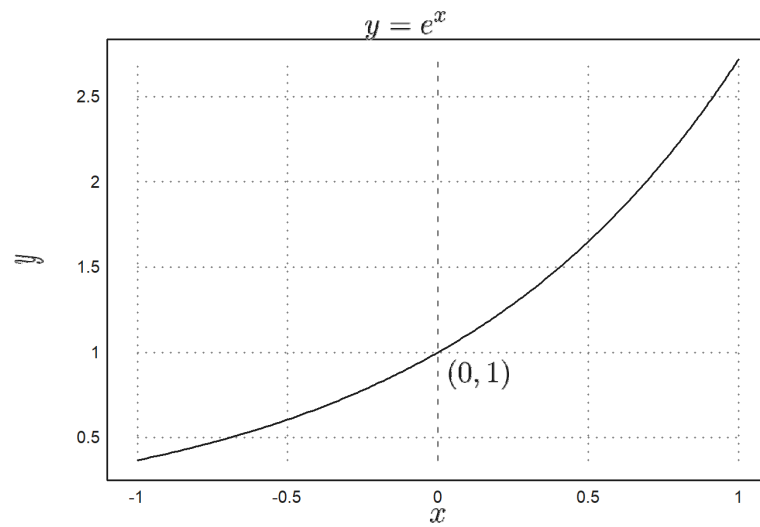
Jika Anda ingin tanda sumbu sendiri, Anda bisa menggunakan `style=0`, dan menambahkan semuanya nanti.

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



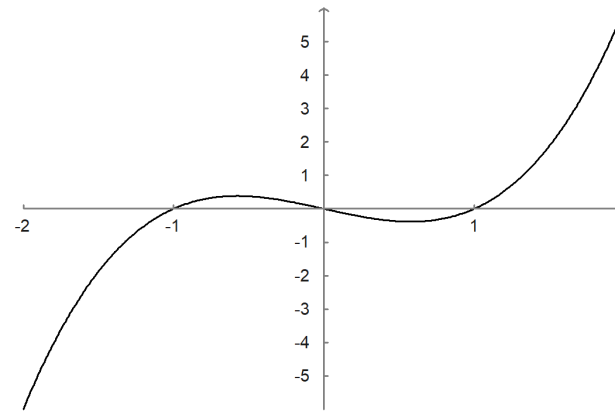
Untuk judul plot dan label sumbu, lihat contoh berikut

```
>plot2d("exp(x)",-1,1);  
>textcolor(black); // atur warna text menjadi hitam  
>title(latex("y=e^x")); // judul di atas plot  
>xlabel(latex("x")); // "x" untuk sumbu-x  
>ylabel(latex("y"),>vertical); // "y" vertikal untuk sumbu-y  
>label(latex("(0,1)"),0,1,color=blue): // label sebuah titik
```

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

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

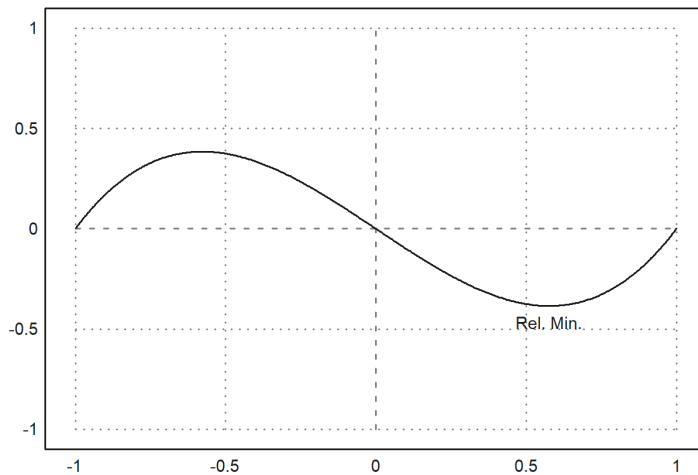


Teks pada plot dapat diatur dengan `label()`. Pada contoh berikut, "lc" berarti lower center. Ia mengatur posisi label relatif terhadap koordinat plot.

```
>function f(x) {x^3-x}
```

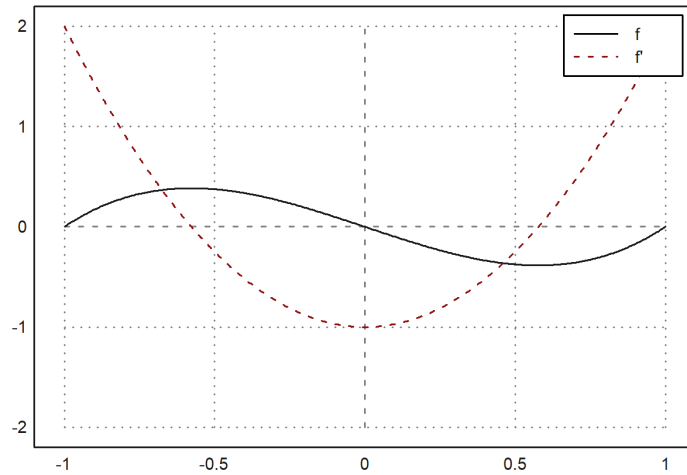
$$x^3 - x$$

```
>plot2d(f,-1,1,>square);
>x0=fmin(f,0,1); // hitung titik minimum
>label("Rel. Min.",x0,f(x0),pos="lc"): // menambah label disana
```

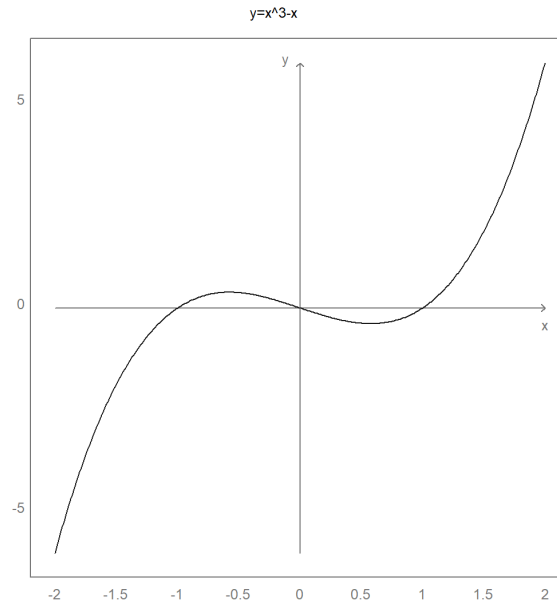


Ada juga kotak teks.

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



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



Untuk kontrol yang lebih besar, sumbu x dan sumbu y dapat diatur secara manual.

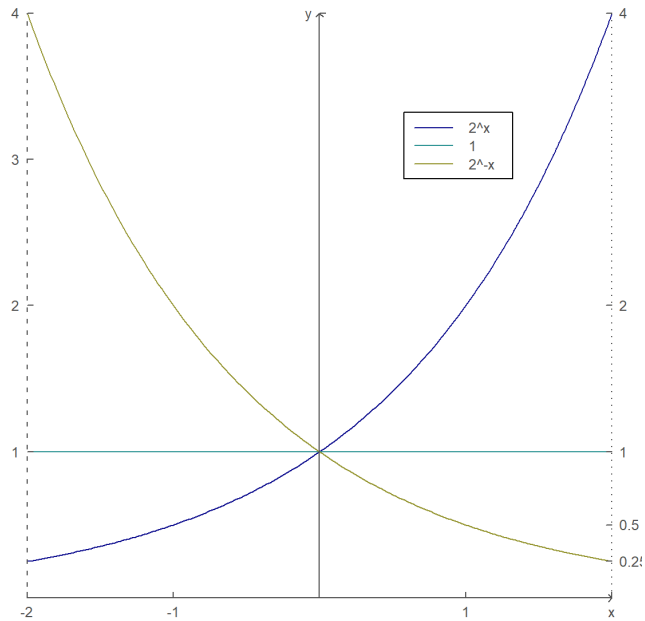
Perintah `fullwindow()` memperluas jendela plot karena kita tidak lagi memerlukan ruang untuk label di luar jendela plot. Gunakan `shrinkwindow()` atau `reset()` untuk mengembalikan ke pengaturan default.

```
>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:

```

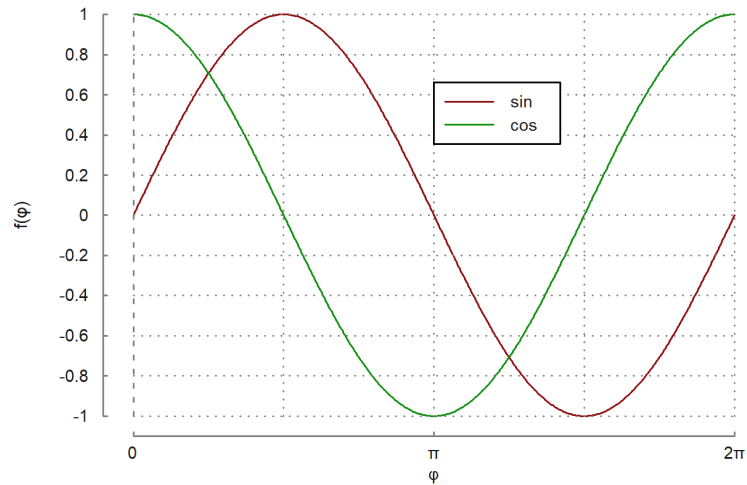


Berikut ini contoh lain, dimana stringUnicode digunakan dan sumbu-sumbu berada diluar area plot.

```

>aspect(1.5);
>plot2d(["sin(x)","cos(x)"],0,2pi,color=[red,green],<grid,<frame); ...
> axis(-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 hal ini, a , b , c , dan d , atau jari-jari r dapat ditentukan, atau jendela plot akan disesuaikan secara otomatis dengan data. Sebagai alternatif, Anda bisa mengatur `>square` untuk menjaga rasio aspek persegi.

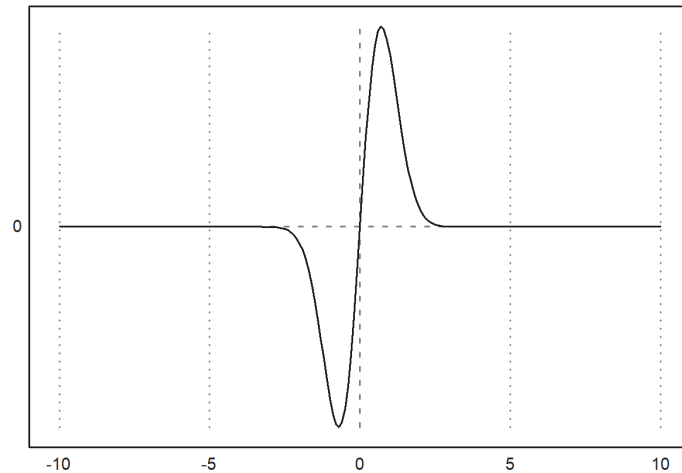
Memplot ekspresi hanya merupakan singkatan untuk plot data. Untuk plot data, Anda memerlukan satu atau lebih baris nilai x dan satu atau lebih baris nilai y . Dari rentang dan nilai x , fungsi `plot2d` akan menghitung data yang akan dipetakan, secara default dengan evaluasi adaptif dari fungsi tersebut.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

Namun, Anda dapat memasukkan data secara langsung.

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

Berikut adalah contoh dengan satu baris x dan y .

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

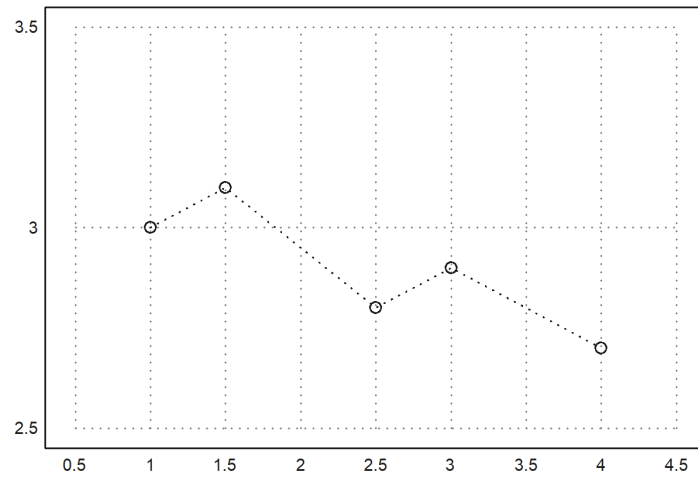
Data juga dapat dipetakan sebagai titik. Gunakan `points=true` untuk ini. Plot berfungsi seperti poligon, tetapi hanya menggambar sudut-sudutnya.

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

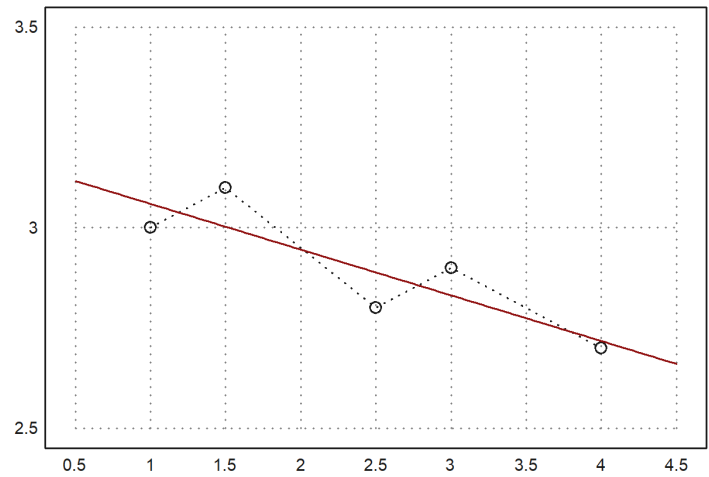
Untuk memplot sekumpulan titik, gunakan `>points`. Jika warna adalah vektor warna, setiap titik akan mendapatkan warna yang berbeda. Untuk matriks koordinat dan vektor kolom, warna akan diterapkan pada baris-baris matriks.

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

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



```
>p=polyfit(xdata,ydata,1); // dapatkan garis regresi  
>plot2d("polyval(p,x)",>add,color=red): // tambahkan plot garis
```



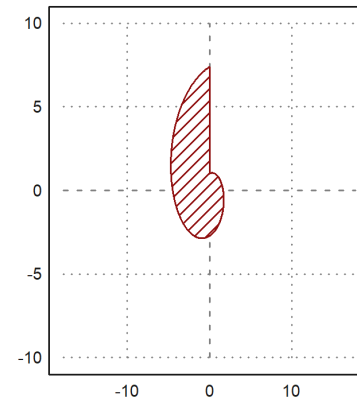
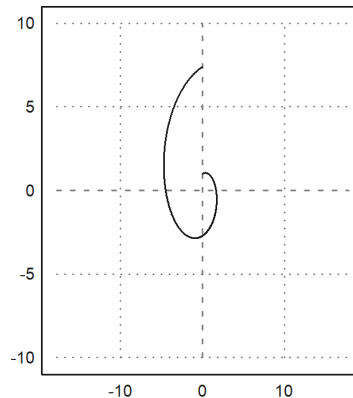
Menggambar Daerah Yang Dibatasi Kurva

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

- filled=true akan memngisi plot
- style="...": pilih dari "", "/", "\", "\/".
- fillcolor: Lihat diatas untuk warna yang tersedia

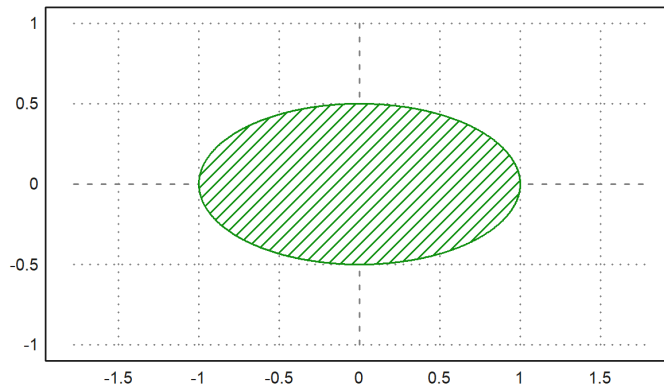
Warna pengisi ditentukan oleh argumen fillcolor, dan opsi <outline> mencegah penggambaran batas untuk semua gaya kecuali gaya default.

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

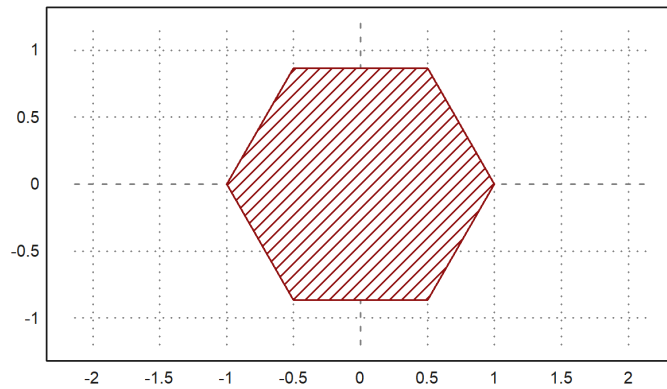


Dalam contoh berikut, kita memplot sebuah elips terisi dan dua heksagon terisi menggunakan kurva tertutup dengan 6 titik dengan gaya pengisian yang berbeda

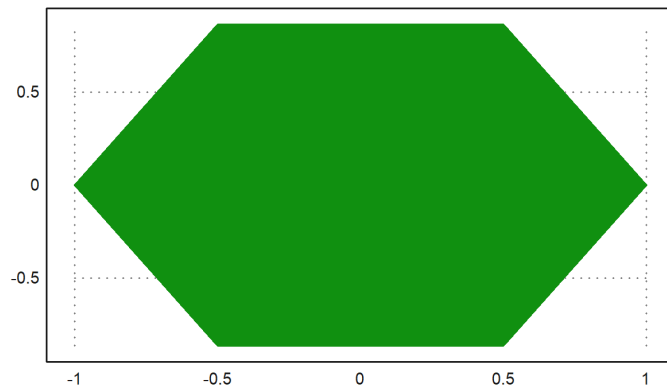
```
>x=linspace(0,2pi,1000); plot2d(sin(x),cos(x)*0.5,r=1,>filled,style="/"):
```



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

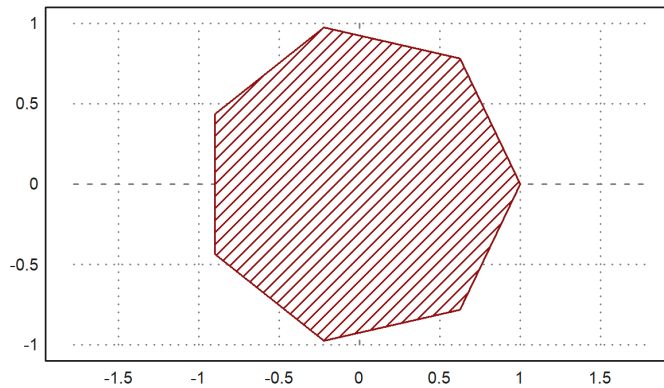


```
>t=linspace(0,2pi,6); plot2d(cos(t),sin(t),>filled,style="#"):
```



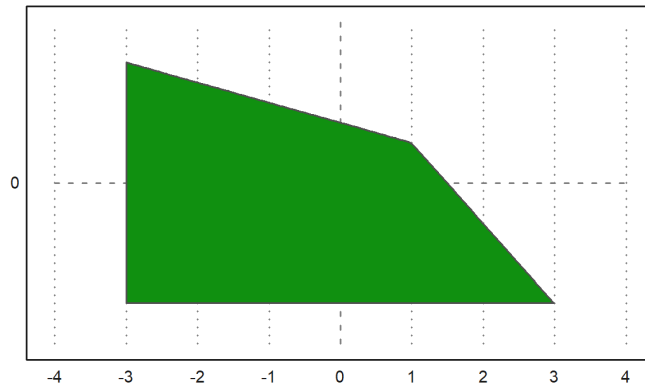
Contoh lainnya adalah septagon, yang kita buat dengan 7 titik pada lingkaran satuan.

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



Berikut adalah himpunan nilai maksimum dari empat kondisi linier yang kurang dari atau sama dengan 3. Ini adalah $[k].v \leq 3$ untuk semua baris dari A. Untuk mendapatkan sudut yang baik, kita menggunakan n yang relatif besar.

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

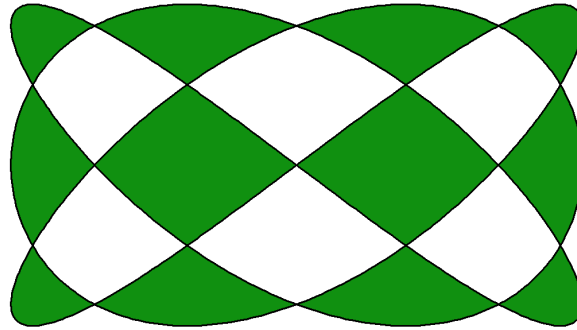


Inti dari bahasa matriks adalah kemampuannya untuk dengan mudah menghasilkan tabel fungsi.

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

Sekarang kita punya vektor x dan y dari nilai-nilai. `plot2d()` dapat memplot nilai-nilai ini sebagai kurva tang menghubungkan titik-titik. Plot dapat diisi. Pada kasus ini, hasilnya terlihat bagus berkat aturan winding yang digunakan untuk pengisian

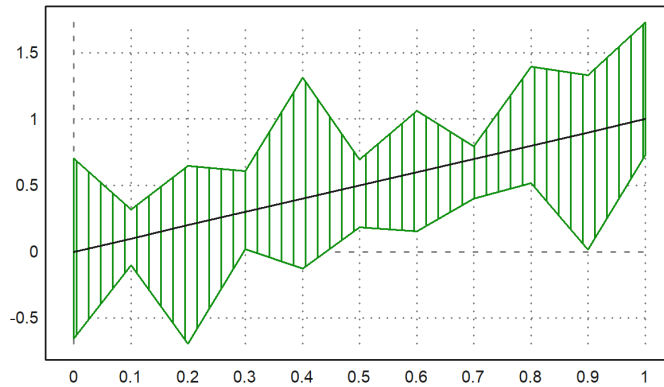
```
>plot2d(x,y,<grid,<frame,>filled):
```

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

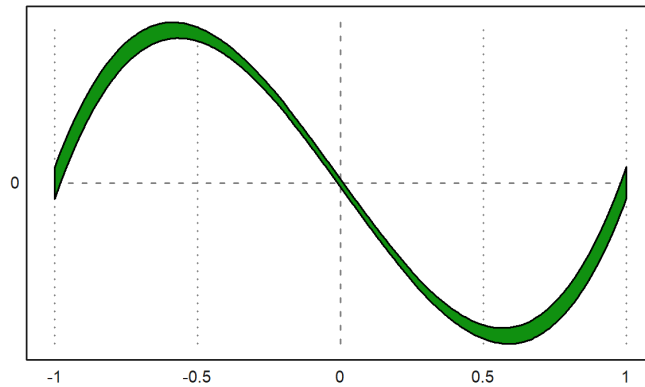
Ini bisa berguna untuk memplot kesalahan dalam perhitungan. Namun, ini juga dapat digunakan untuk memplot kesalahan statistik.

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



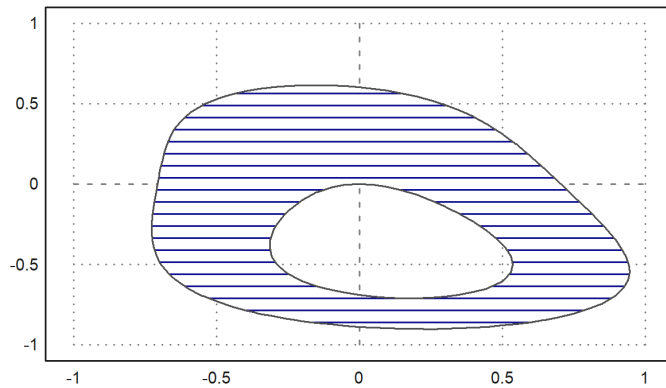
ika x adalah vektor yang terurut dan y adalah vektor interval, maka `plot2d` akan memplot rentang terisi dari interval tersebut di bidang. Gaya pengisian yang digunakan sama dengan gaya poligon.

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



Dimungkinkan untuk mengisi daerah nilai untuk fungsi tertentu. Untuk ini, level harus berupa matriks $2 \times n$. Baris pertama adalah batas bawah, dan baris kedua berisi batas atas.

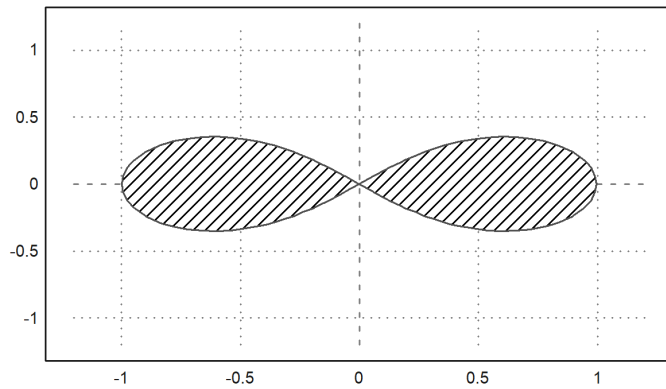
```
>expr := "2*x^2+x*y+3*y^4+y"; // mendefinisikan ekspresi f(x,y)
>plot2d(expr,level=[0;1],style="-",color=blue): // 0 <= f(x,y) <= 1
```



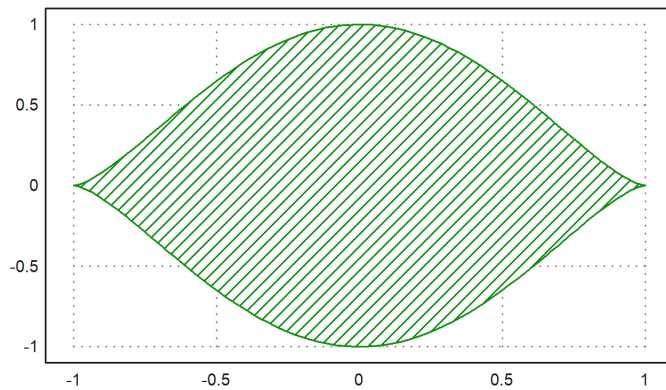
Kita juga dapat mengisi rentang nilai seperti

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

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



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



Grafik Fungsi Parametrik

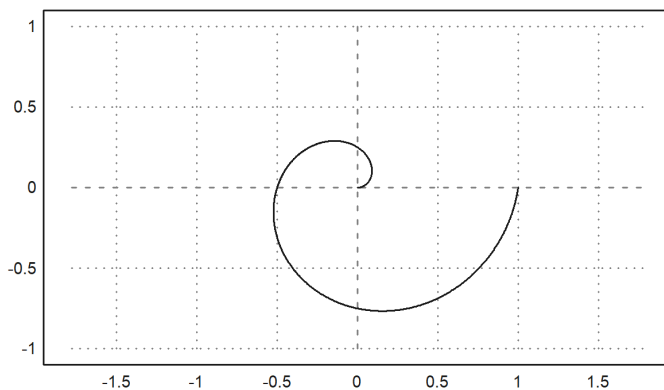
Nilai x tidak perlu diurutkan. (x,y) hanya menggambarkan sebuah kurva. Jika terurut, kurva tersebut menjadi grafik dari sebuah fungsi.

Pada contoh berikut, kita memplot spiral

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

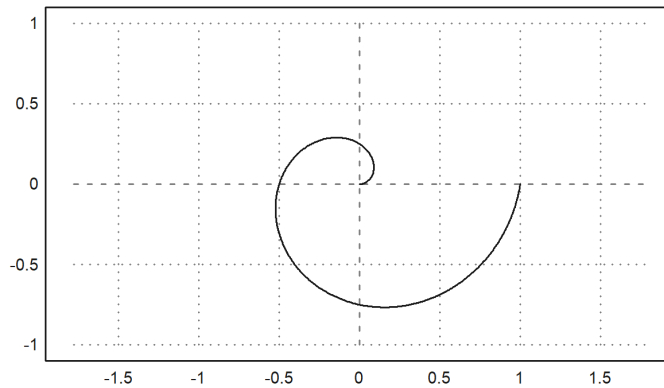
Kita perlu menggunakan banyak titik untuk tampilan yang halus atau menggunakan fungsi `adaptive()` untuk mengevaluasi ekspresi. (Lihat fungsi `adaptive()` untuk lebih lanjut)

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

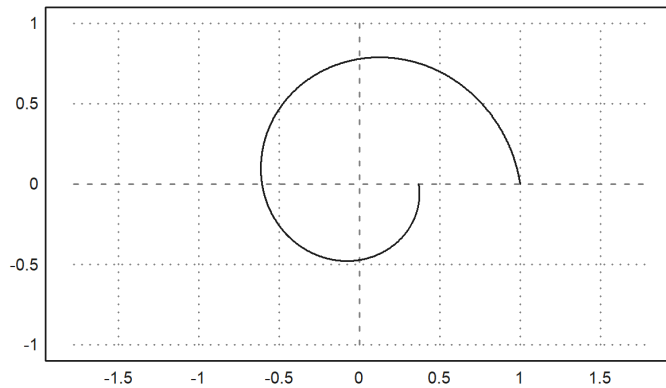


Sebagai alternatif, Anda dapat menggunakan dua ekspresi untuk kurva. Berikut ini memplot kurva yang sama seperti di atas.

```
>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):
```



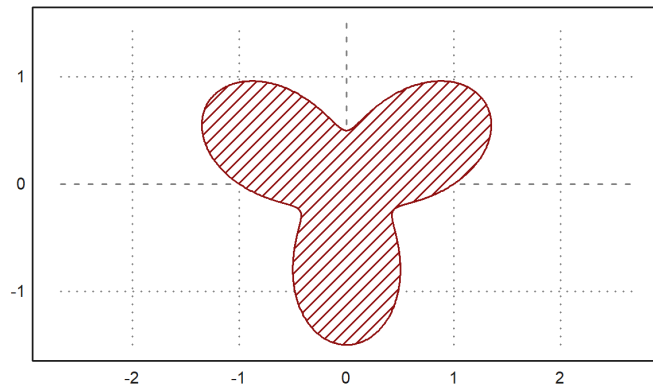
Dalam contoh berikut, kita memplot kurva

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

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):
```

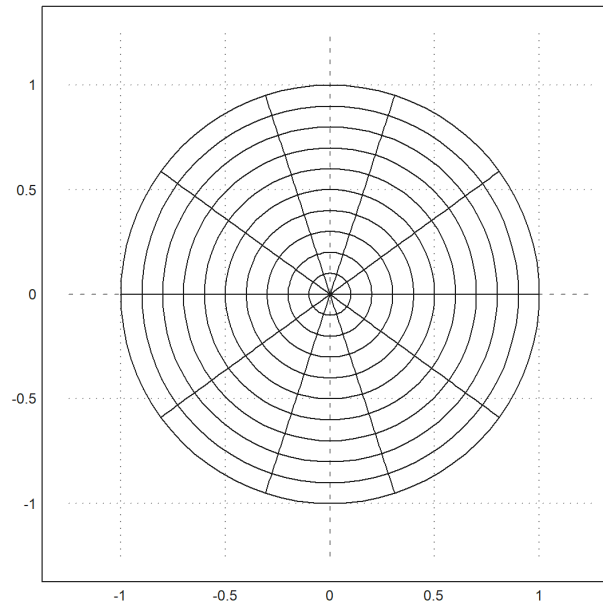
Menggambar Grafik Bilangan Kompleks

Sekumpulan bilangan kompleks juga dapat dipetakan. Dalam hal ini, titik-titik grid akan dihubungkan. Jika sejumlah garis grid ditentukan (atau vektor 1x2 dari garis grid) dalam argumen `cgrid`, hanya garis grid tersebut yang akan terlihat.

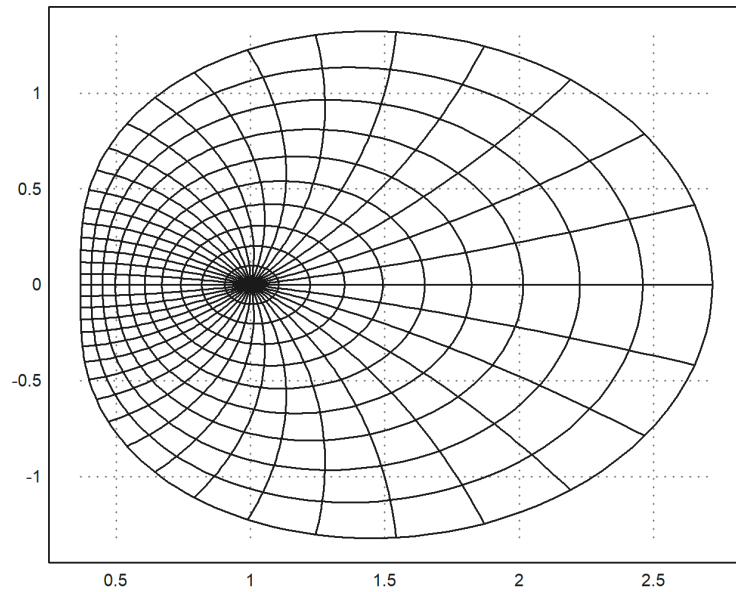
Matriks bilangan kompleks secara otomatis akan dipetakan sebagai grid di bidang kompleks.

Dalam contoh berikut, kita memplot citra dari lingkaran satuan di bawah fungsi eksponensial. Parameter `cgrid` menyembunyikan beberapa kurva grid.

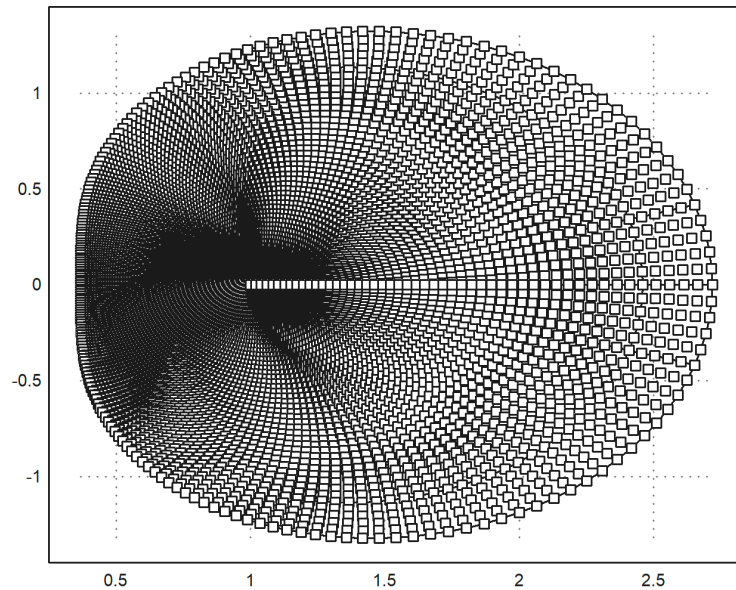
```
>aspect(); r=linspace(0,1,50); a=linspace(0,2pi,80)'; z=r*exp(I*a);...  
>plot2d(z,a=-1.25,b=1.25,c=-1.25,d=1.25,cgrid=10):
```



```
>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):
```

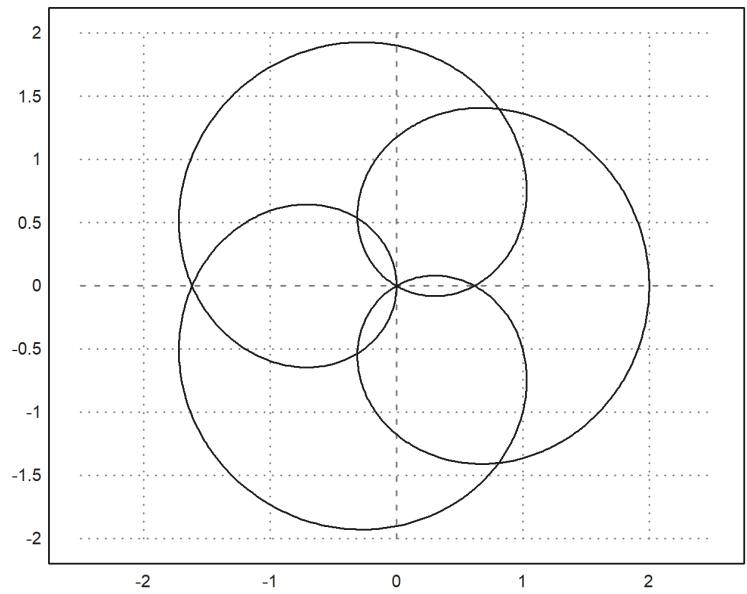


Vektor bilangan kompleks secara otomatis dipetakan sebagai kurva di bidang kompleks dengan bagian nyata dan bagian imajiner.

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 khusus untuk plot statistik. Salah satu plot yang sering digunakan adalah plot kolom.

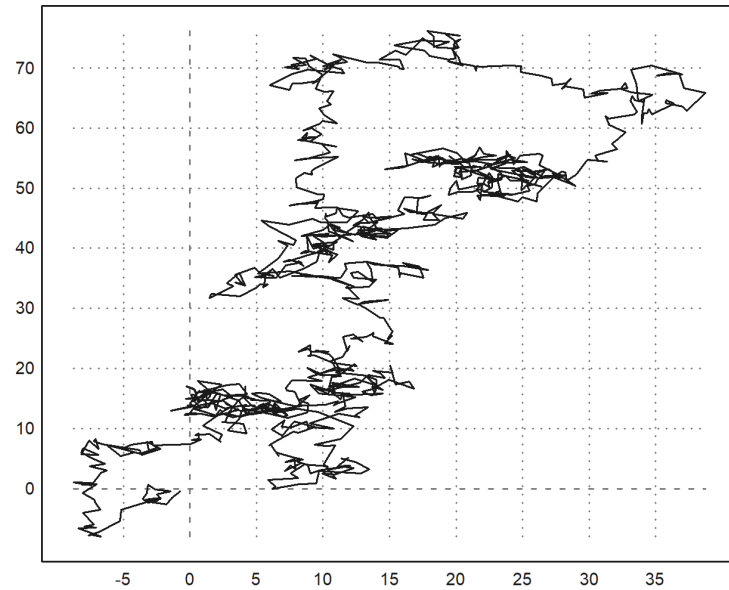
Jumlah kumulatif dari nilai-nilai yang terdistribusi normal 0-1 menghasilkan sebuah random walk

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

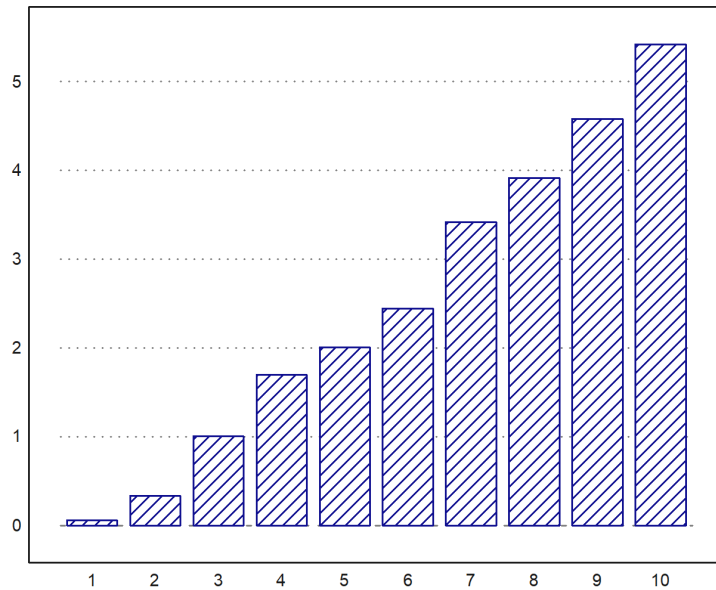


Menggunakan dua baris menunjukkan sebuah langkah dalam dua dimensi.

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

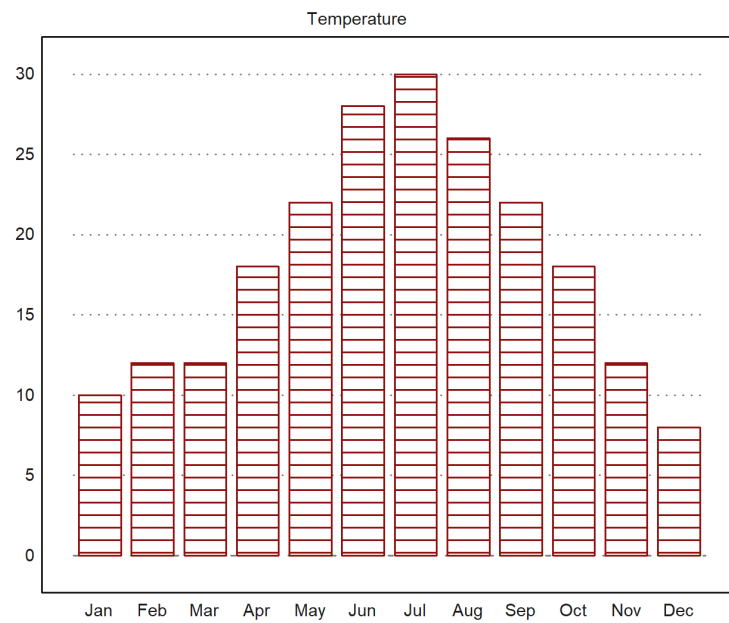


```
>columnplot(cumsum(random(10)),style="/",color=blue):
```

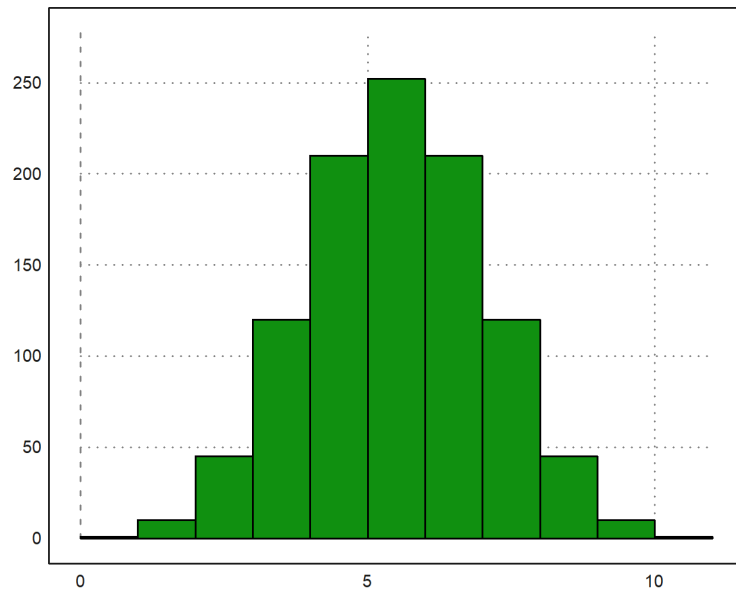



Ini juga dapat menampilkan string sebagai label.

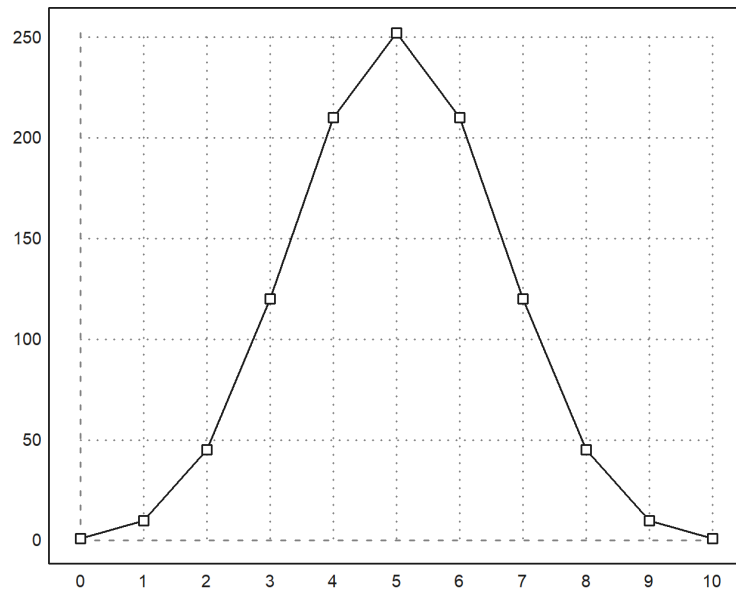
```
>months=["Jan","Feb","Mar","Apr","May","Jun", ...  
> "Jul","Aug","Sep","Oct","Nov","Dec"];  
>values=[10,12,12,18,22,28,30,26,22,18,12,8];  
>columnsplot(values,lab=months,color=red,style="-");  
>title("Temperature"):
```



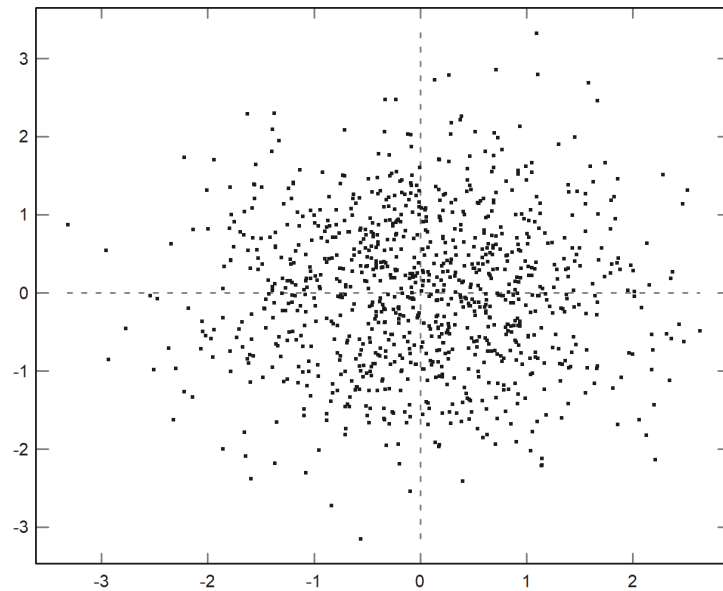
```
>k=0:10;  
>plot2d(k,bin(10,k),>bar):
```



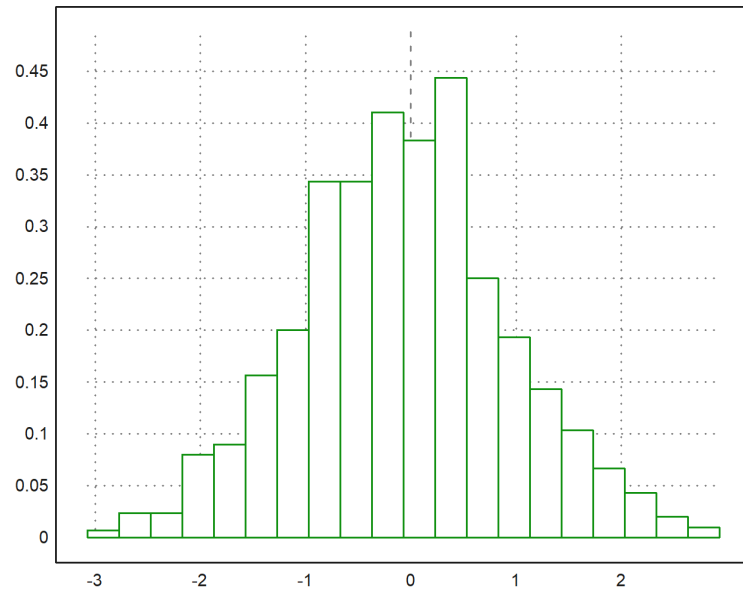
```
>plot2d(k,bin(10,k)); plot2d(k,bin(10,k),>points,>add):
```



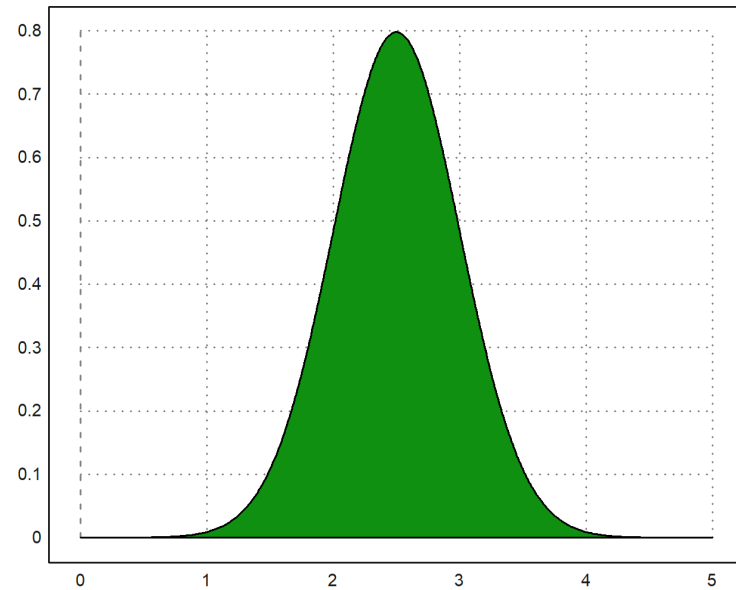
```
>plot2d(normal(1000),normal(1000),>points,grid=6,style=".."):
```



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

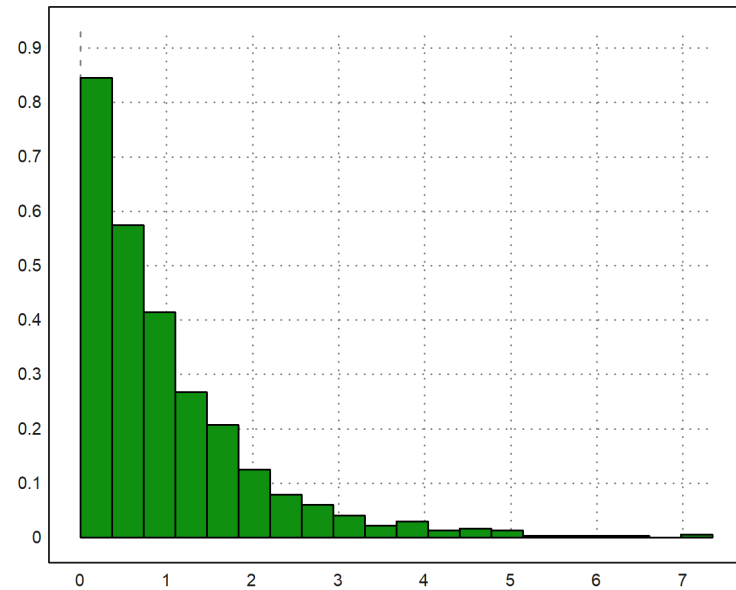


```
>plot2d("qnormal",0,5;2.5,0.5,>filled):
```



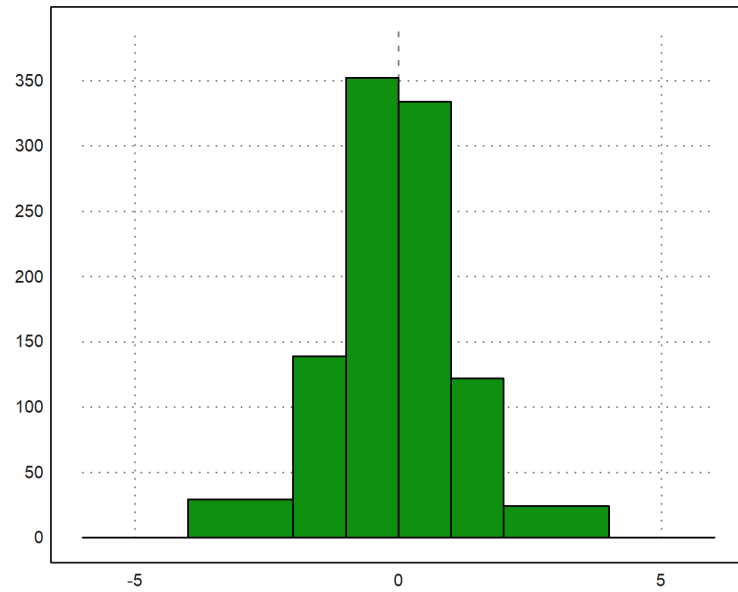
Untuk memplot distribusi statistik eksperimental, Anda dapat menggunakan `distribution=n` dengan `plot2d`.

```
>w=randexponential(1,1000); // distribusi exponensial  
>plot2d(w,>distribution): // atau distribution=n dengan n interval
```



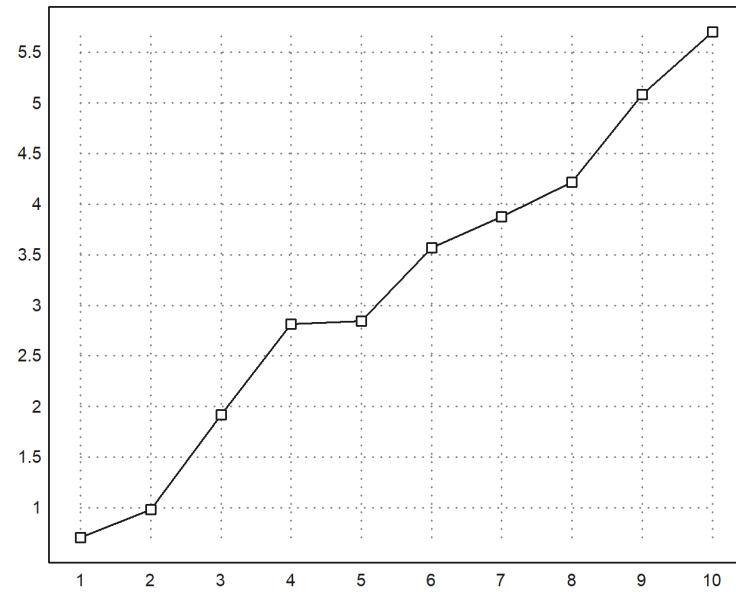
Atau, Anda bisa menghitung distribusi dari data dan memplot hasilnya dengan `>bar` dalam `plot3d`, atau menggunakan `plot kolom`.

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

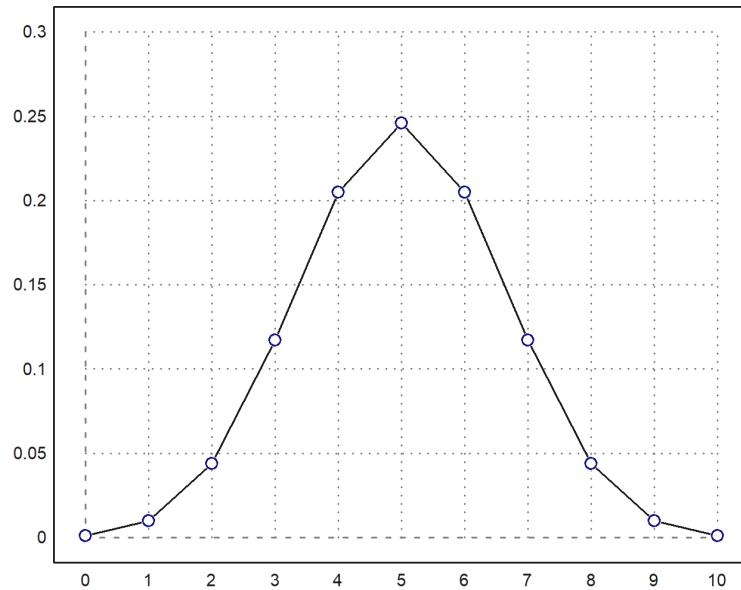



Fungsi `statplot()` mengatur gaya dengan menggunakan sebuah string sederhana.

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



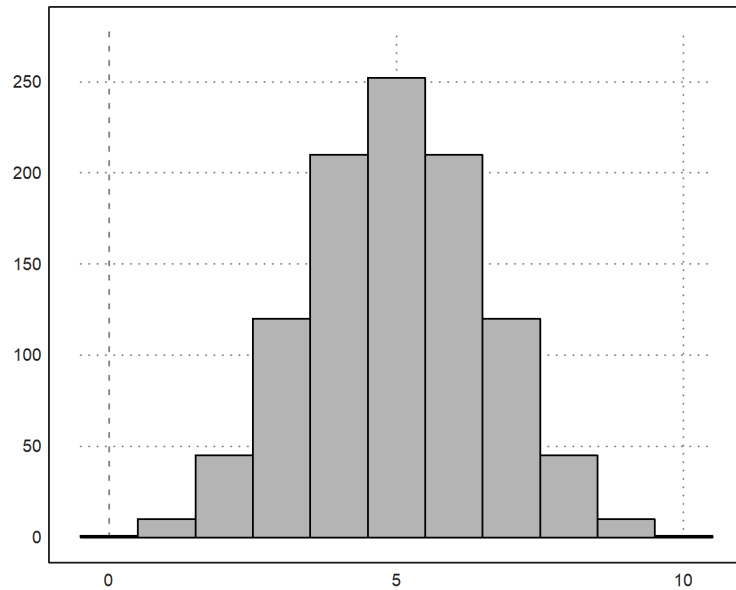
```
>n=10; i=0:n; ...  
>plot2d(i,bin(n,i)/2^n,a=0,b=10,c=0,d=0.3); ...  
>plot2d(i,bin(n,i)/2^n,points=true,style="ow",add=true,color=blue):
```



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

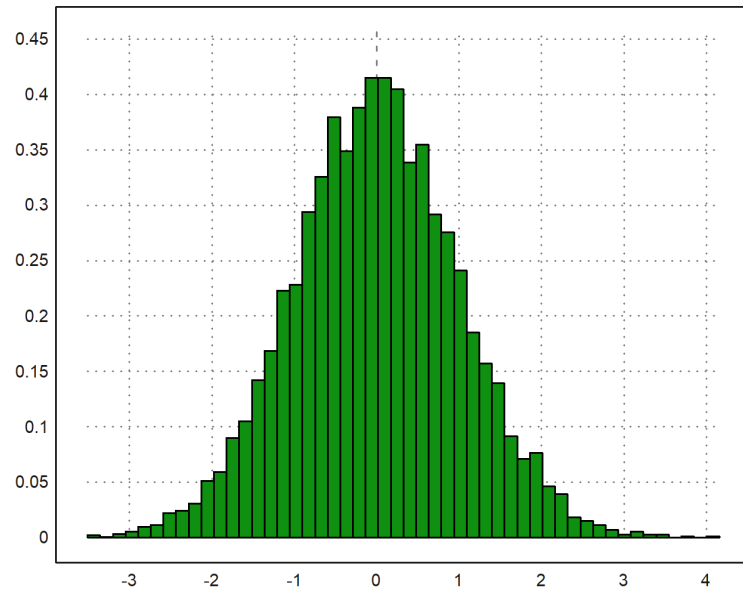
Gaya pengisian dapat digunakan sama seperti sebelumnya.

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

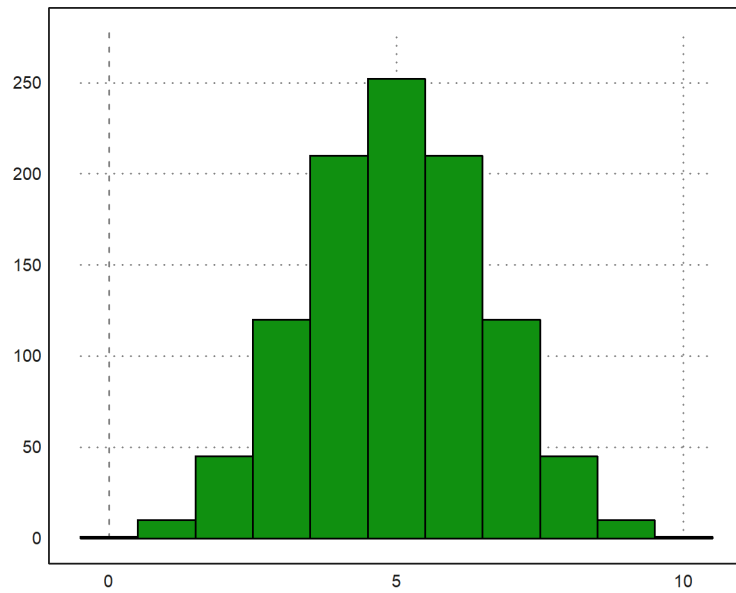


Data untuk plot batang (`bar=1`) and histogram (`histogram=1`) dapat diberikan secara eksplisit dalam `xv` dan `yv`, atau dapat dihitung dari distribusi empiris dalam `xv` dengan `>distribution` (atau `distribution=n`). Histogram dari nilai `xv` akan dihitung secara otomatis dengan `>histogram`. Jika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

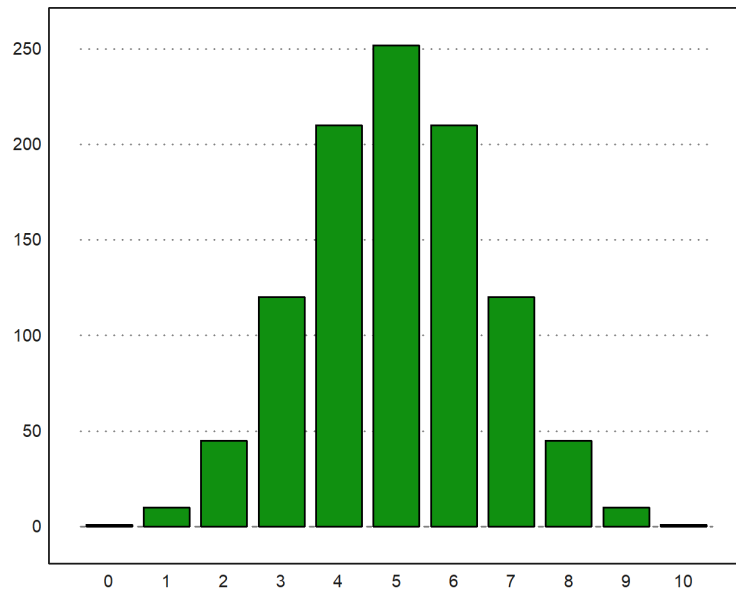
```
>plot2d(normal(10000),distribution=50):
```



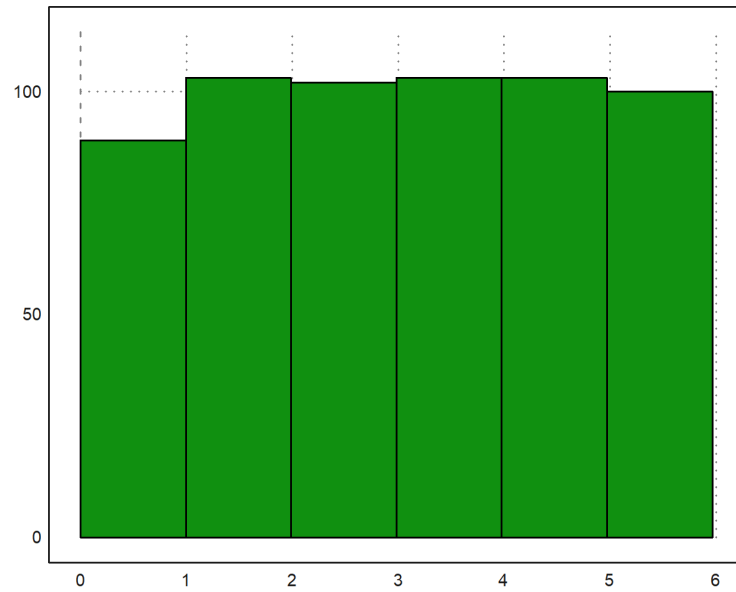
```
>k=0:10; m=bin(10,k); x=(0:11)-0.5; plot2d(x,m,>bar):
```



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

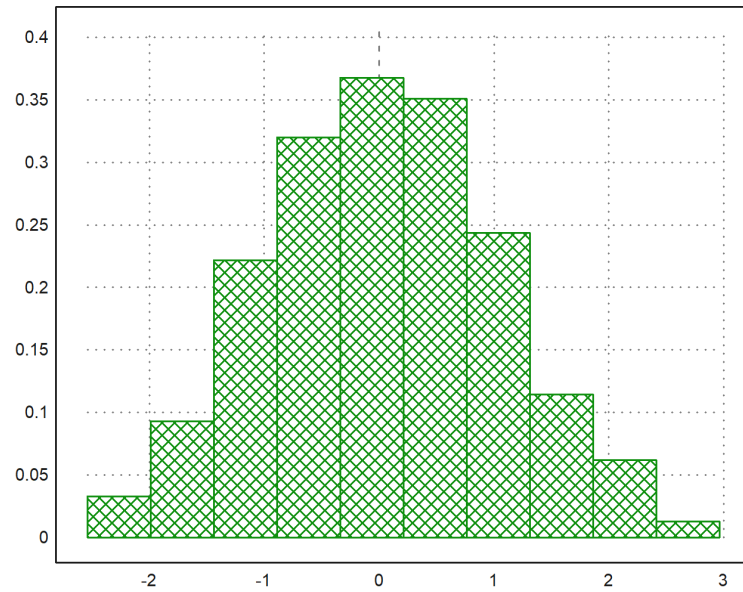


```
>plot2d(random(600)*6,histogram=6):
```



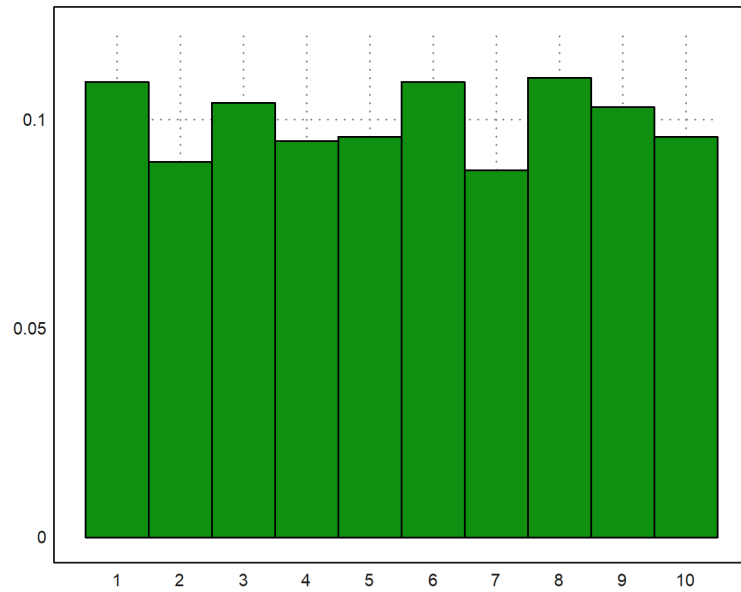
Untuk distribusi, ada parameter `distribution=n`, yang menghitung nilai secara otomatis dan mencetak distribusi relatif dengan `n` sub-interval.

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

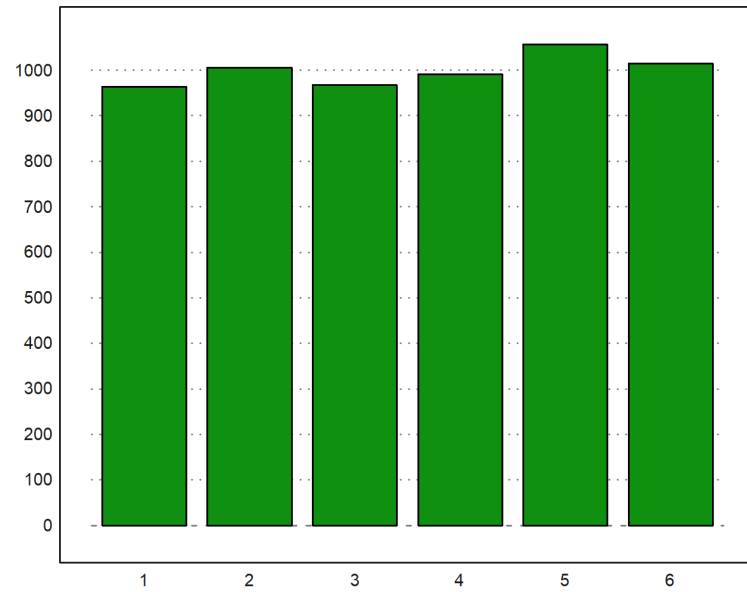
Dengan parameter `even=true`, Ini akan menggunakan interval bilangan bulat.

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

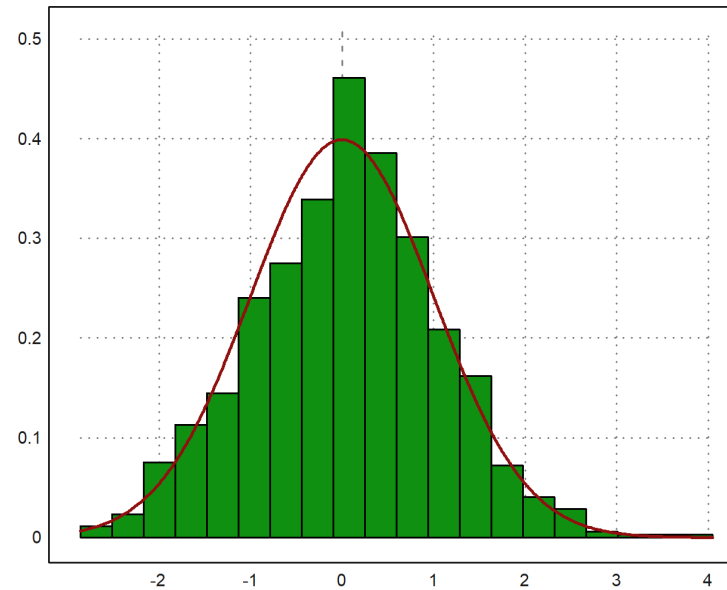


Perlu dicatat bahwa ada banyak plot statistik yang mungkin berguna. Silakan lihat tutorial tentang statistik.

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

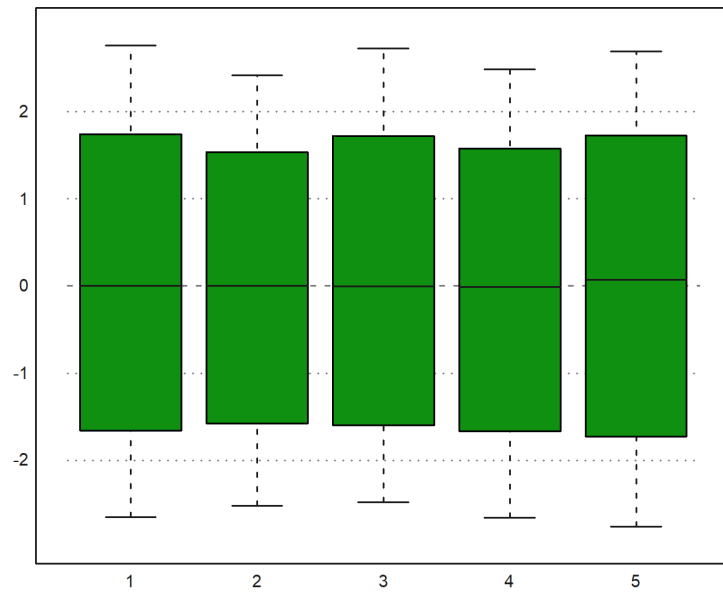


```
>plot2d(normal(1,1000),>distribution); ...  
> plot2d("qnormal(x)",color=red,thickness=2,>add):
```



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

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



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

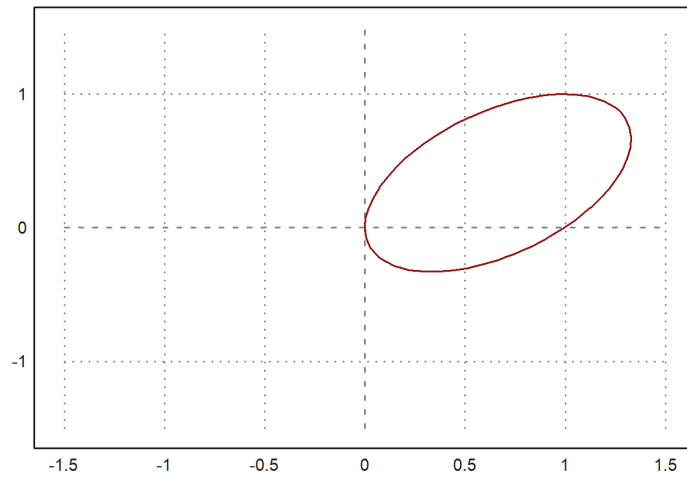
Euler dapat menandai garis level tersebut.

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

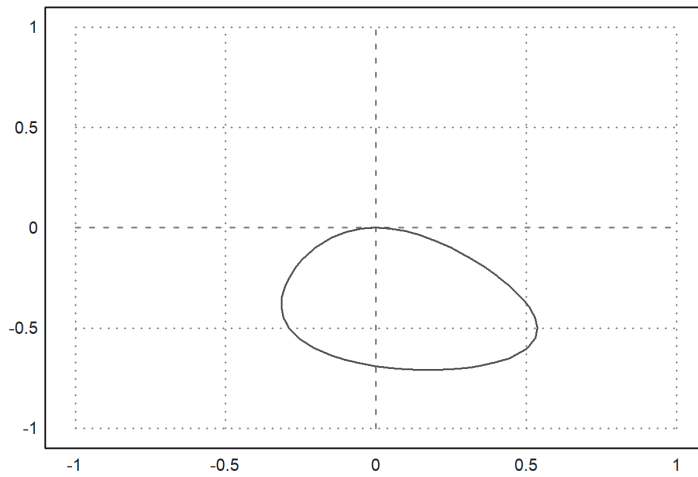
untuk fungsi apapun.

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

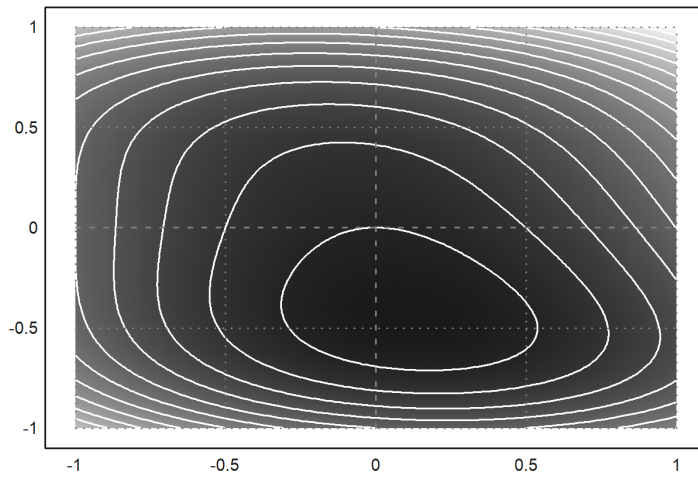
```
>aspect(1.5);  
>plot2d("x^2+y^2-x*y-x",r=1.5,level=0,contourcolor=red):
```



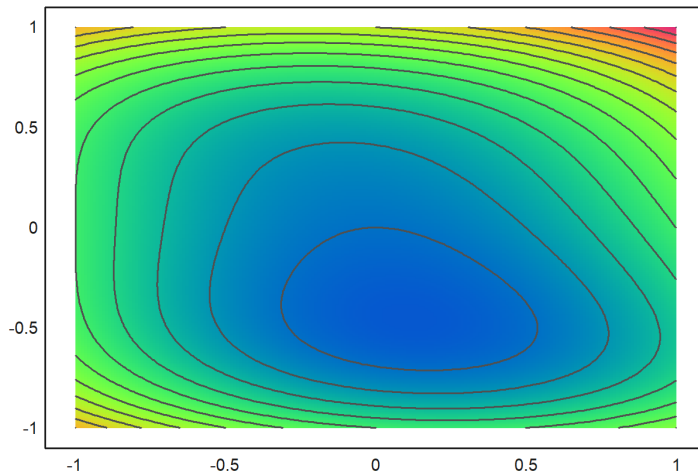
```
>expr := "2*x^2+x*y+3*y^4+y"; // Mendefinisikan suatu ekspresi f(x,y)
>plot2d(expr,level=0): // Solusi darif(x,y)=0
```



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

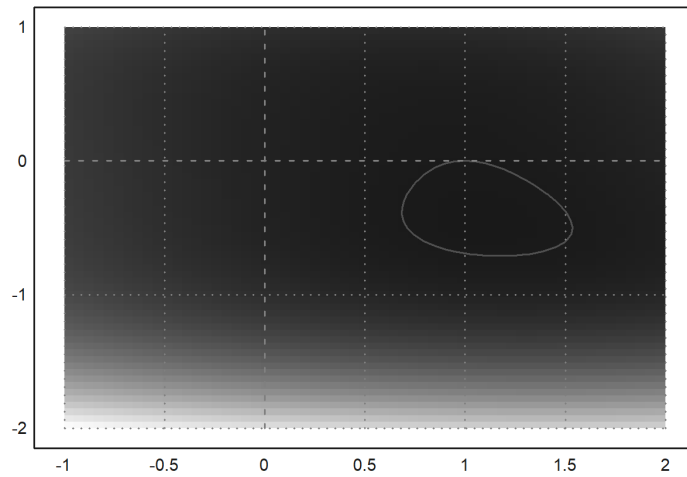



```
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=200,grid=4): // lebih bagus
```

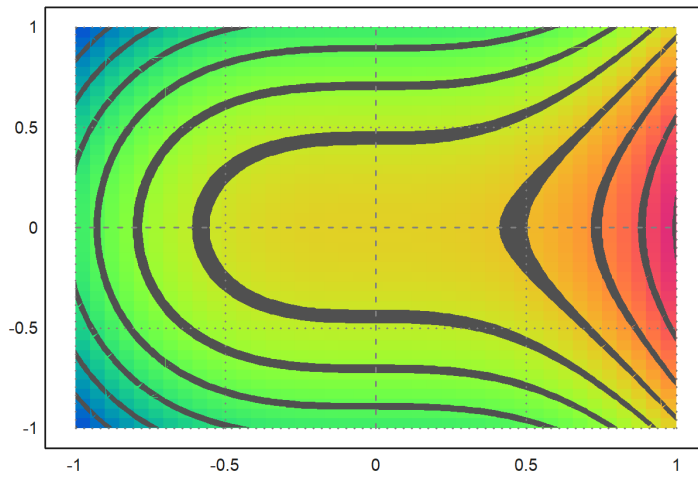


Ini juga berlaku untuk plot data. Namun, Anda harus menentukan rentang untuk label sumbu.

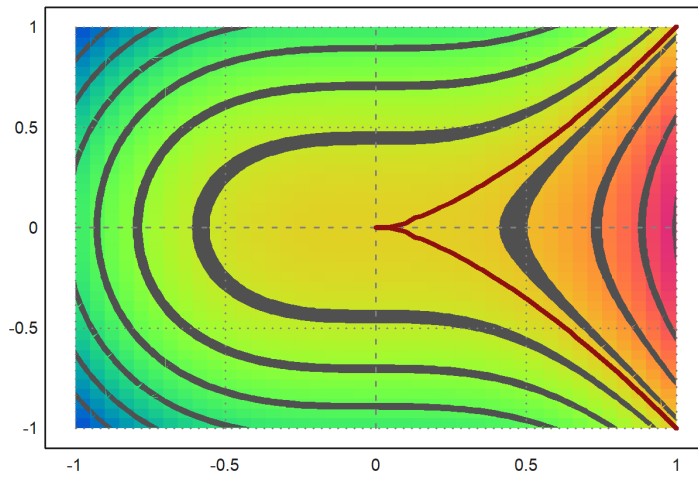
```
>x=-2:0.05:1; y=x'; z=expr(x,y);  
>plot2d(z,level=0,a=-1,b=2,c=-2,d=1,>hue):
```



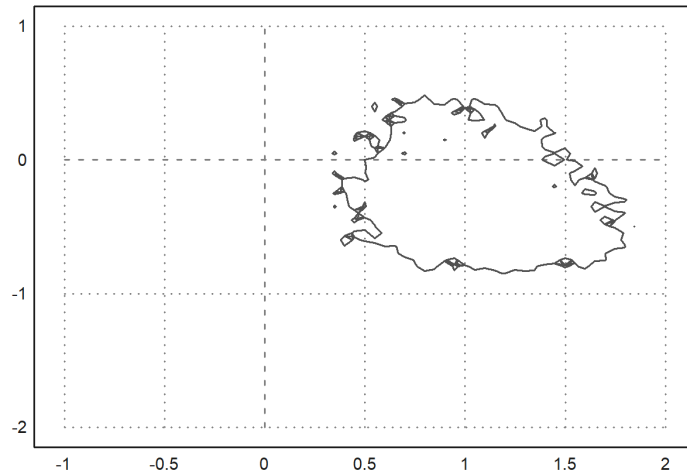
```
>plot2d("x^3-y^2",>contour,>hue,>spectral):
```



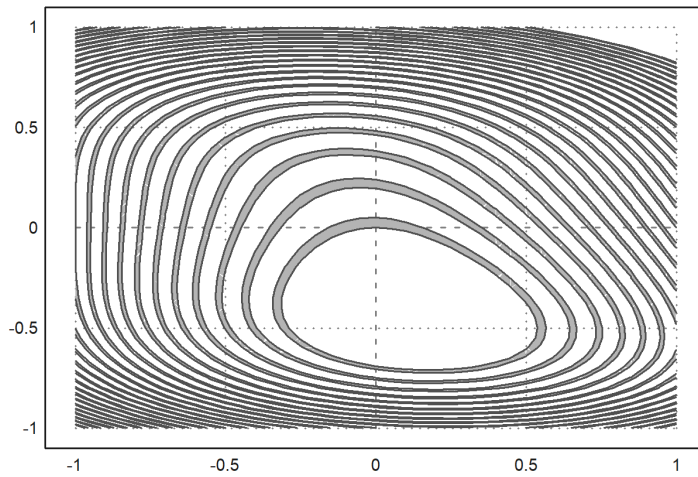
```
>plot2d("x^3-y^2",level=0,contourwidth=3,>add,contourcolor=red):
```



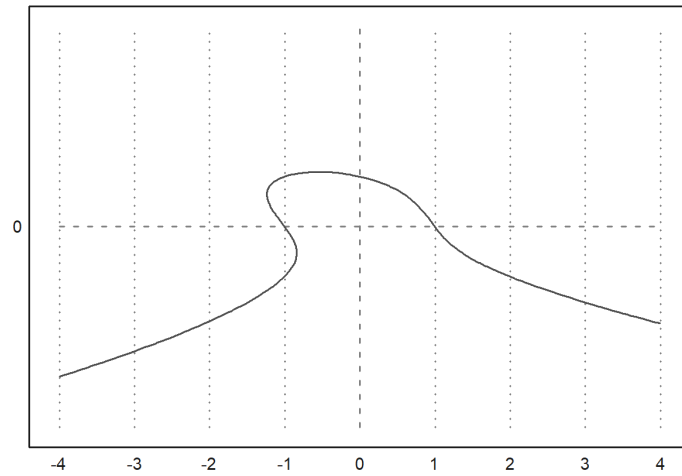
```
>z=z+normal(size(z))*0.2;  
>plot2d(z,level=0.5,a=-1,b=2,c=-2,d=1):
```



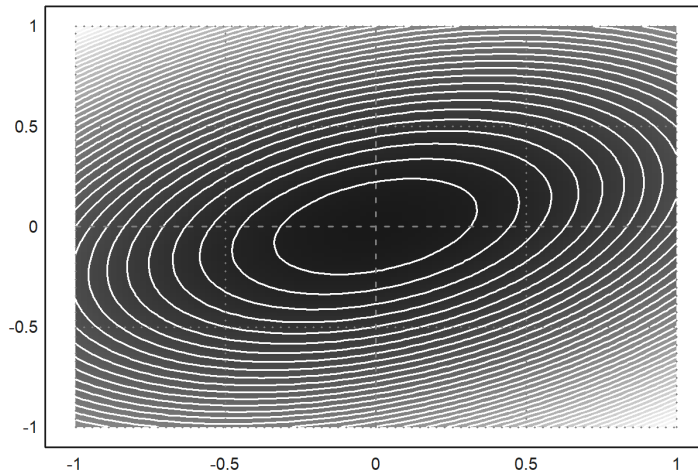
```
>plot2d(expr,level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
```



```
>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):
```

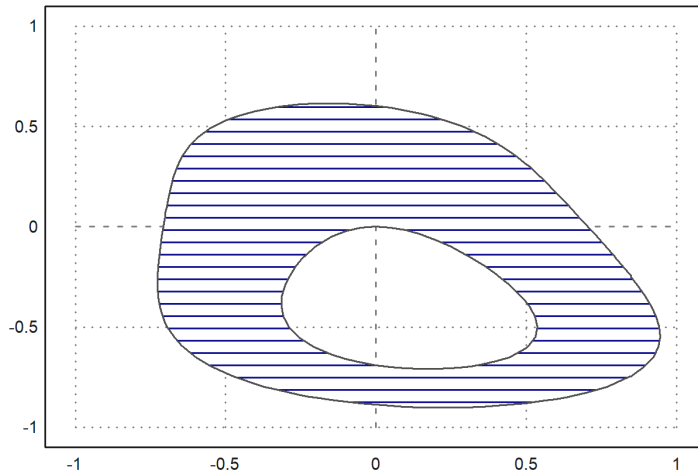
Anda juga dapat mengisi himpunan tersebut.

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

dengan rentang level.

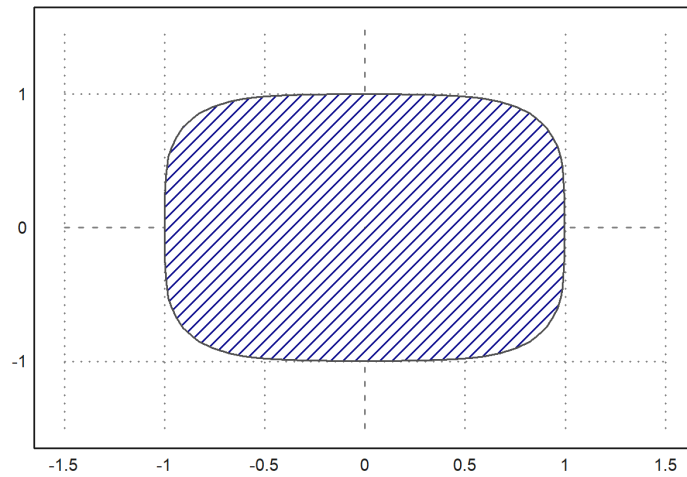
Dimungkinkan untuk mengisi daerah nilai untuk fungsi tertentu. Untuk ini, level harus berupa matriks 2xn. Baris pertama adalah batas bawah, dan baris kedua berisi batas atas.

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

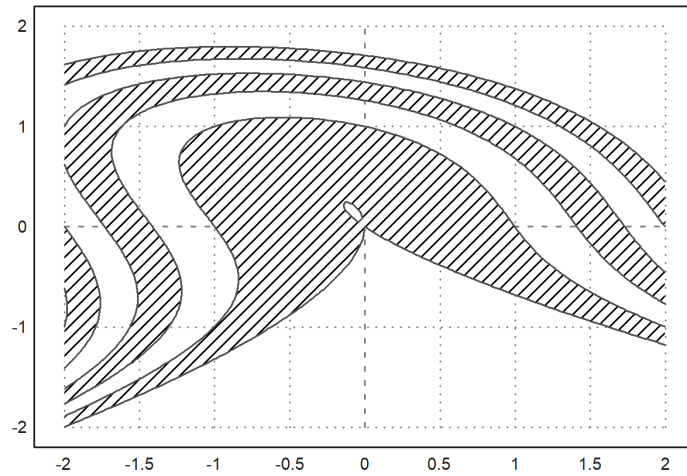


Plot implisit juga dapat menunjukkan rentang level. Dalam hal ini, level harus berupa matriks 2xn dari interval level, di mana baris pertama berisi awal setiap interval dan baris kedua berisi akhir setiap interval. Sebagai alternatif, vektor baris sederhana dapat digunakan untuk level, dan parameter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

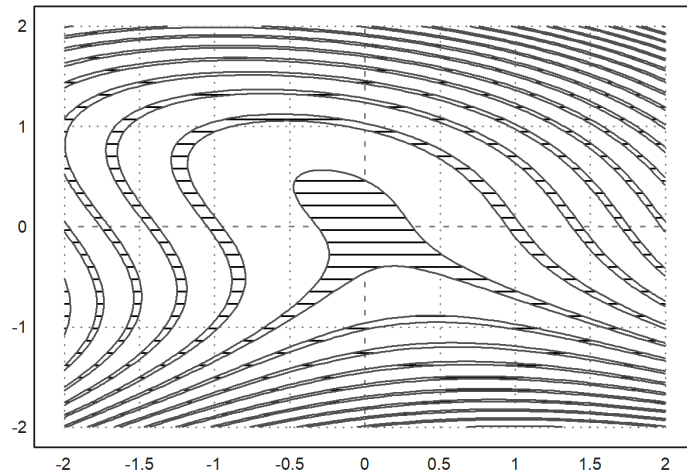
```
>plot2d("x^4+y^4",r=1.5,level=[0;1],color=blue,style="/"):
```



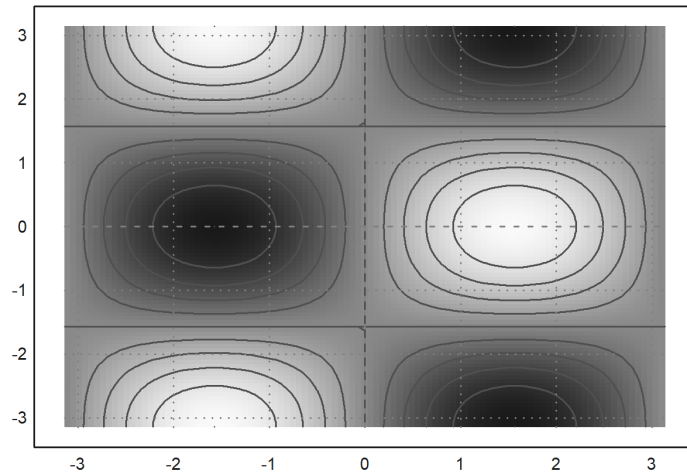
```
>plot2d("x^2+y^3+x*y",level=[0,2,4;1,3,5],style="/",r=2,n=100):
```



```
>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):
```

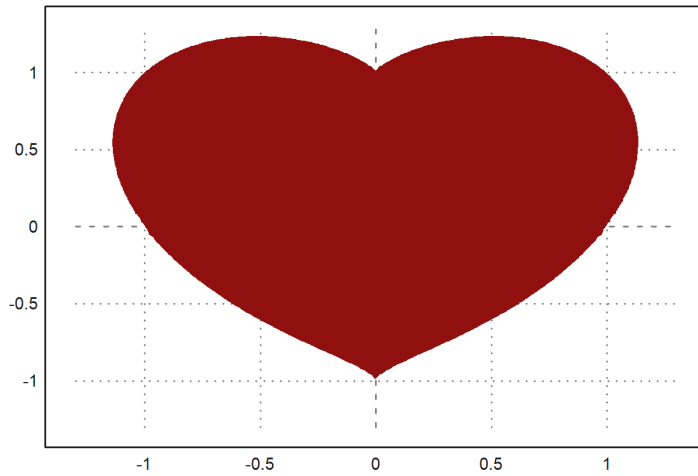


Dimungkinkan juga untuk menandai sebuah daerah

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

Ini dilakukan dengan menambahkan level yang terdiri dari dua baris.

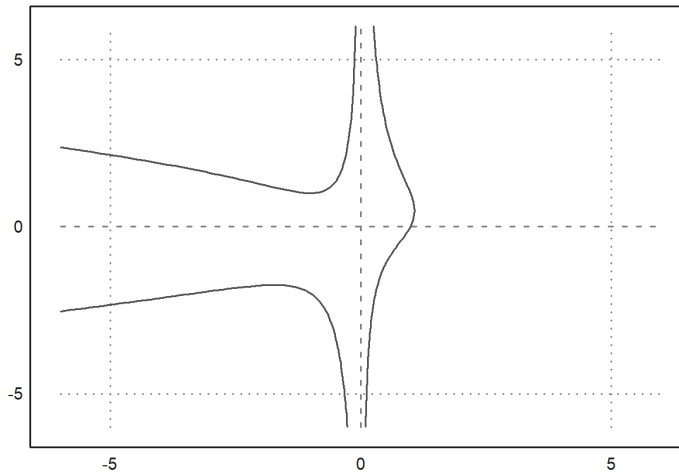
```
>plot2d("(x^2+y^2-1)^3-x^2*y^3",r=1.3, ...
> style="#",color=red,<outline, ...
> level=[-2;0],n=100):
```



Dimungkinkan untuk menentukan level tertentu. Misalnya, kita bisa memplot solusi dari sebuah persamaan seperti

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

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



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

```

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

```

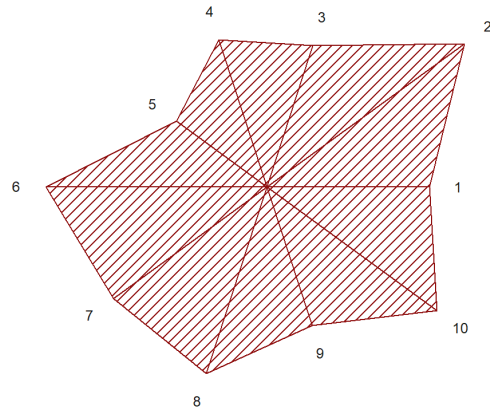


```
        ctext(""+lab[#],col,row-textheight()/2);
    endif;
end;
barcolor(cl); barstyle(st);
holding(h);
window(w);
endfunction
```

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

Kita memanggil "reset" sebelum menguji plot ini untuk mengembalikan pengaturan grafis ke default. Ini tidak diperlukan jika Anda yakin bahwa plot Anda berfungsi dengan baik.

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



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

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

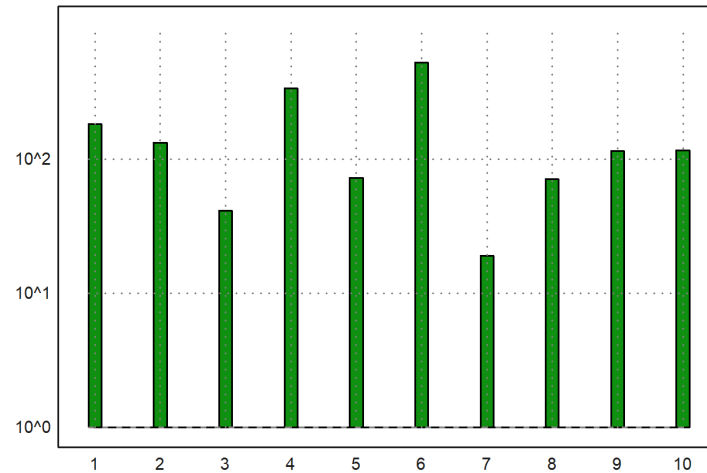
```
>function logimpulseplot1 (x,y) ...
```

```
{x0,y0}=makeimpulse(x,log(y)/log(10));  
plot2d(x0,y0,>bar,grid=0);  
h=holding(1);
```

```
frame();  
xgrid(ticks(x));  
p=plot();  
for i=-10 to 10;  
    if i<=p[4] and i>=p[3] then  
        ygrid(i,yt="10^"+i);  
    endif;  
end;  
holding(h);  
endfunction
```

Mari kita uji dengan nilai-nilai yang terdistribusi eksponensial.

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



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

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

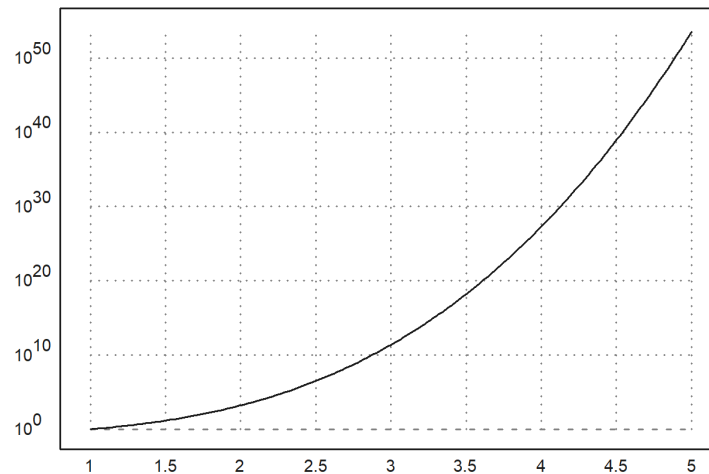
```
>function animliss (n,m) ...
```

```
t=linspace(0,2pi,500);  
f=0;  
c=framecolor(0);  
l=linewidth(2);  
setplot(-1,1,-1,1);  
repeat  
    clg;
```

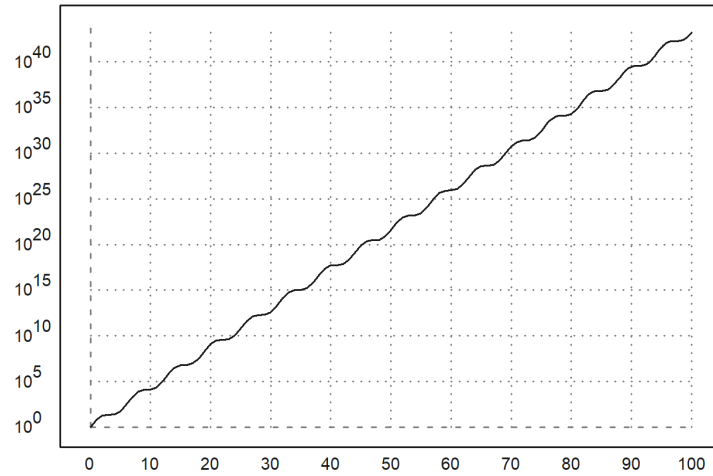
```
plot(sin(n*t),cos(m*t+f));  
wait(0);  
if testkey() then break; endif;  
f=f+0.02;  
end;  
framecolor(c);  
linewidth(1);  
endfunction
```

Tekan sembarang tombol untuk menghentikan animasi ini.

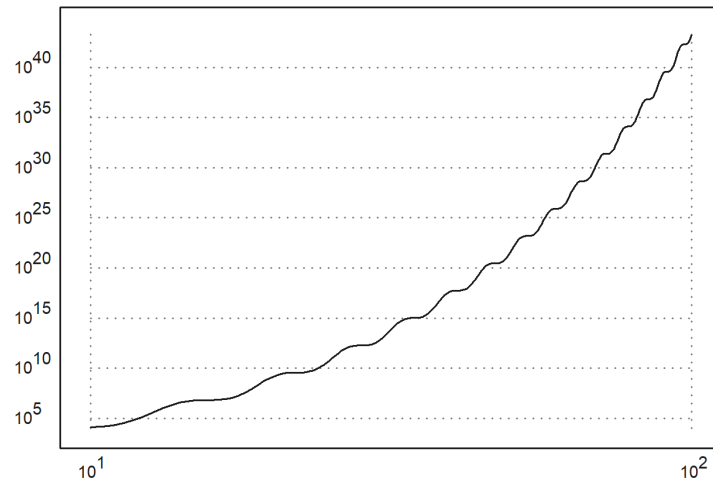
```
>animliss(2,3); // lihat hasilnya, jika sudah puas, tekan ENTER
```



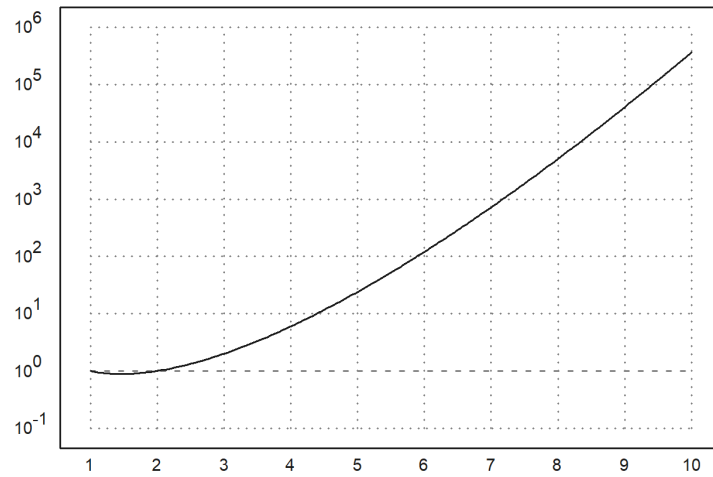
```
>plot2d("exp(x+sin(x))",0,100,logplot=1):
```



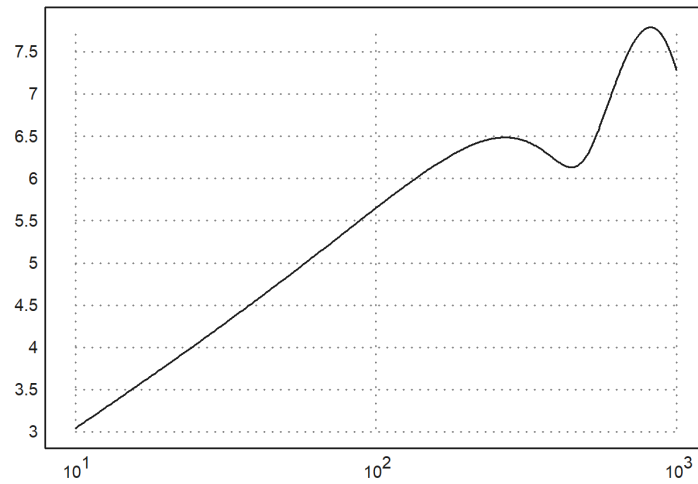
```
>plot2d("exp(x+sin(x))",10,100,logplot=2):
```



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

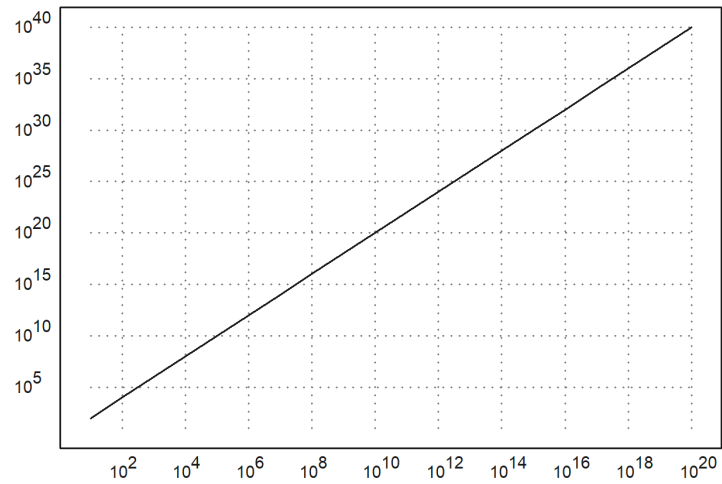



```
>plot2d("log(x*(2+sin(x/100)))",10,1000,logplot=3):
```



Ini juga berlaku untuk plot data.

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



```
>reset;
```

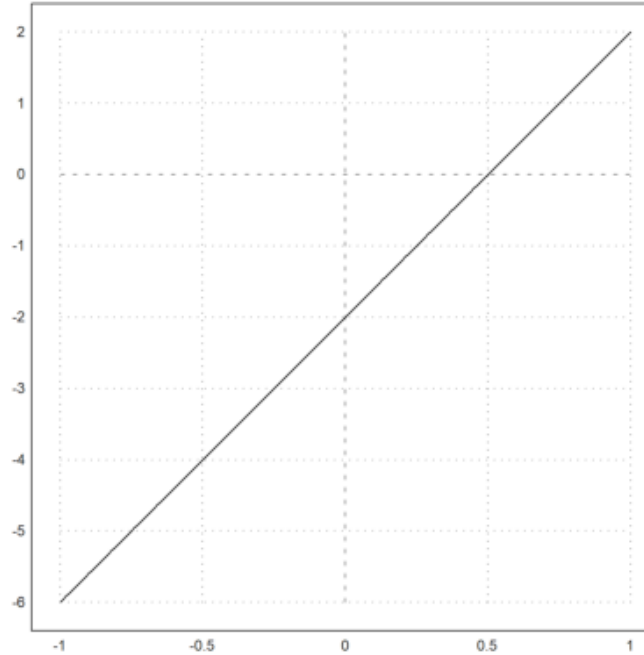
Latihan soal

Ummi Nurrohmah (24030130095)

1. Tentukan titik potong garis berikut dengan sumbu x dan sumbu y menggunakan metode grafik

$$y = 4x - 2$$

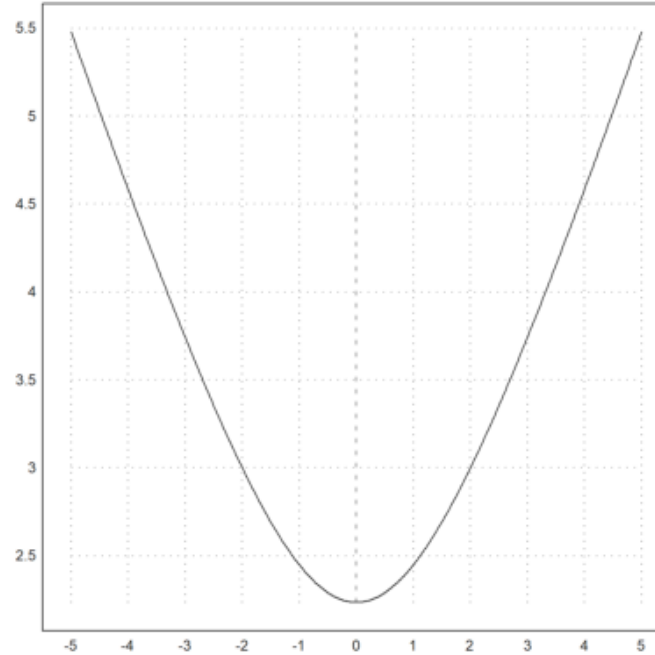
```
>plot2d("4*x-2",-1,1) :
```



2. Gambarlah grafik fungsi berikut pada interval x $(-5,5)$

$$y = \sqrt{x^2 + 5}$$

```
>expr &= sqrt(x^2+5);  
>plot2d(expr,-5,5):
```

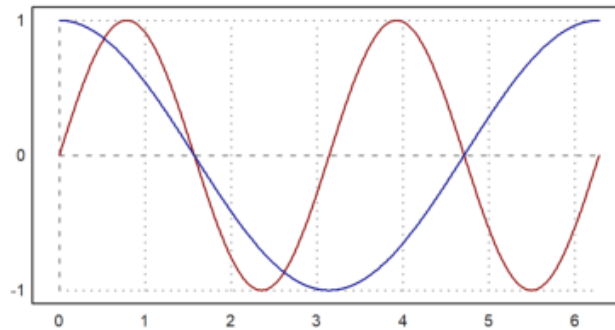


3. Gambarkan kurva fungsi-sungsi berikut dalam satu bidang koordinat

$$y = \sin(2x)$$

$$y = \cos(x)$$

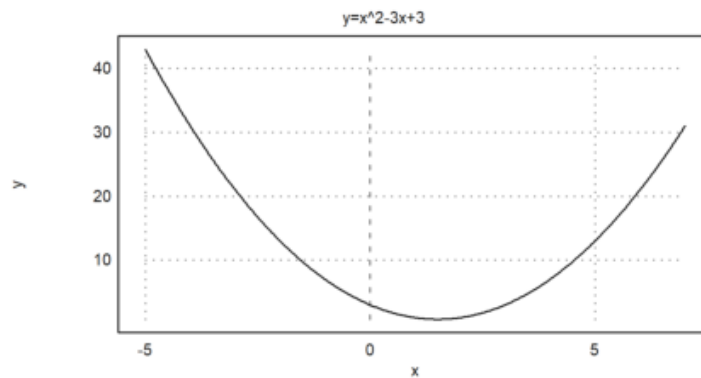
```
>aspect(2);
>plot2d(["sin(2*x)","cos(x)"],0,2pi,color=[red,blue]):
```



4. Gambarlah kurva fungsi berikut, disertai keterangannya

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

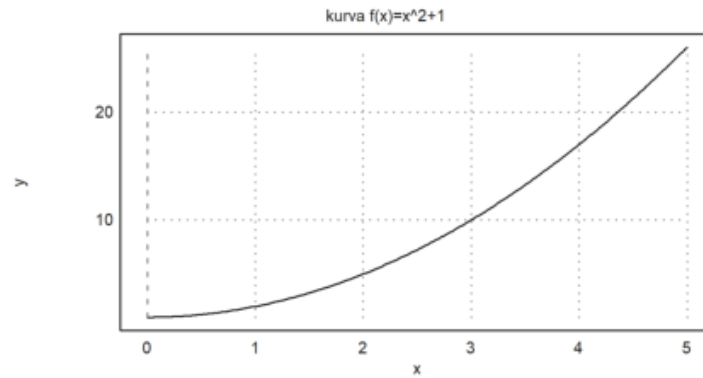
```
>plot2d("x^2-3x+3",-5,7,title="y=x^2-3x+3",xl="x",yl="y") :
```



5. Buatlah plot area di bawah kurva $f(x)$ untuk nilai x dari 0 hingga 5

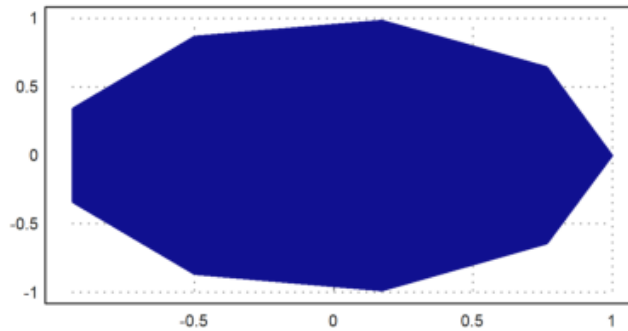
$$f(x) = x^2 + 1$$

```
>plot2d("x^2+1",0,5,xl="x",yl="y",title="kurva f(x)=x^2+1"):
```



6. Gambarkanlah bangun segi-sembilan pada bidang koordinat

```
>t=linspace(0,2pi,9); plot2d(cos(t),sin(t),>filled,style="#",fillcolor=blue):
```

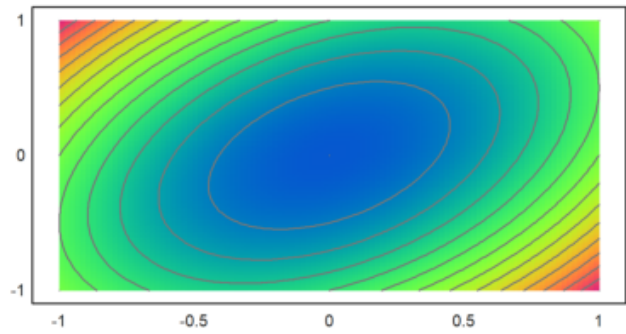



```
>expr := "3*x^2-2*x*y+2*y^2";...
```

7. Gambarlah grafik solusi fungsi implisit berikut :

$$f(x, y) = 3x^2 - 2xy + 2y^2 = 0$$

```
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=100,grid=4):
```



Rujukan Lengkap Fungsi plot2d()

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

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

Parameters

x,y : equations, functions or data vectors

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

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

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

xmin,xmax : range of the parameter for curves

auto : Determine y-range automatically (default)

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

n : number of intervals (default is adaptive)

grid : 0 = no grid and labels,

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

frame : 0 = no frame

framecolor: color of the frame and the grid

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

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

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

thickness : line thickness for curves

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

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

```

For points use
"[]", "<>", ".", "..", "...",
"*", "+", "|", "-", "o"
"[]#", "<>#", "o#" (filled shapes)
"[]w", "<>w", "ow" (non-transparent)
For lines use
"-", "--", "-.", ".", "-.-", "-.-", "->"
For filled polygons or bar plots use
"#", "#0", "0", "/", "\", "\/",
"+", "|", "-", "t"

```

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

value

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

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

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

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

own :

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

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

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

cgrid :

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

Overview

The function can plot

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

If a function or expression for `xv` is given, `plot2d()` will compute values in the given range using the function or expression. The

expression must be an expression in the variable `x`. The range must be defined in the parameters `a` and `b` unless the default range should be used. The y-range will be computed automatically, unless `c` and `d` are specified, or a radius `r`, which yields the range `r,r`

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

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