

Menggambar Grafik 2D dengan EMT

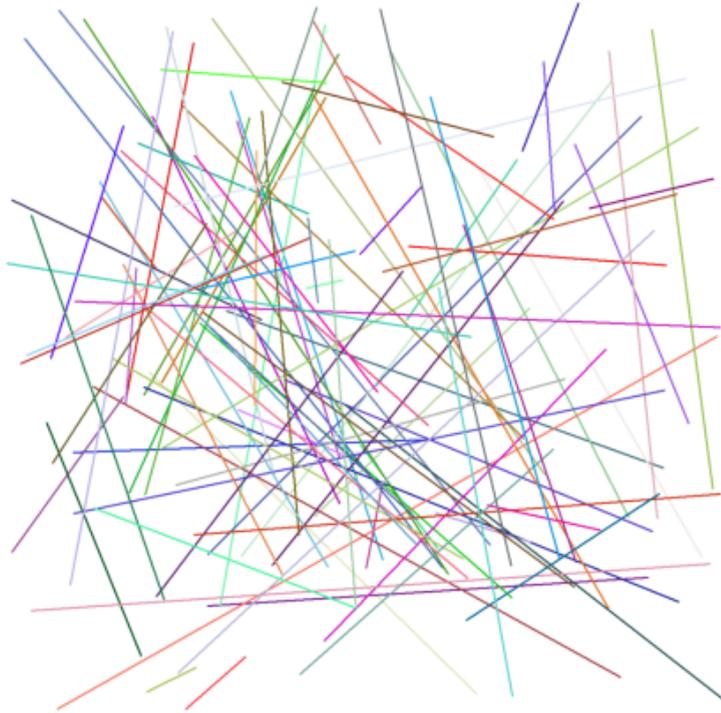
Notebook ini menjelaskan tentang cara menggambar berbagai kurva dan grafik 2D dengan software EMT. EMT menyediakan fungsi `plot2d()` untuk menggambar berbagai kurva dan grafik dua dimensi (2D).

Plot Dasar

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

Dalam contoh, kita hanya menggambar beberapa garis acak dalam berbagai warna. Untuk detail tentang fungsi ini, pelajari fungsi inti EMT.

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



```
>reset;
```

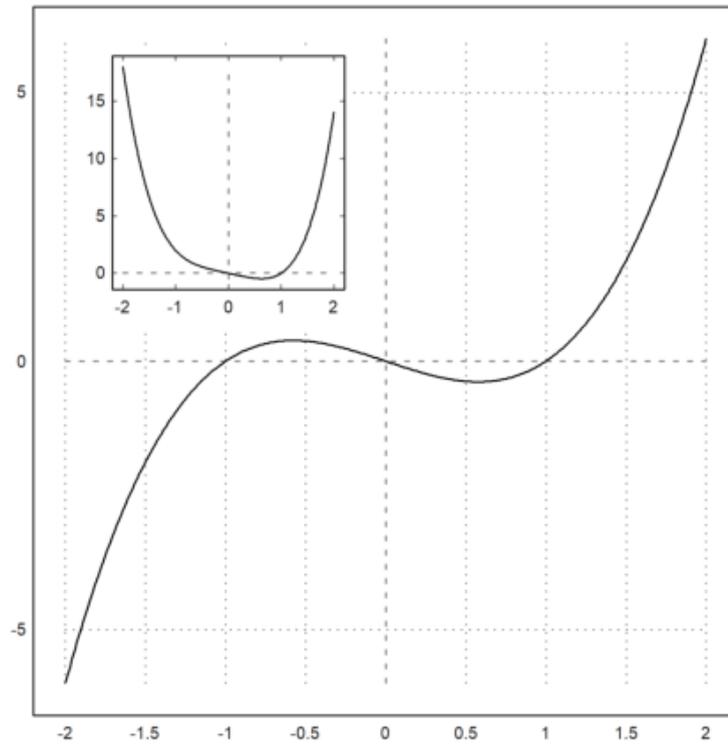
Grafik perlu ditahan, karena perintah `plot()` akan menghapus jendela plot.

Untuk menghapus semua yang kami lakukan, kami 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.

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

```
>plot2d("x^3-x");
>xw=200; yw=100; ww=300; hw=300;
>ow=window();
>window(xw,yw,xw+ww,yw+hw);
>hold on;
>barclear(xw-50,yw-10,ww+60,ww+60);
>plot2d("x^4-x",grid=6):
```



```
>hold off;  
>window(ow);
```

Plot dengan banyak angka dicapai dengan cara yang sama. Ada fungsi `figure()` utilitas untuk ini.

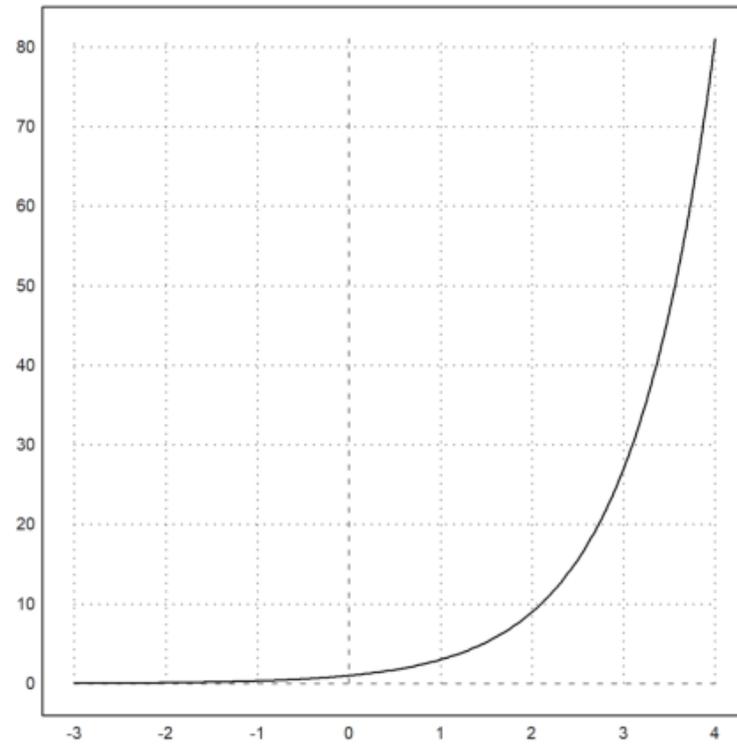
CONTOH SOAL:

$$3^x$$

```
>reset;
```

untuk menghapus yang sebelumnya

```
>function f(x):= 3^x  
>plot2d("f(x)",-3,4):
```

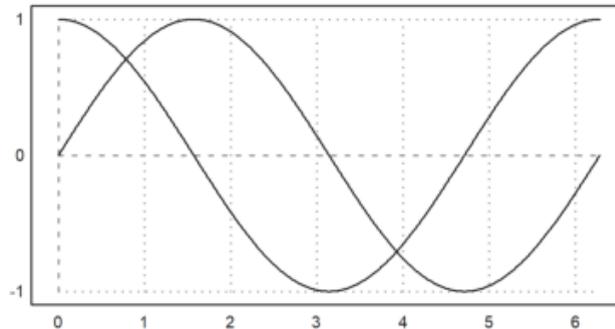


Aspek Plot

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

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

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



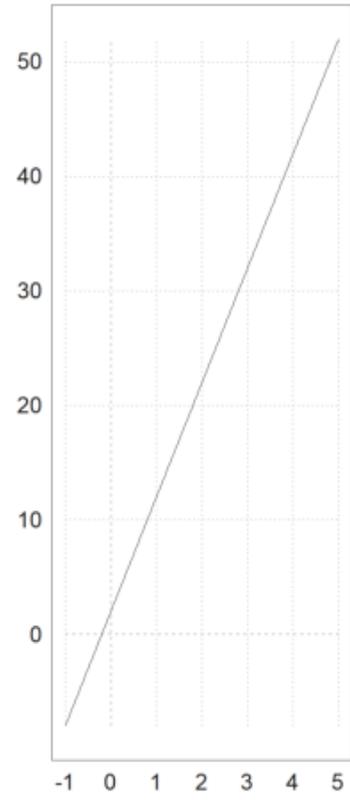
```
>aspect();  
>reset;
```

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

CONTOH SOAL:

$$10x + 2$$

```
>aspect(2,5), plot2d("10*x+2",-1,5):
```



```
>reset;
```

Plot 2D di Euler

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

Dimungkinkan untuk membuat plot di Maxima menggunakan Gnuplot atau dengan Python menggunakan Math Plot Lib.

Euler dapat memplot plot 2D dari

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

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

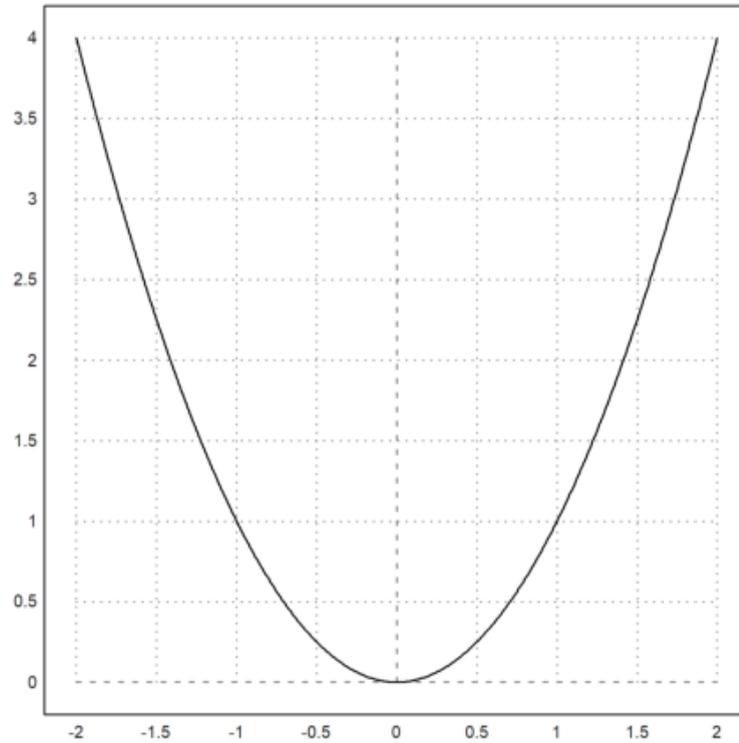
Plot Ekspresi atau Variabel

Ekspresi tunggal dalam "x" (mis. "4*x^2") atau nama fungsi (mis. "f") menghasilkan grafik fungsi.

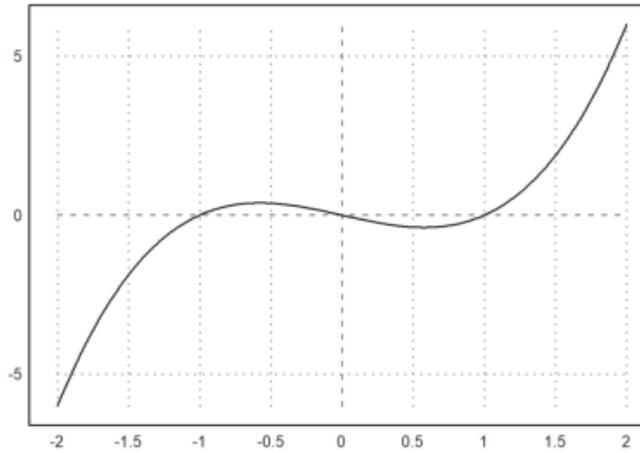
Berikut adalah contoh paling dasar, yang menggunakan rentang default dan menetapkan rentang y yang tepat agar sesuai dengan plot fungsi.

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

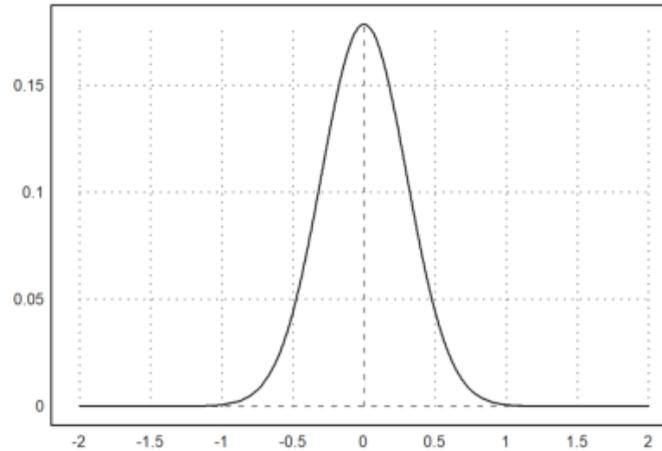
```
>plot2d("x^2"):
```



```
>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
```

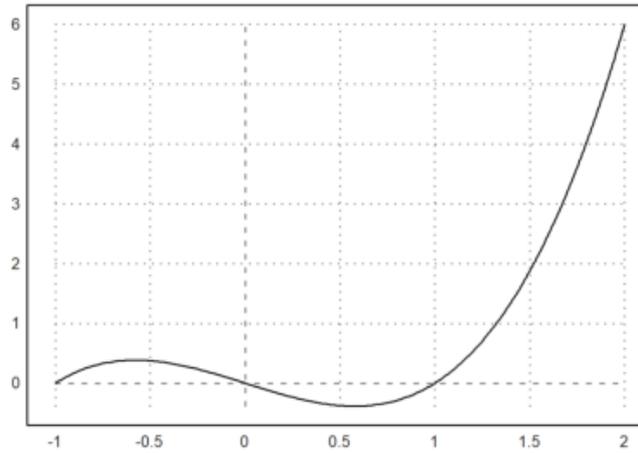


Dari beberapa contoh sebelumnya Anda dapat melihat bahwa aslinya gambar plot menggunakan sumbu X dengan rentang nilai dari -2 sampai dengan 2. Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan nilai-nilai batas X (dan Y) di belakang ekspresi yang digambar.

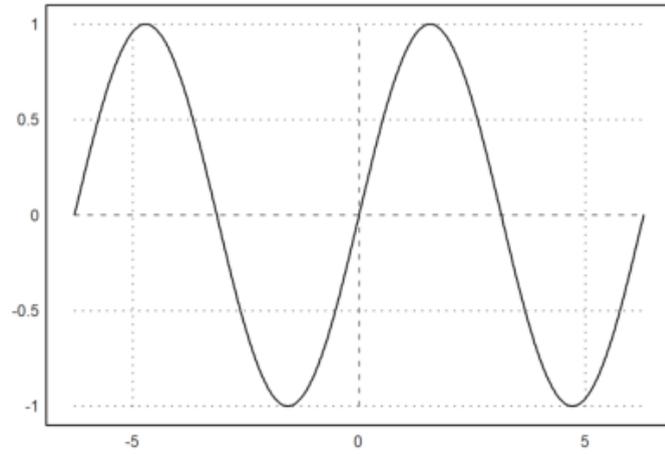
Rentang plot diatur dengan parameter yang ditetapkan berikut:

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

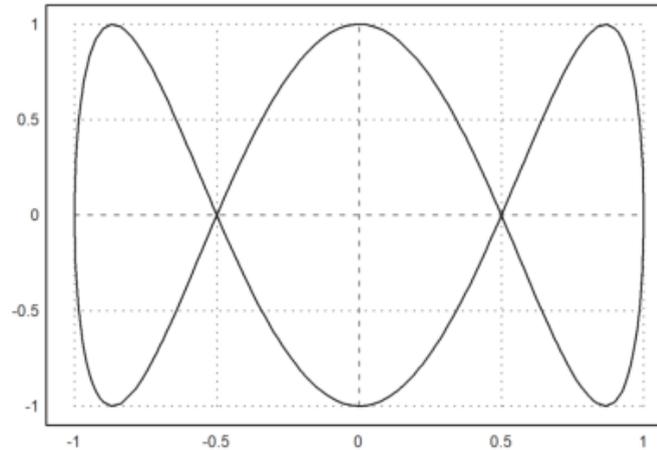
```
>plot2d("x^3-x",-1,2):
```



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



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



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

Dalam opsi, plot dapat diatur untuk muncul

- di jendela terpisah yang dapat diubah ukurannya,

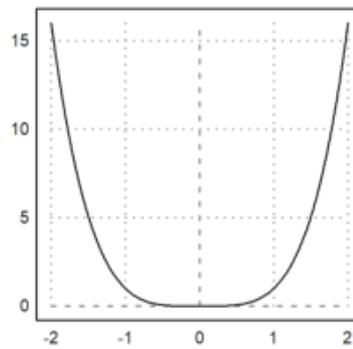
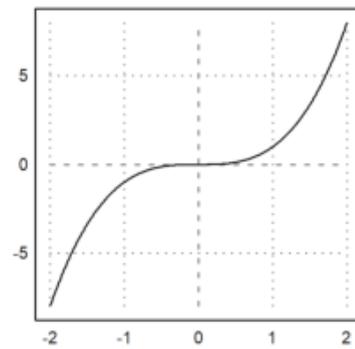
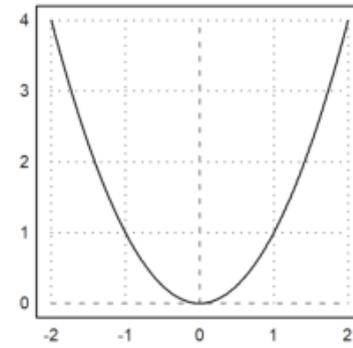
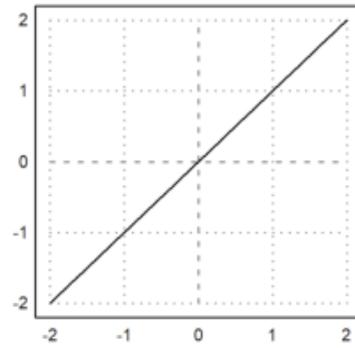
di jendela buku catatan.

Lebih banyak gaya dapat dicapai dengan perintah plot tertentu.

agaimanapun, tekan tombol tabulator untuk melihat plot, jika disembunyikan.

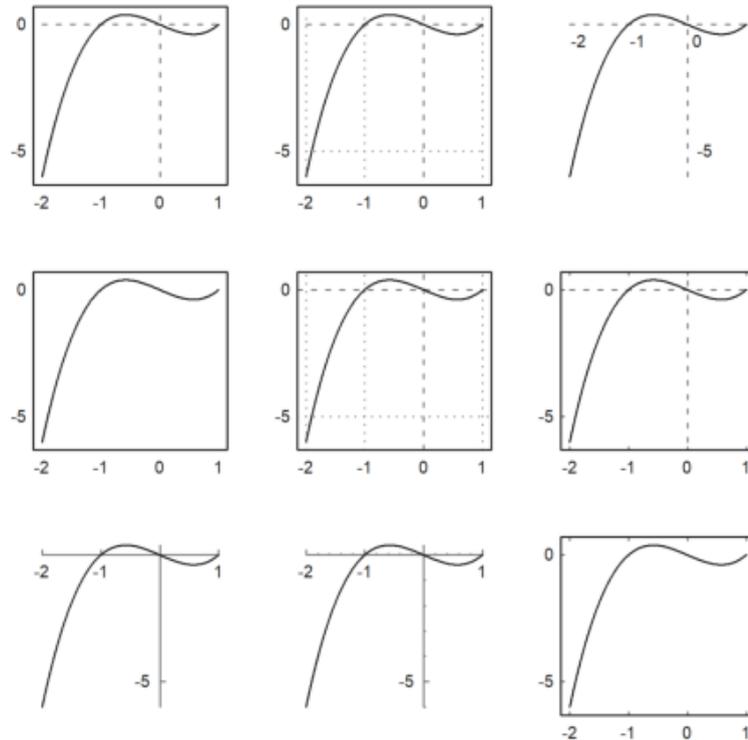
Untuk membagi jendela menjadi beberapa plot, gunakan perintah `figure()`. Dalam contoh, kami memplot x^1 hingga x^4 menjadi 4 bagian jendela. `figure(0)` mengatur ulang jendela default.

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



Di `plot2d()`, ada gaya alternatif yang tersedia dengan `grid=x`. Untuk gambaran umum, kami menunjukkan berbagai gaya kisi dalam satu gambar (lihat di bawah untuk perintah `figure()`). Gaya `kisi=0` tidak disertakan. Ini menunjukkan tidak ada 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):
```

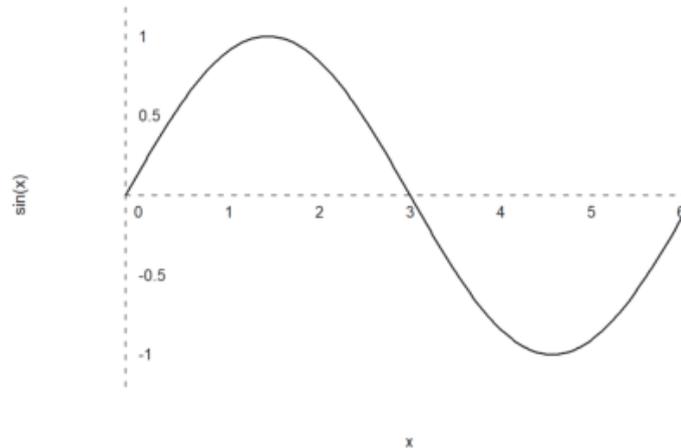


Jika argumen ke `plot2d()` adalah ekspresi yang diikuti oleh empat angka, angka-angka ini adalah rentang x dan y untuk plot.

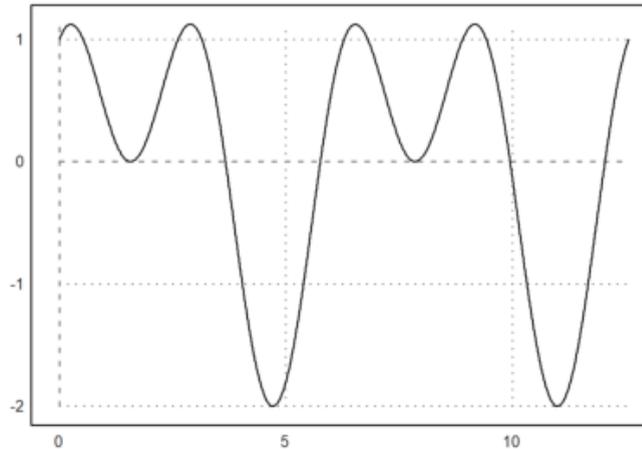
Atau, a, b, c, d dapat ditentukan sebagai parameter yang ditetapkan sebagai `a=...` dll.

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

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



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

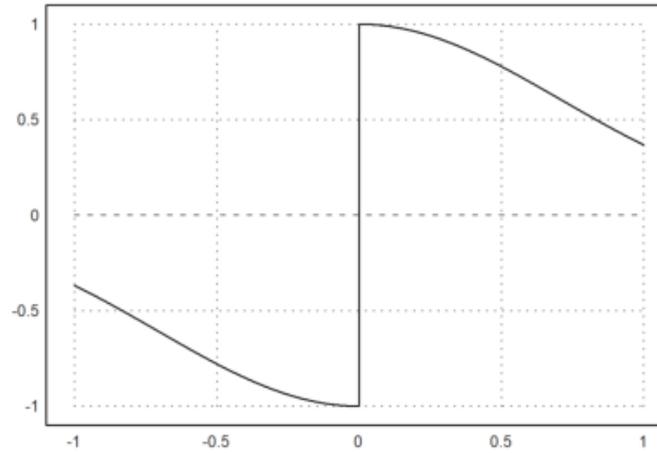


Gambar yang dihasilkan dengan memasukkan plot ke dalam jendela teks disimpan di direktori yang sama dengan buku catatan, secara default di subdirektori bernama "gambar". Mereka juga digunakan oleh ekspor HTML.

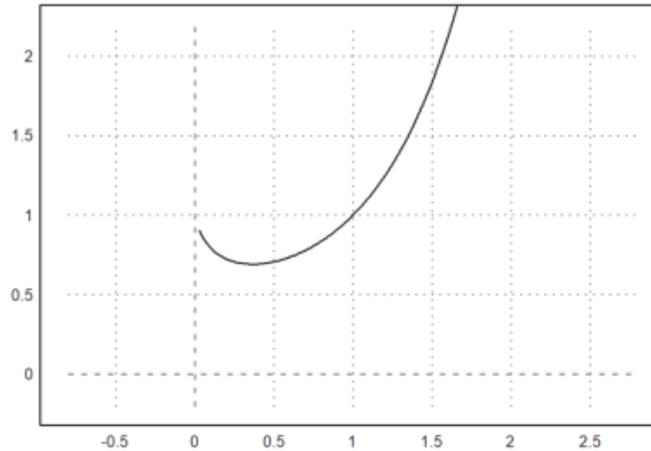
Anda cukup menandai gambar apa saja dan menyalinnya ke clipboard dengan Ctrl-C. Tentu saja, Anda juga dapat mengekspor grafik saat ini dengan fungsi di menu File.

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

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

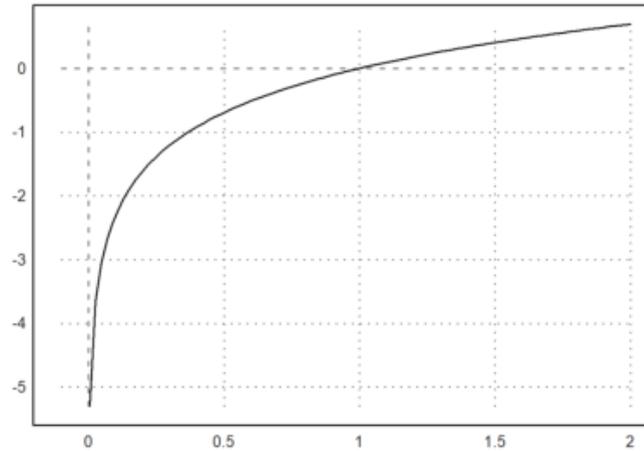


```
>plot2d("x^x",r=1.2,cx=1,cy=1):
```



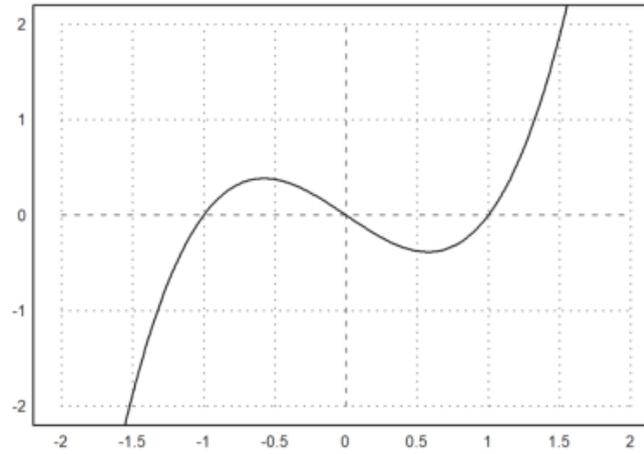
Perhatikan bahwa x^x tidak didefinisikan untuk $x \leq 0$. Fungsi `plot2d` menangkap kesalahan ini, dan mulai merencanakan segera setelah fungsi didefinisikan. Ini berfungsi untuk semua fungsi yang mengembalikan NAN keluar dari jangkauan definisinya.

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

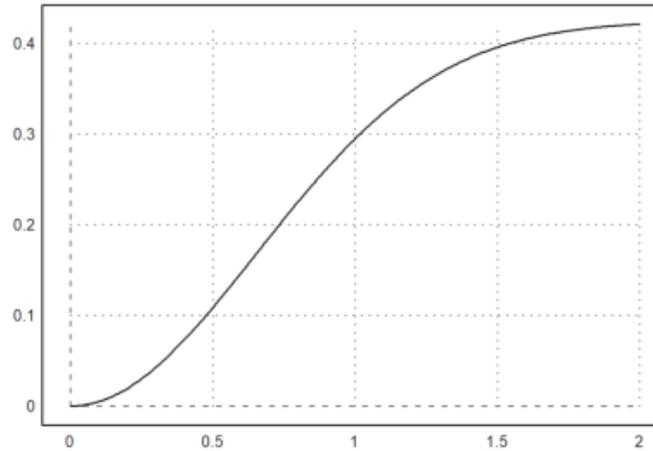


Parameter `square=true` (atau `>square`) memilih y-range secara otomatis sehingga hasilnya adalah jendela plot persegi. Perhatikan bahwa secara default, Euler menggunakan ruang persegi di dalam jendela plot.

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

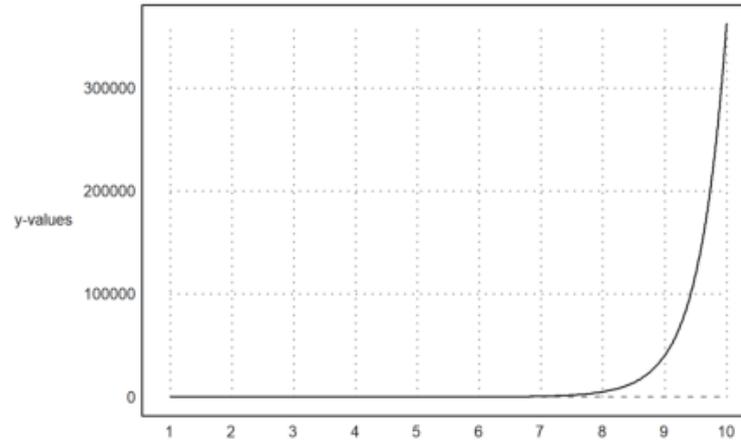


```
>plot2d('integrate("sin(x)*exp(-x^2)",0,x)',0,2): // plot integral
```



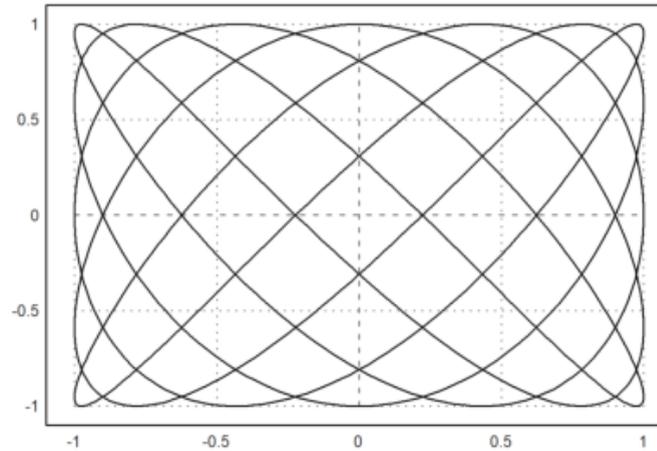
Jika Anda membutuhkan lebih banyak ruang untuk label-y, panggil `shrinkwindow()` dengan parameter yang lebih kecil, atau tetapkan nilai positif untuk "lebih kecil" di `plot2d()`.

```
>plot2d("gamma(x)",1,10,y1="y-values",smaller=6,<vertical):
```

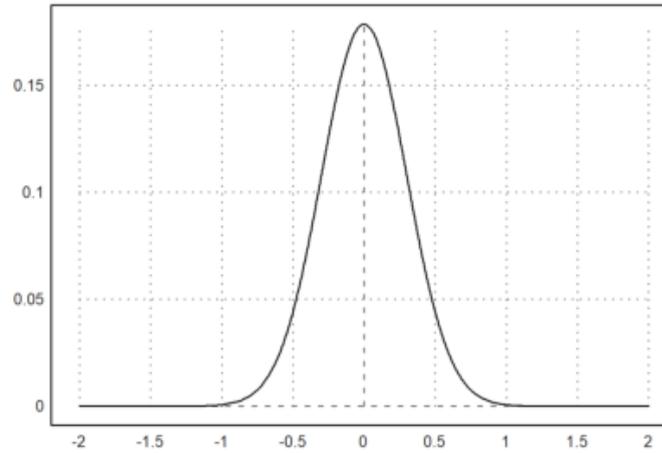


Ekspresi simbolik juga dapat digunakan, karena disimpan sebagai ekspresi string sederhana.

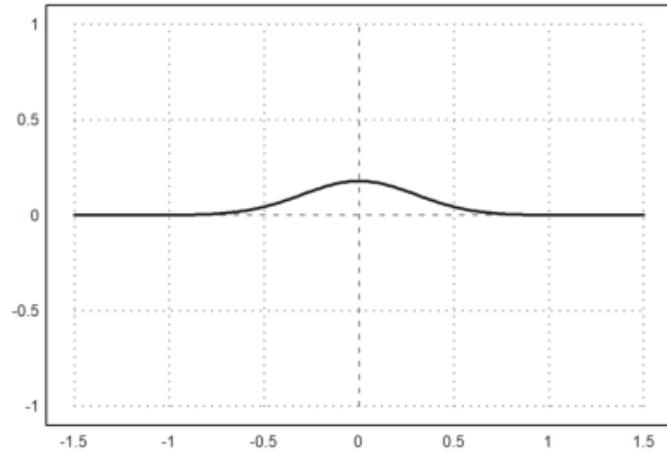
```
>x=linspace(0,2pi,1000); plot2d(sin(5x),cos(7x)):
```



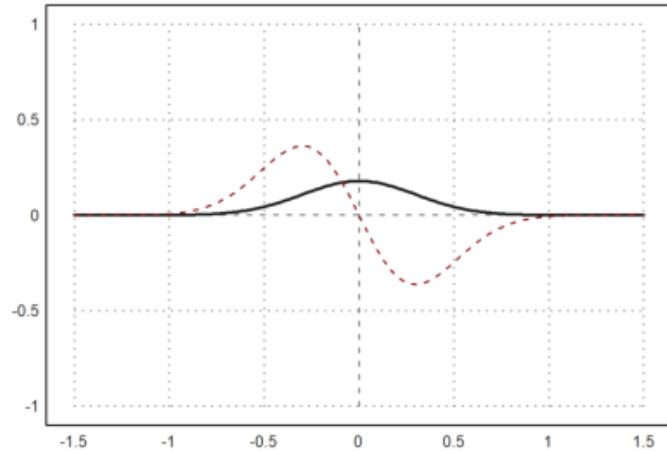
```
>a:=5.6; expr &= exp(-a*x^2)/a; // mendefinisikan ekspresi  
>plot2d(expr,-2,2): // plot dari -2 sampai 2
```



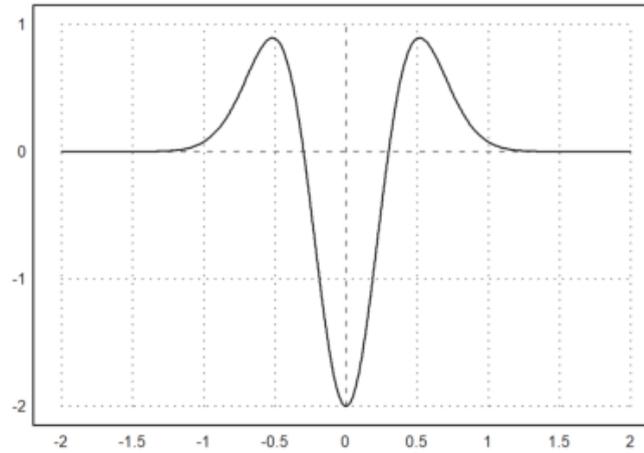
```
>plot2d(expr,r=1,thickness=2): // plot dalam bentuk persegi di sekitar (0,0)
```



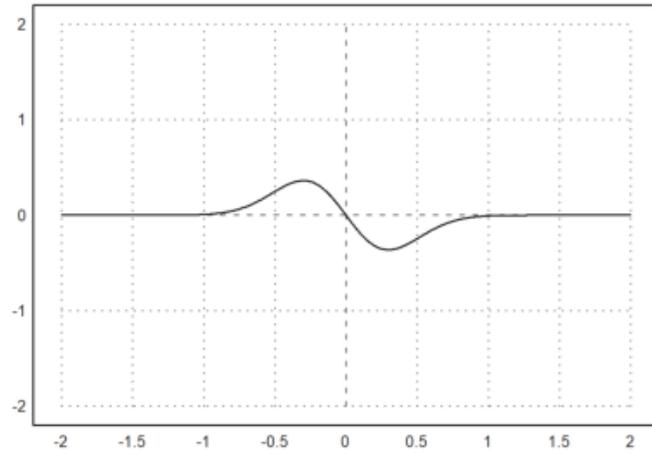
```
>plot2d(&diff(expr,x),>add,style="--",color=red): // menambahkan plot lain
```



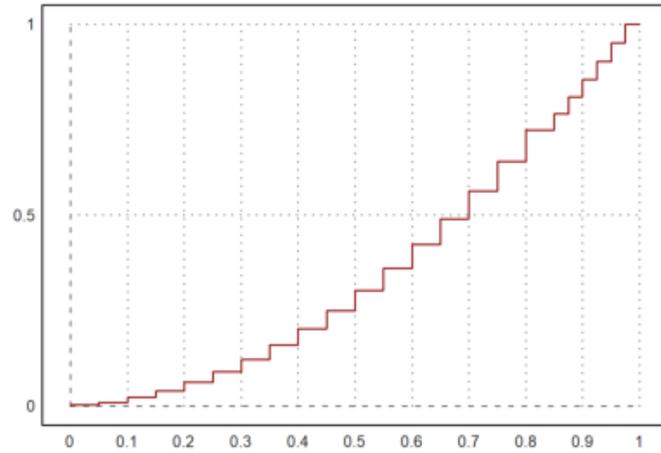
```
>plot2d(&diff(expr,x,2),a=-2,b=2,c=-2,d=1): // plot dalam persegi panjang
```



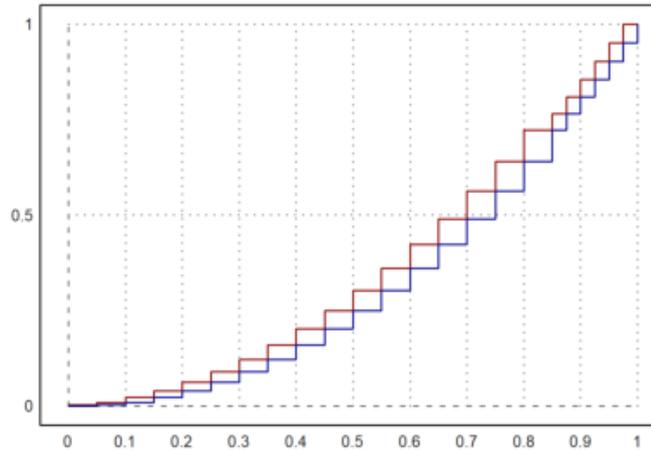
```
>plot2d(&diff(expr,x),a=-2,b=2,>square): // mempertahankan plot agar tetap persegi
```



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



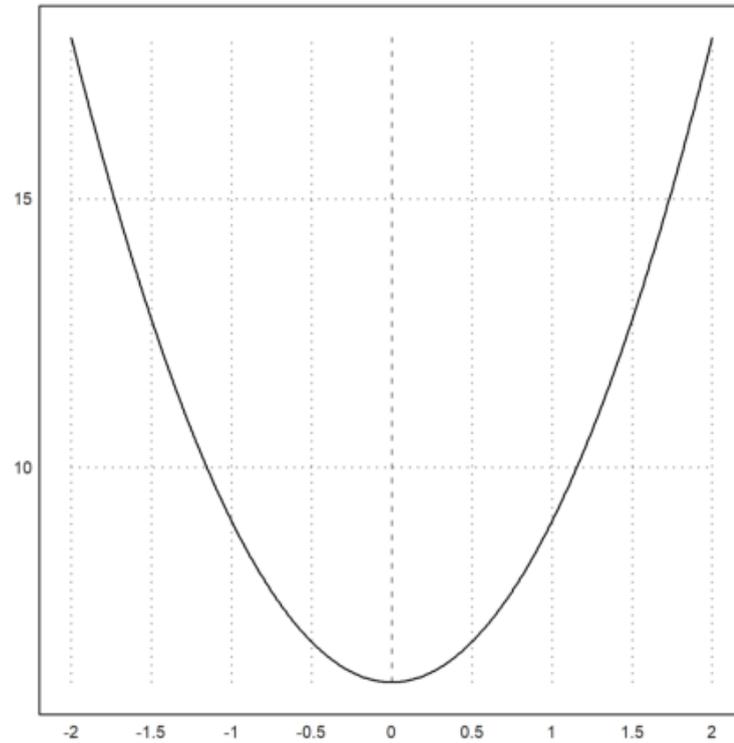
```
>plot2d("x^2",>add,steps=2,color=blue,n=10):
```



CONTOH SOAL:
Soal 1

$$f(x) = 3x^2 - 6$$

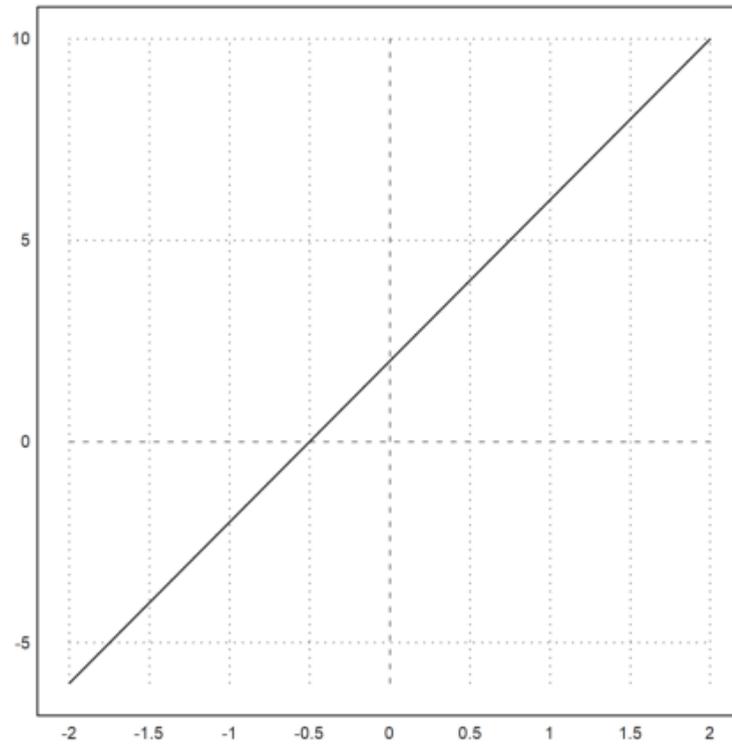
```
>reset;  
>plot2d("3*x^2+6"):
```



Soal 2

$$f(x) = 4x + 2$$

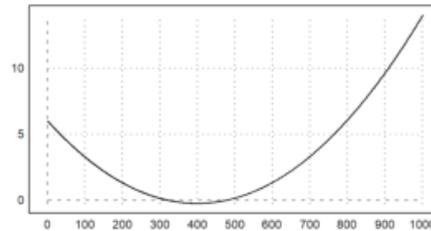
```
>reset;  
>plot2d("4*x+2"):
```



Soal 3

$$f(x) = x^2 - 5x + 6$$

```
>reset;  
>aspect(6,3);  
>plot2d(x^2-5x+6,-1,4);  
>insimg(10);
```



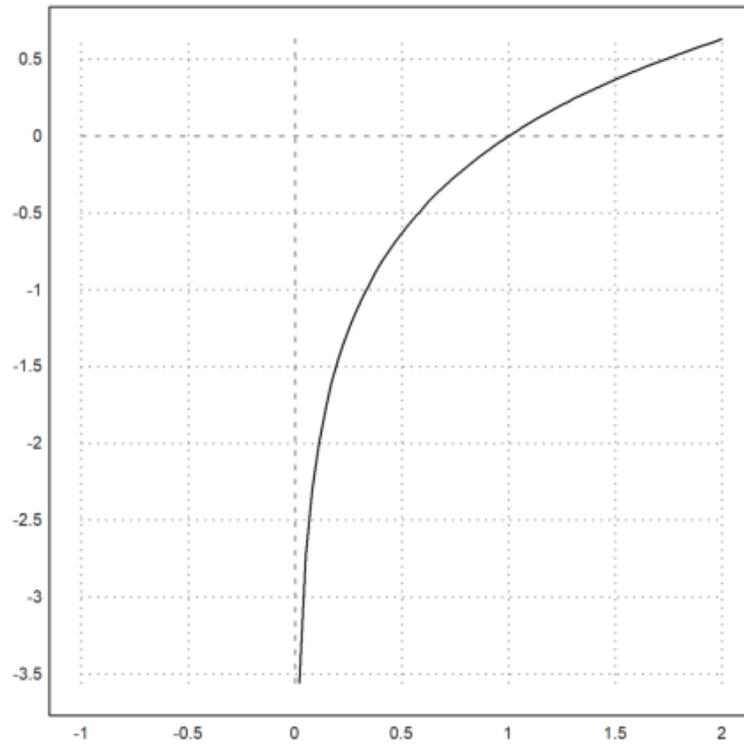
Soal 4

$$\log_3(x)$$

```
>reset;  
>$$ logbase(x,3); $$3^y=x
```

$$3^y = x$$

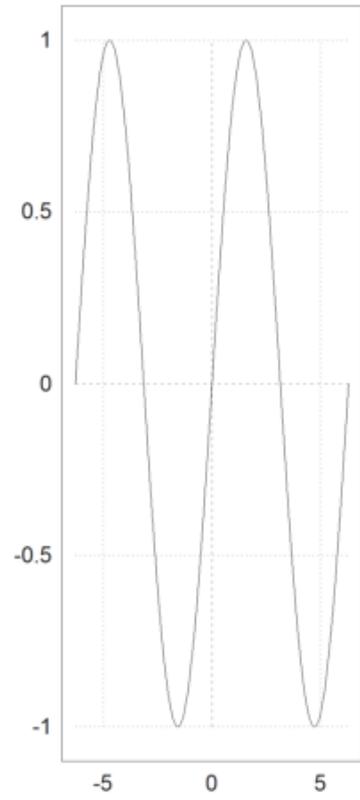
```
>plot2d("logbase(x,3)",-1,2):
```



Soal 5

$$f(x) = \sin(x)$$

```
>reset;  
>aspect(2,5), plot2d("sin(x)",-2*pi,2*pi):
```

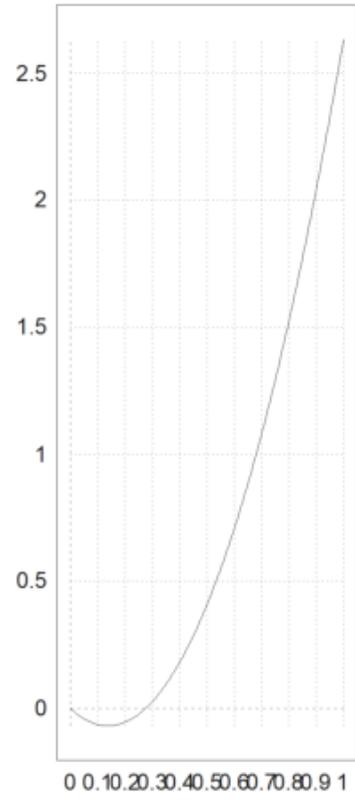


Fungsi dalam satu Parameter

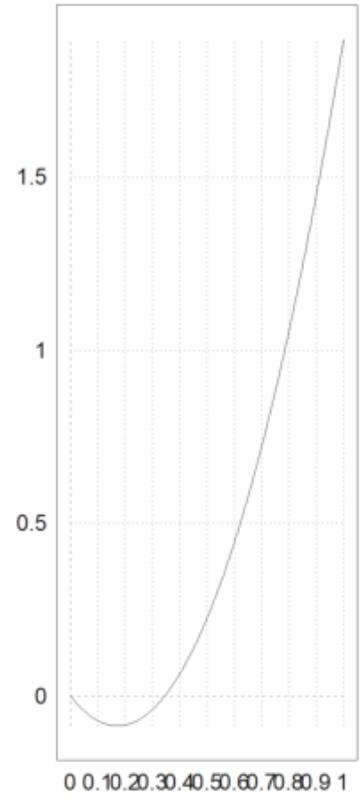
Fungsi plot yang paling penting untuk plot planar adalah `plot2d()`. Fungsi ini diimplementasikan dalam bahasa Euler dalam file "plot.e", yang dimuat di awal program.

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

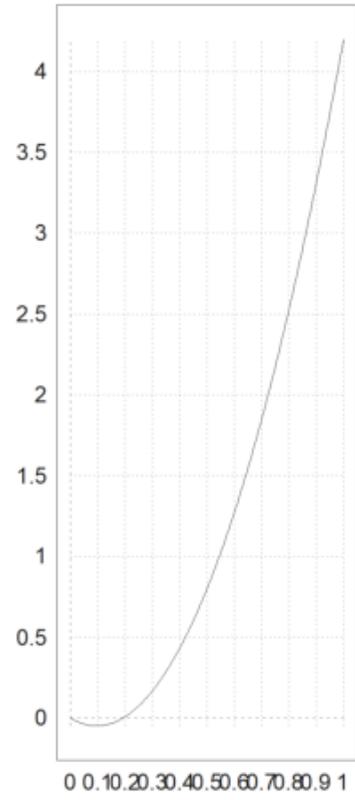
```
>function f(x,a) := x^2/a+a*x^2-x; // mendefinisikan fungsi
>a=0.3; plot2d("f",0,1;a): // plot dengan a=0.3
```



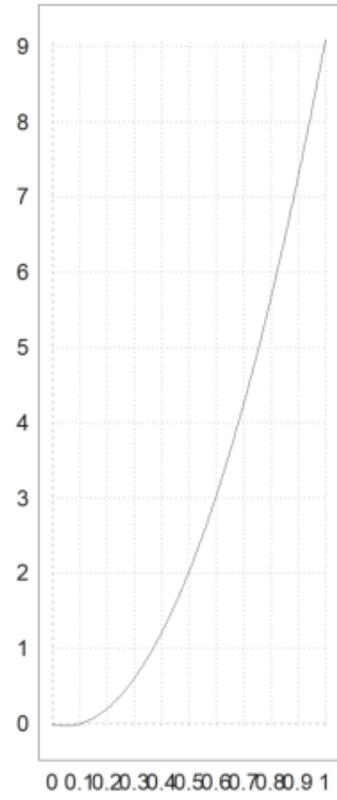
```
>plot2d("f",0,1;0.4): // plot dengan a=0.4
```



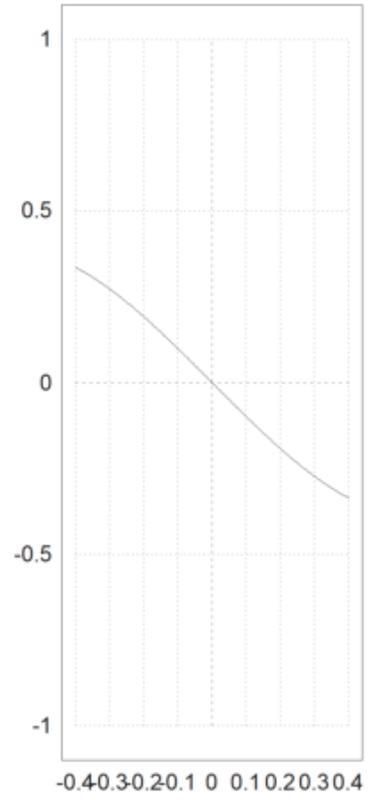
```
>plot2d({{"f",0.2}},0,1): // plot dengan a=0.2
```



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



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



Berikut adalah ringkasan dari fungsi yang diterima

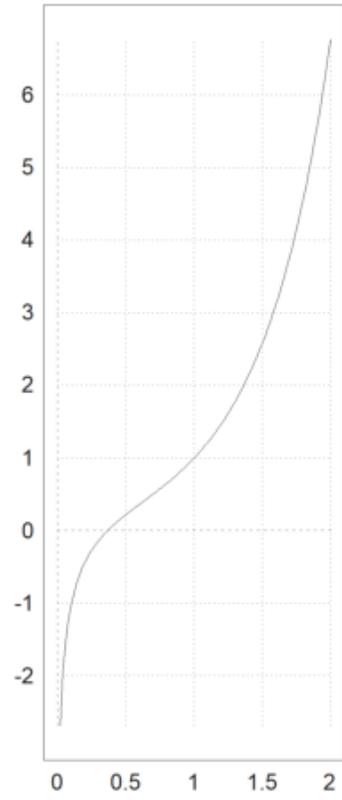
- ekspresi atau ekspresi simbolik dalam x
- fungsi atau fungsi simbolis dengan nama sebagai "f"
- fungsi simbolis hanya dengan nama f

Fungsi `plot2d()` juga menerima fungsi simbolis. Untuk fungsi simbolis, nama saja yang berfungsi.

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

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

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

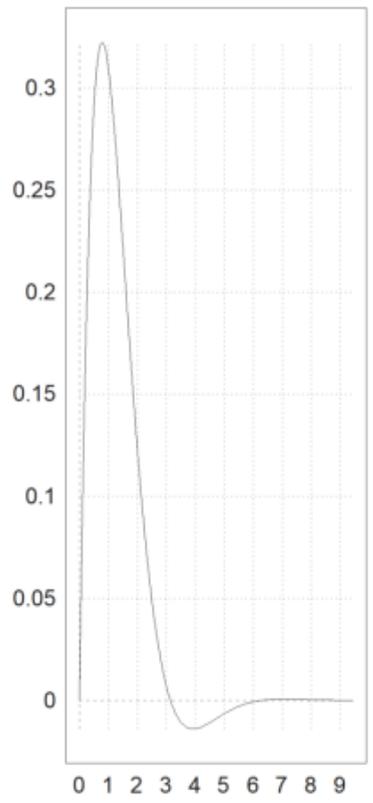


Tentu saja, untuk ekspresi atau ekspresi simbolik, nama variabel sudah cukup untuk memplotnya.

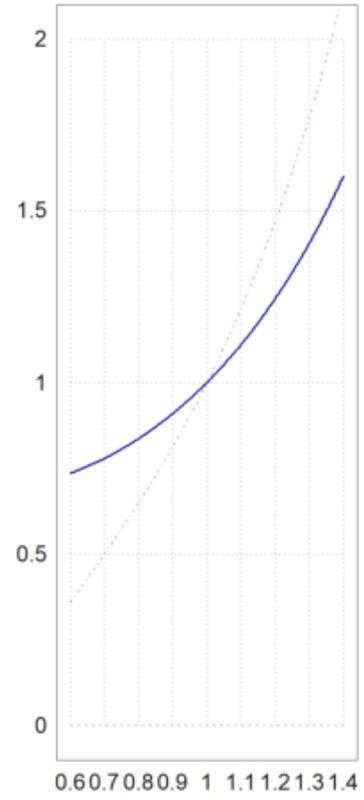
```
>expr &= sin(x)*exp(-x)
```

```
E      - 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="-.-"):
```



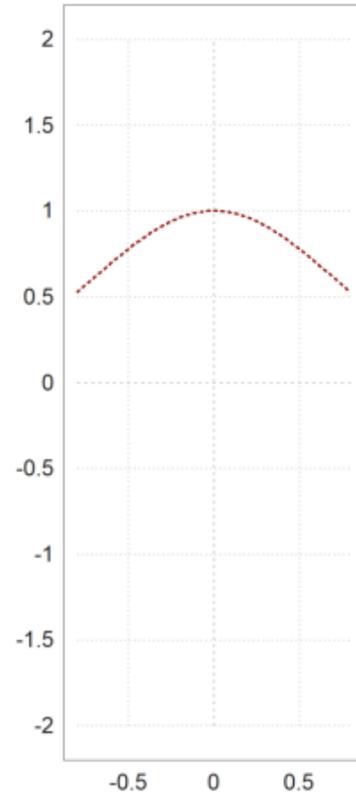
Untuk gaya garis ada berbagai pilihan.

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

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

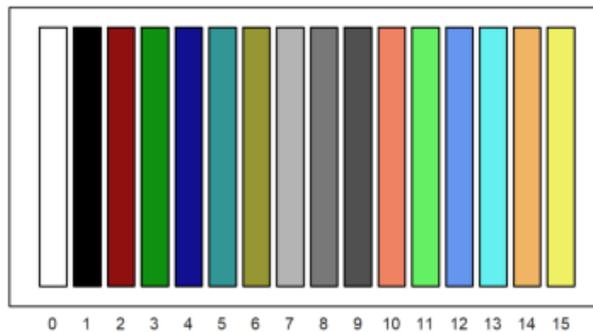
- 0.15: indeks warna default.
- konstanta warna: putih, hitam, merah, hijau, biru, cyan, zaitun, abu-abu muda, abu-abu, abu-abu tua, oranye, hijau muda, pirus, biru muda, oranye terang, kuning
- rgb(merah, hijau, biru): parameter adalah real dalam [0,1].

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



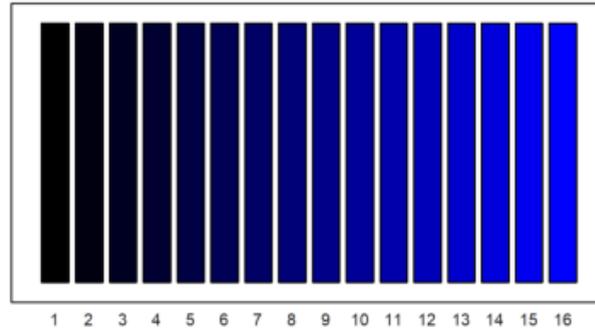
Berikut adalah tampilan warna EMT yang telah ditentukan sebelumnya.

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



Tapi Anda bisa menggunakan warna apa saja.

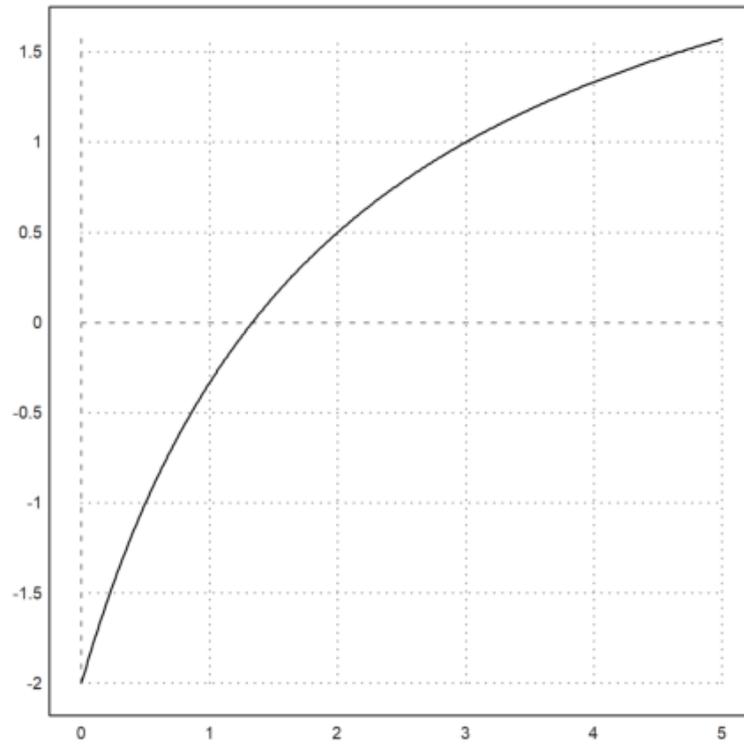
```
>columnsplot(ones(1,16),grid=0,color=rgb(0,0,linspace(0,1,15))):
```



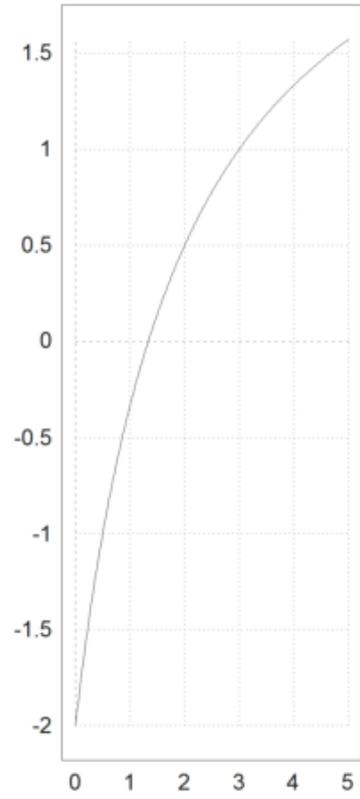
CONTOH SOAL:

$$\frac{3x - 4}{x + 2}$$

```
>reset;  
>function f(x):=(3*x-4)/(x+2)  
>plot2d("f(x)",0,5):
```



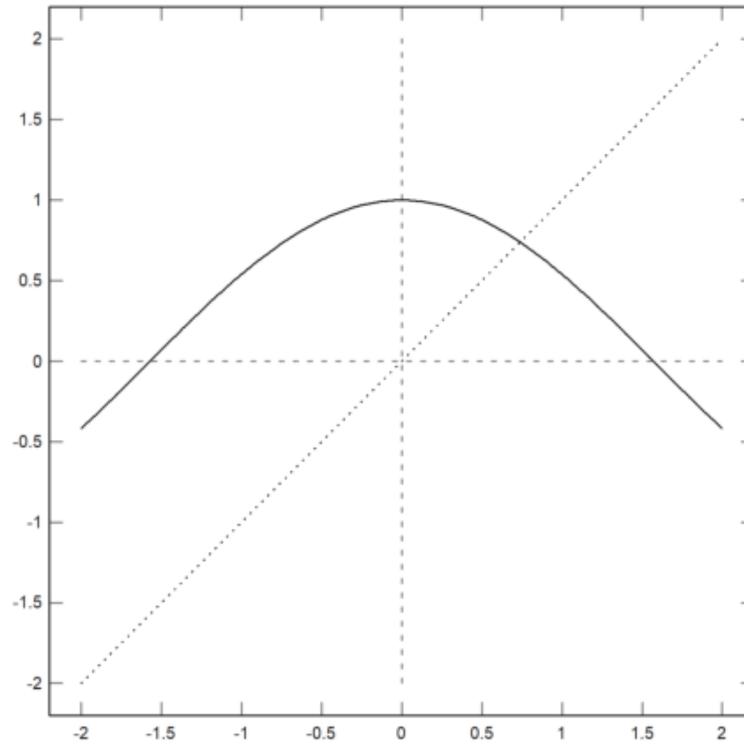
```
>aspect(2,5), plot2d("f(x)",0,5):
```



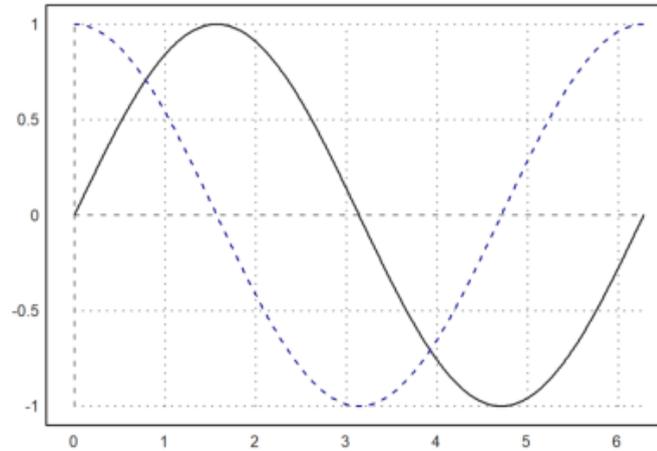
Menggambar Beberapa Kurva pada bidang koordinat yang sama

Plot lebih dari satu fungsi (multiple function) ke dalam satu jendela dapat dilakukan dengan berbagai cara. Salah satu metode menggunakan `>add` untuk beberapa panggilan ke `plot2d` secara keseluruhan, tetapi panggilan pertama. Kami telah menggunakan fitur ini dalam contoh di atas.

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

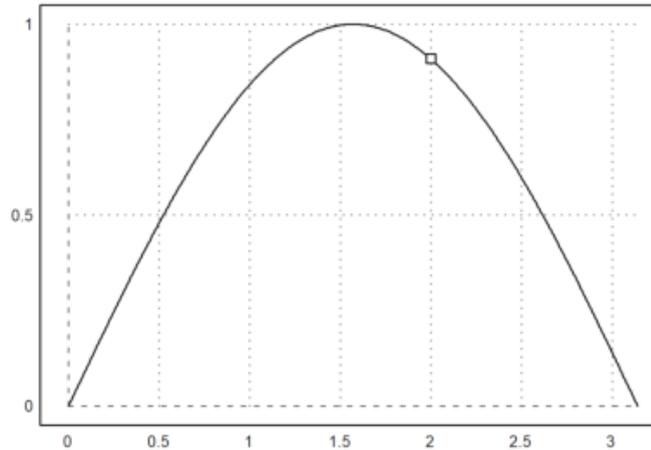


```
>aspect(1.5); plot2d("sin(x)",0,2pi); plot2d("cos(x)",color=blue,style="--",>add):
```



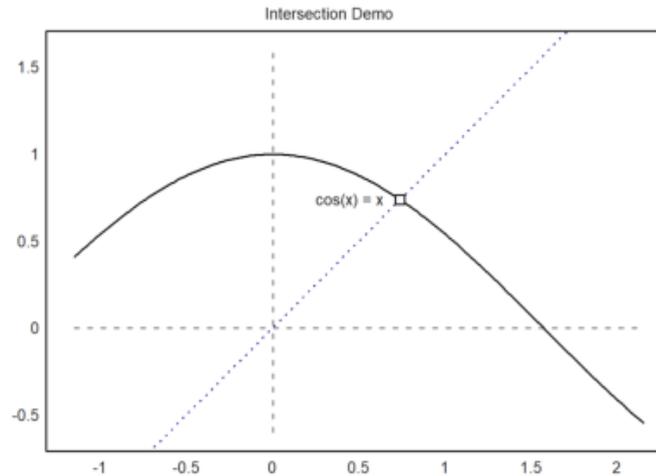
Salah satu kegunaan `>add` adalah untuk menambahkan titik pada kurva.

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



Kami menambahkan titik persimpangan dengan label (pada posisi "cl" untuk kiri tengah), dan memasukkan 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. Kami menghitung ekspansi ini menggunakan Maxima melalui ekspresi simbolis. Ini dilakukan dalam perintah multi-baris berikut dengan tiga panggilan ke `plot2d()`. Yang kedua dan yang ketiga memiliki set flag `>add`, yang membuat plot menggunakan rentang sebelumnya. Kami menambahkan kotak label yang menjelaskan fungsi.

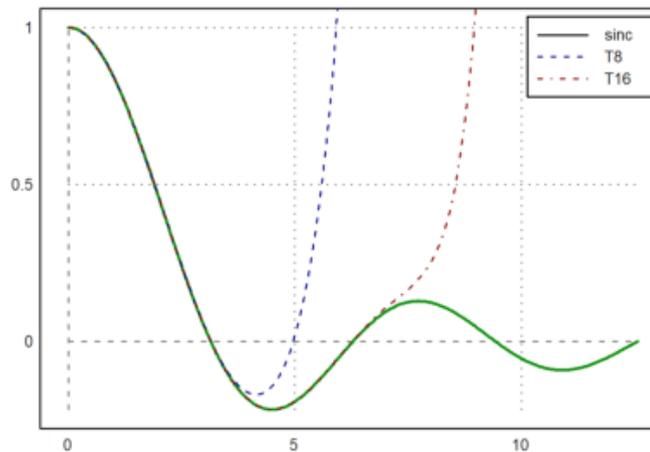
```
>$taylor(sin(x)/x,x,0,4)
```

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

```

>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]):

```



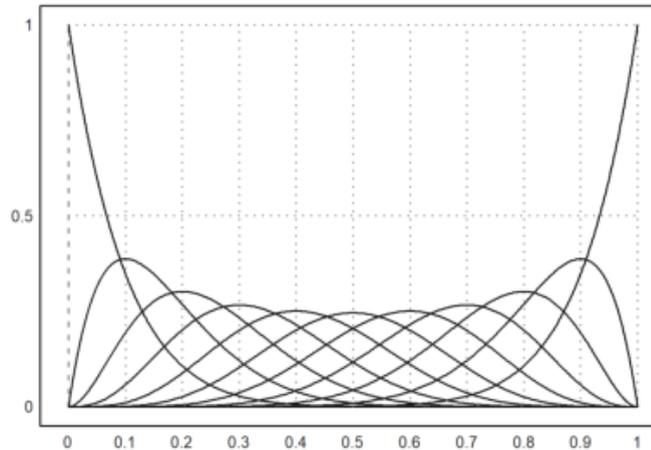
Dalam contoh berikut, kami menghasilkan Bernstein-Polinomial.

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

```

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

```



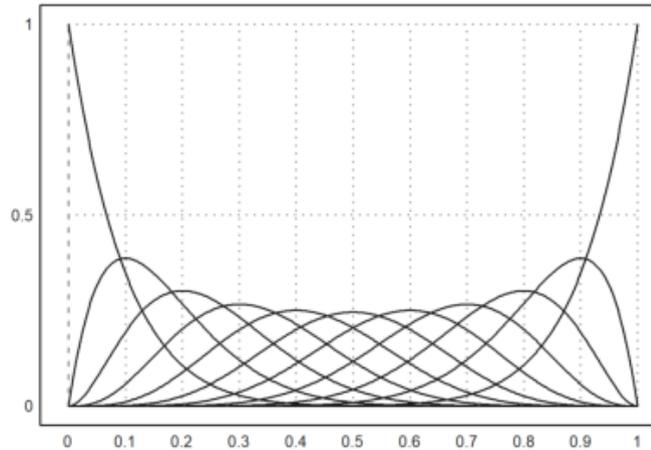
Metode kedua menggunakan pasangan matriks nilai-x dan matriks nilai-y yang berukuran sama.

Kami menghasilkan matriks nilai dengan satu Polinomial Bernstein di setiap baris. Untuk ini, kita cukup menggunakan vektor kolom i . Lihat pengantar tentang bahasa matriks untuk mempelajari lebih detail.

```

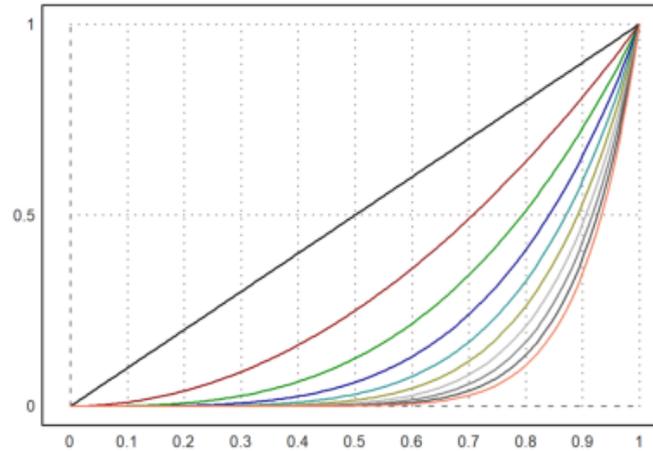
>x=linspace(0,1,500);
>n=10; k=(0:n)'; // n adalah vektor baris, k adalah vektor kolom
>y=bin(n,k)*x^k*(1-x)^(n-k); // maka y adalah sebuah matriks
>plot2d(x,y):

```



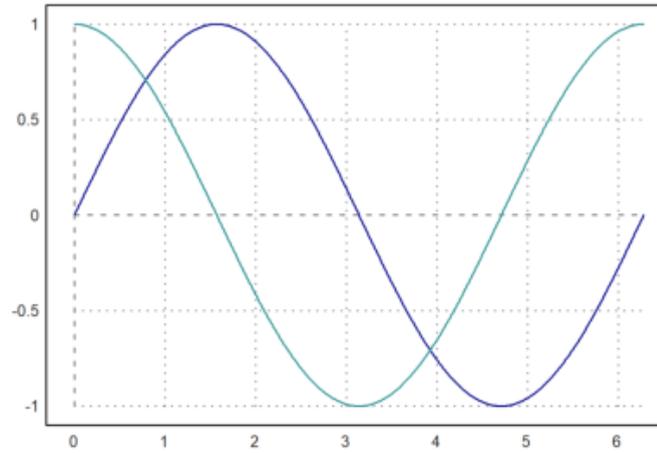
Perhatikan bahwa parameter warna dapat berupa vektor. Kemudian setiap warna digunakan untuk setiap baris matriks.

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

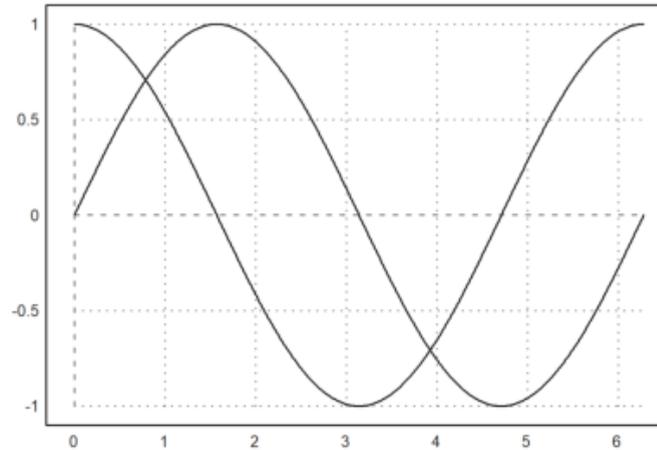


Metode lain adalah menggunakan vektor ekspresi (string). Anda kemudian dapat menggunakan larik warna, larik gaya, dan larik ketebalan dengan panjang yang sama.

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



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



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

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

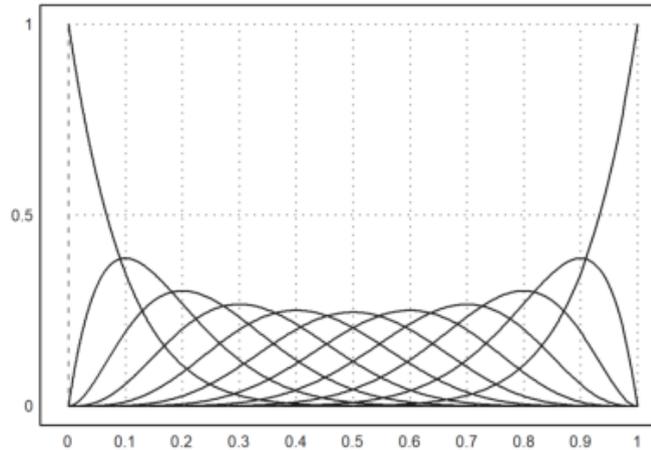
```

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

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

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

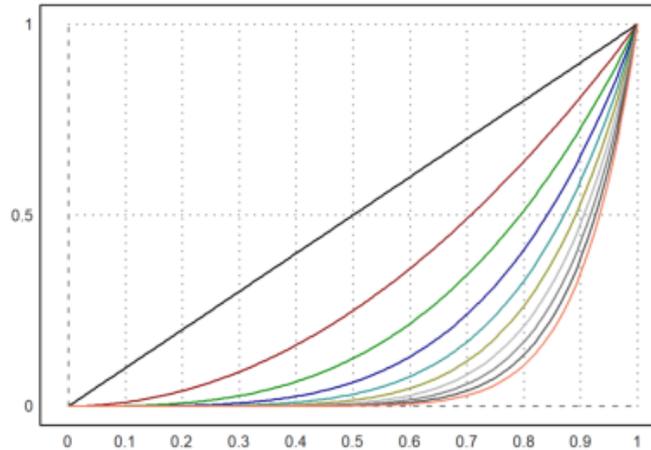


Alternatif lain adalah dengan menggunakan bahasa matriks Euler.

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

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

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

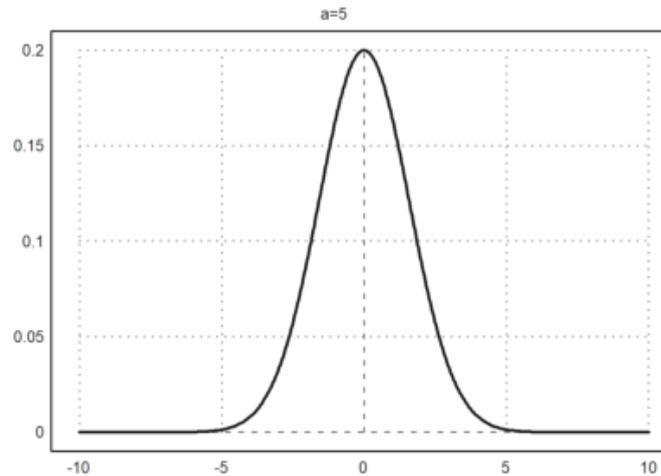


Ekspresi dan fungsi satu baris dapat melihat variabel global.

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

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

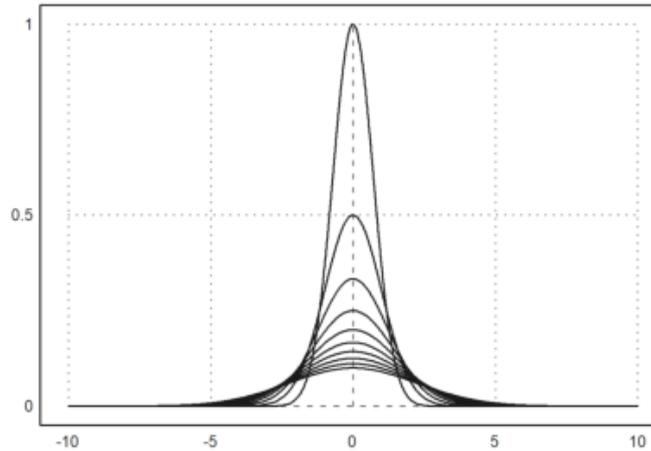
```
>function f(x,a) := 1/a*exp(-x^2/a); ...  
>plot2d("f",-10,10;5,thickness=2,title="a=5"):
```



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

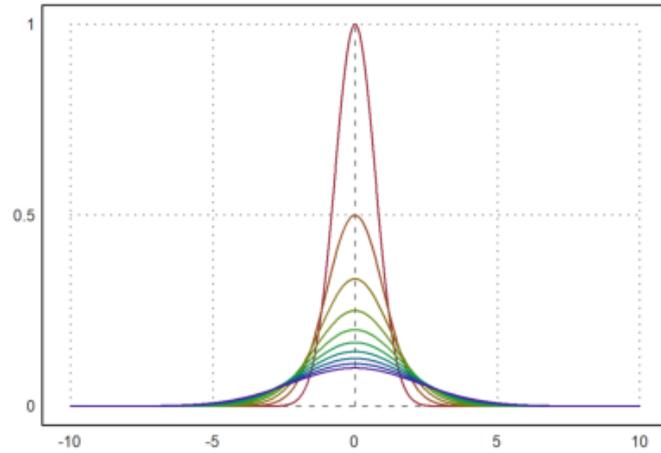
Dalam contoh berikut, kami menggunakan loop untuk memplot beberapa fungsi (lihat tutorial tentang pemrograman untuk loop).

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



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. Klik dua kali pada fungsi `getspectral()` untuk penjelasannya.

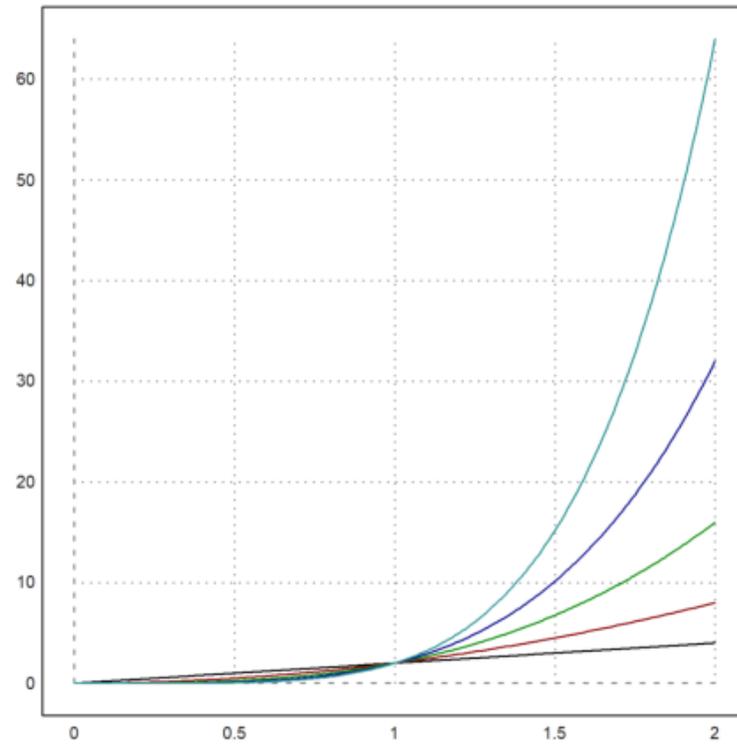
```
>x=-10:0.01:10; a=(1:10)'; plot2d(x,f(x,a),color=getspectral(a/10)):
```



CONTOH SOAL:
Soal 1

$$2x^n, n = 1, 2, 3, 4, 5$$

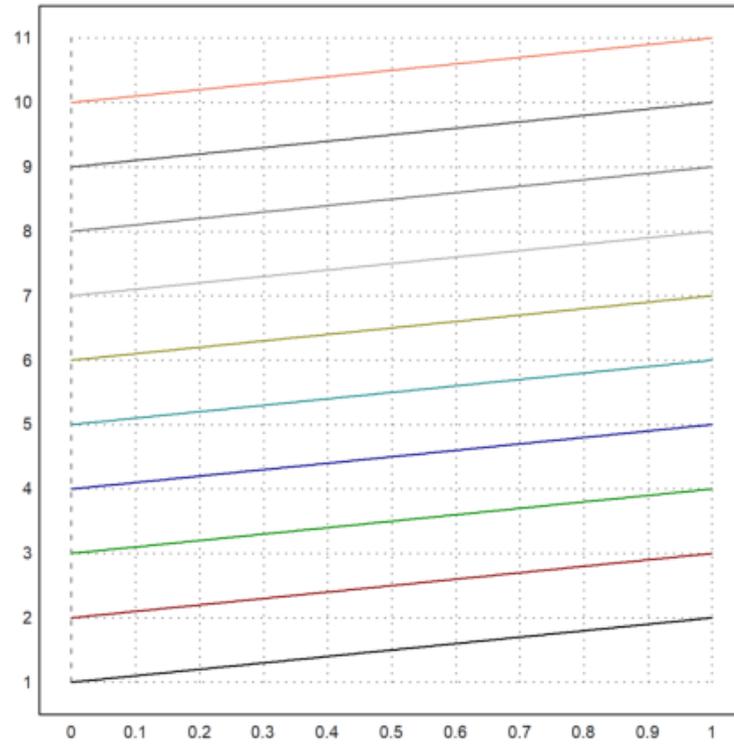
```
>reset;  
>n=(1:5)'; plot2d("2x^n",0,2,color=1:5):
```



Soal 2

$$x + n, n = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$$

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



```
>reset;
```

Contoh soal

Suatu perusahaan yang mempunyai 3 pabrik akan memproduksi 2 jenis produk. Pabrik 1 dapat menghasilkan satu unit produk I selama 1 jam dan produk II selama 2 jam. Pabrik 2 dapat menghasilkan satu unit produk I selama 1 jam dan menghasilkan produk II selama 1 jam . Pabrik 3 dapat menghasilkan satu unit produk I selama 3 jam dan menghasilkan satu unit produk II selama 2 jam. Kapasitas produksi pabrik 1 setiap pekannya adalah dapat beroperasi paling lama 10 jam, pabrik 2 paling lama 6 jam, dan pabrik 3 paling lama 16 jam. Adapun keuntungan produk I per unit adalah 3\$ dan produk II adalah 5\$. Perusahaan tersebut ingin memaksimalkan laba yang diperoleh dengan keterbatasan kapasitas produksi setiap pabriknya. Masalahnya adalah berapa unit masing-masing produk I dan produk II yang harus diproduksi

Kemudian dijadikan model matematika

-fungsi tujuan:

$$f(x, y) = 3x + 5y$$

-Kendala

$$x + 2y$$

$$x + y$$

$$3x + 2y$$

Pabrik 1 dapat beroperasi ≤ 10 jam

Pabrik 2 dapat beroperasi ≤ 6 jam

Pabrik 3 dapat beroperasi ≤ 16 jam

dan diperoleh kendala:

$$x + 2y \leq 10$$

$$x + y \leq 6$$

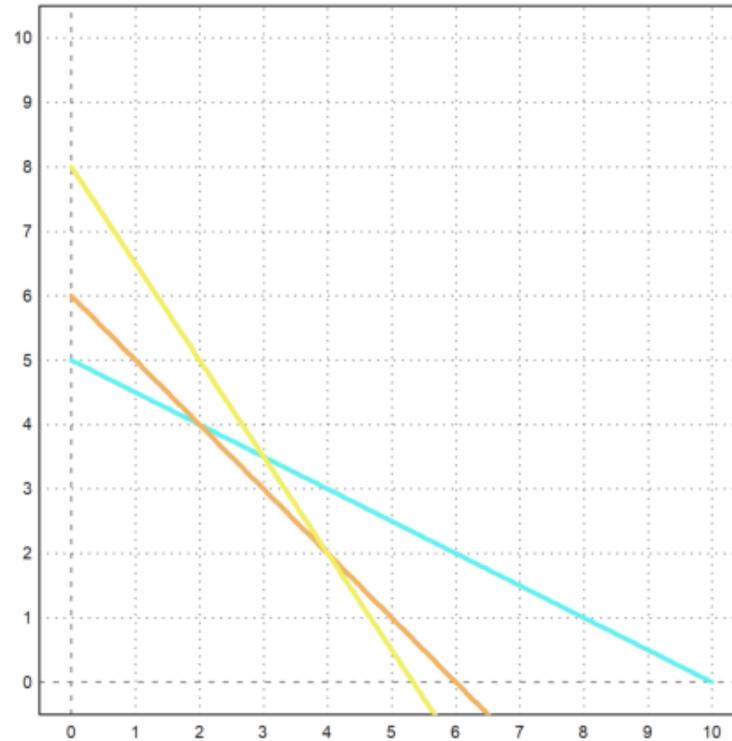
$$3x + 2y \leq 16$$

$$x \geq 0$$

$$y \geq 0$$

Kemudian akan digambar grafik

```
>plot2d(["(10-x)/2", "(6-x)", "(3*x-16)/-2"] ...  
>,0,10,0,10,color=13:15,thickness=3,grid=5,):
```



Mencari perpotongan bisa dengan menggunakan solve
Kemudian kendala dibuat kedalam bentuk matriks koefisien

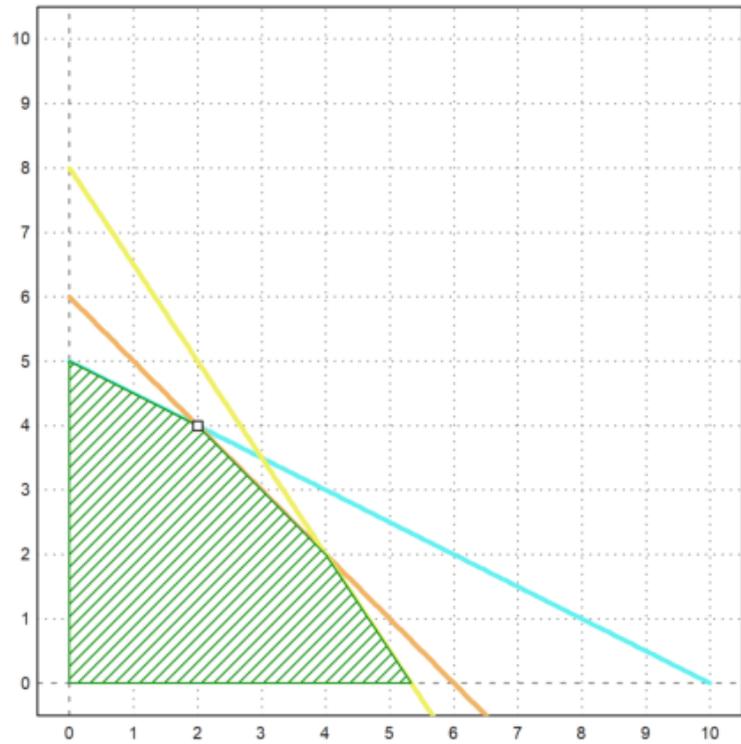
```
>A=[1,2;1,1;3,2]; b=[10;6;16];
```

Metode simpleks untuk maksimisasi atau minimisasi fungsi linier fungsi dengan kendala linier, dan variabel yang dibatasi atau tidak dibatasi variabel. Fungsi ini memanggil algoritme Simpleks bawaan.

```
>x=simplex(A,b,[3,5],>max); fraction x
```

```
2  
4
```

```
>xa=feasibleArea(A,b);  
>plot2d(xa[1],xa[2],>filled,style="/",>add);  
>plot2d(x[1],x[2],>add,>points):
```



Kemudian tinggal di substitusikan saja titik (2,4) ke dalam fungsi tujuan maka ditemukan nilai optimal

$$3x + 5y$$

$$3 * 2 + 5 * 4$$

Jadi maksimal pabrik memperoleh laba adalah 26 dan produk 1 dibuat sebanyak 2 dan produk 2 dibuat sebanyak 4

```
>reset;
```

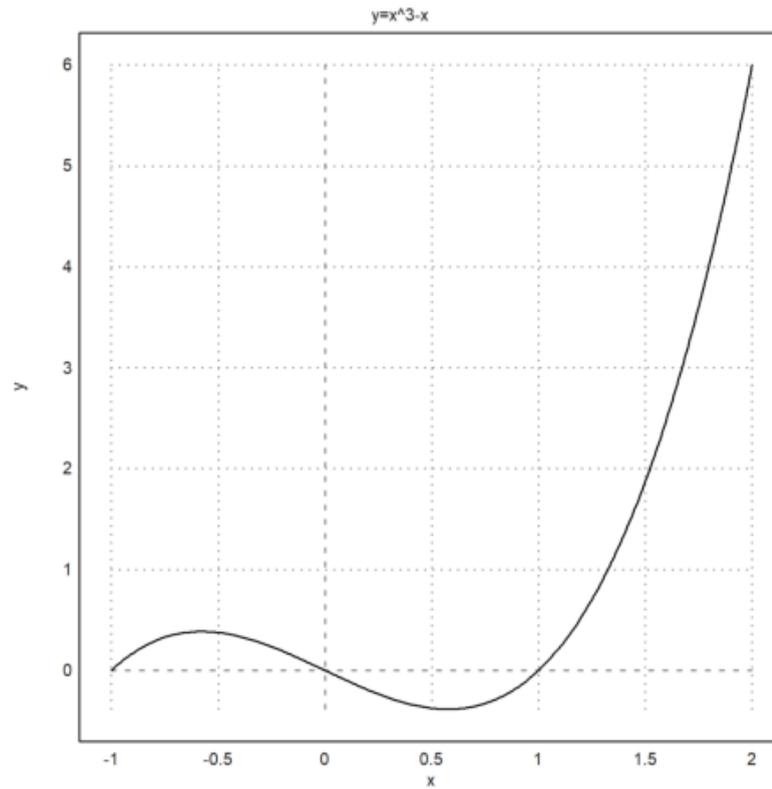
Label Teks

Dekorasi sederhana bisa

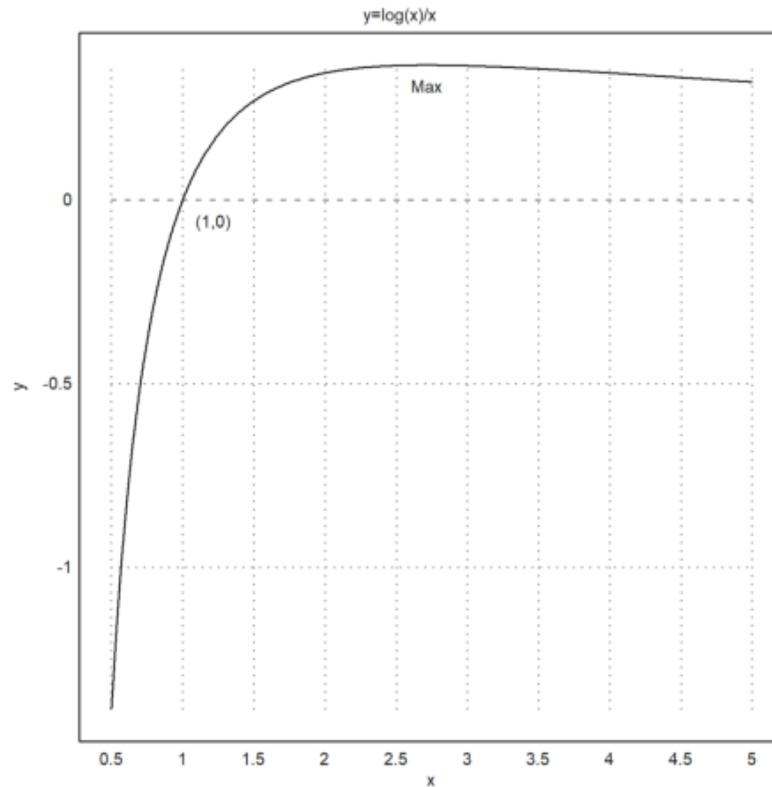
- judul dengan `judul="..."`
- x- dan y-label dengan `xl="..."`, `yl="..."`
- label teks lain dengan `label("...",x,y)`

Perintah label akan memplot ke dalam plot saat ini pada koordinat plot (x,y). Itu bisa mengambil argumen posisi.

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



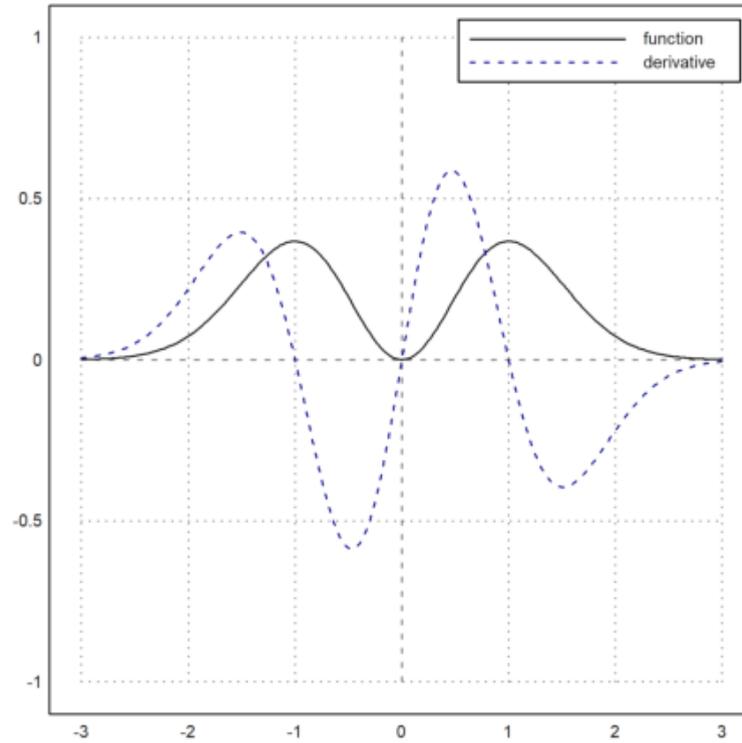
```
>expr := "log(x)/x"; ...  
> plot2d(expr,0.5,5,title="y="+expr,xl="x",yl="y"); ...  
> label("(1,0)",1,0); label("Max",E,expr(E),pos="lc");
```



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

```
>function f(x) &= x^2*exp(-x^2); ...
>plot2d(&f(x),a=-3,b=3,c=-1,d=1); ...
>plot2d(&diff(f(x),x),>add,color=blue,style="--"); ...
```

```
>labelbox(["function","derivative"],styles=["-","--"], ...  
> colors=[black,blue],w=0.4):
```

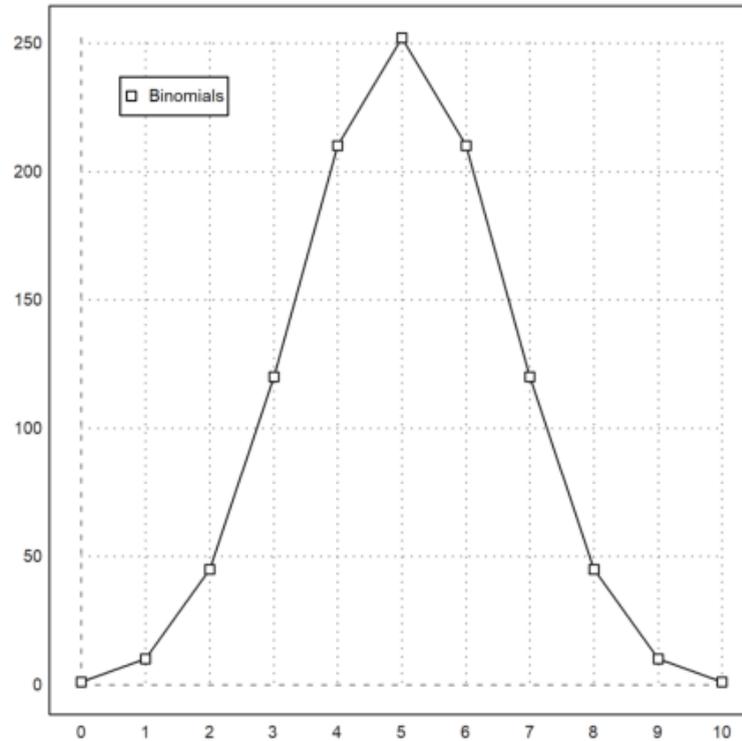


Kotak ditambahkan di kanan atas secara default, tetapi `> kiri` menambatkannya di kiri atas. Anda dapat memindahkannya ke tempat yang Anda suka. Posisi jangkar adalah sudut kanan atas kotak, dan angkanya adalah pecahan dari ukuran jendela grafik. Lebarinya otomatis.

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

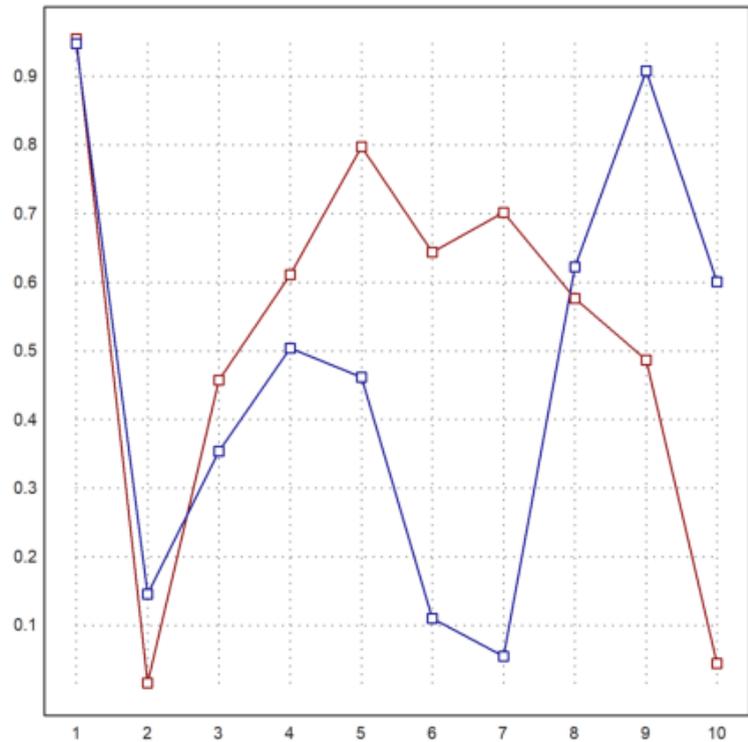
Dalam contoh berikut, hanya ada satu fungsi. Jadi kita bisa menggunakan string sebagai pengganti vektor string. Kami mengatur warna teks menjadi hitam untuk contoh ini.

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



Gaya plot ini juga tersedia di `statplot()`. Seperti di `plot2d()` warna dapat diatur untuk setiap baris plot. Ada lebih banyak plot khusus untuk keperluan statistik (lihat tutorial tentang statistik).

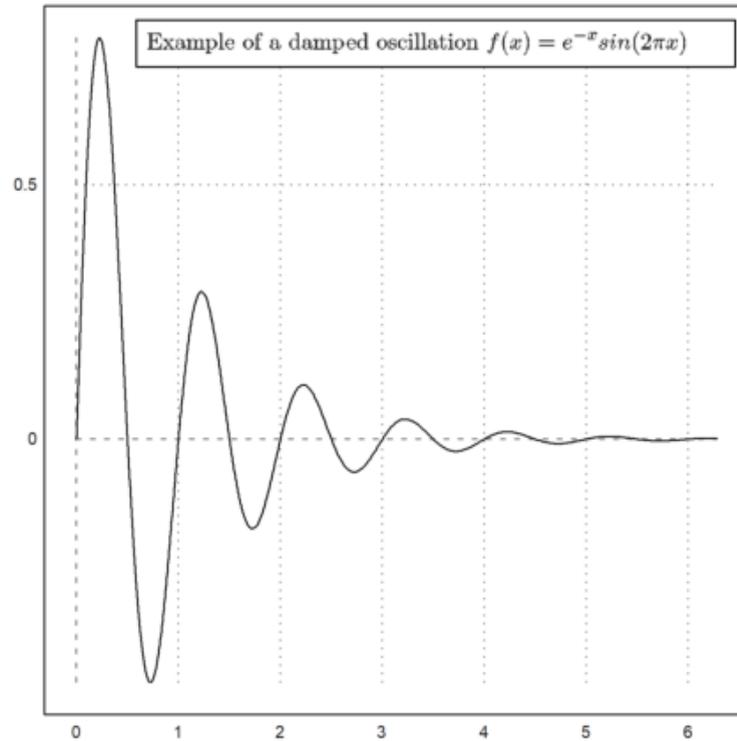
```
>statplot(1:10,random(2,10),color=[red,blue]):
```



Fitur serupa adalah fungsi `textbox()`.

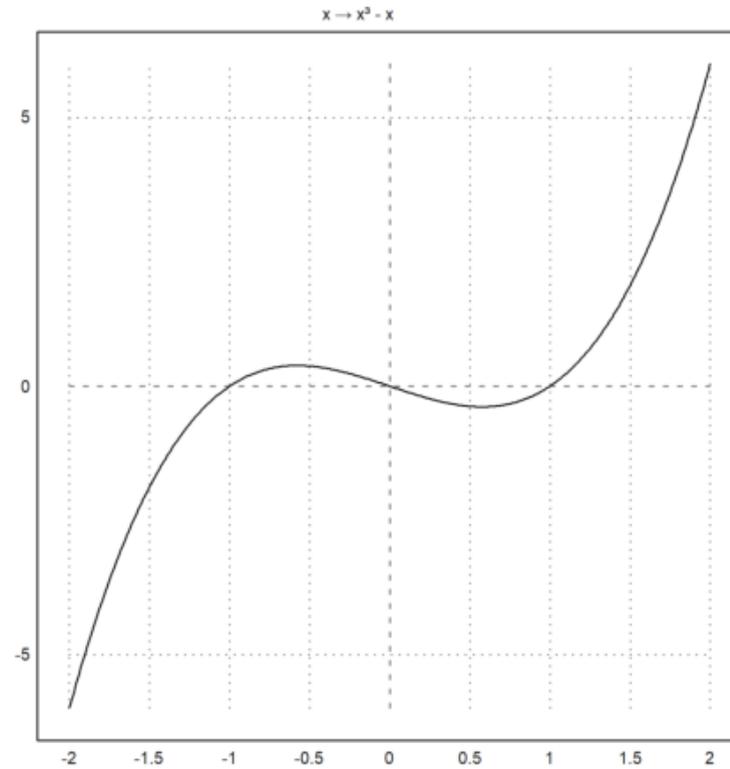
Lebar secara default adalah lebar maksimal dari baris teks. Tapi itu bisa diatur oleh pengguna juga.

```
>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):
```



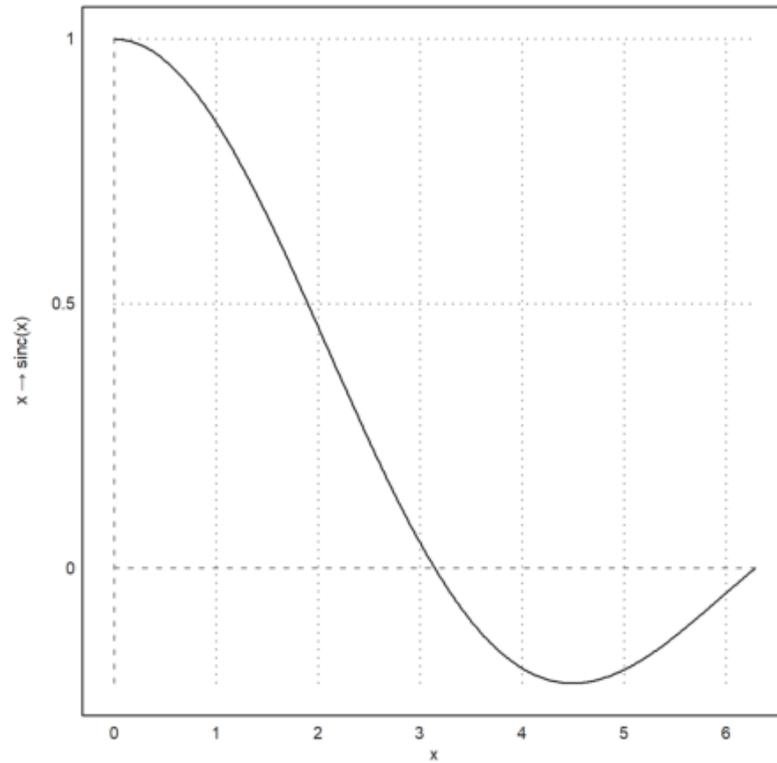
Label teks, judul, kotak label, dan teks lainnya dapat berisi string Unicode (lihat sintaks EMT untuk mengetahui lebih lanjut tentang string Unicode).

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



Label pada sumbu x dan y bisa vertikal, begitu juga sumbunya.

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

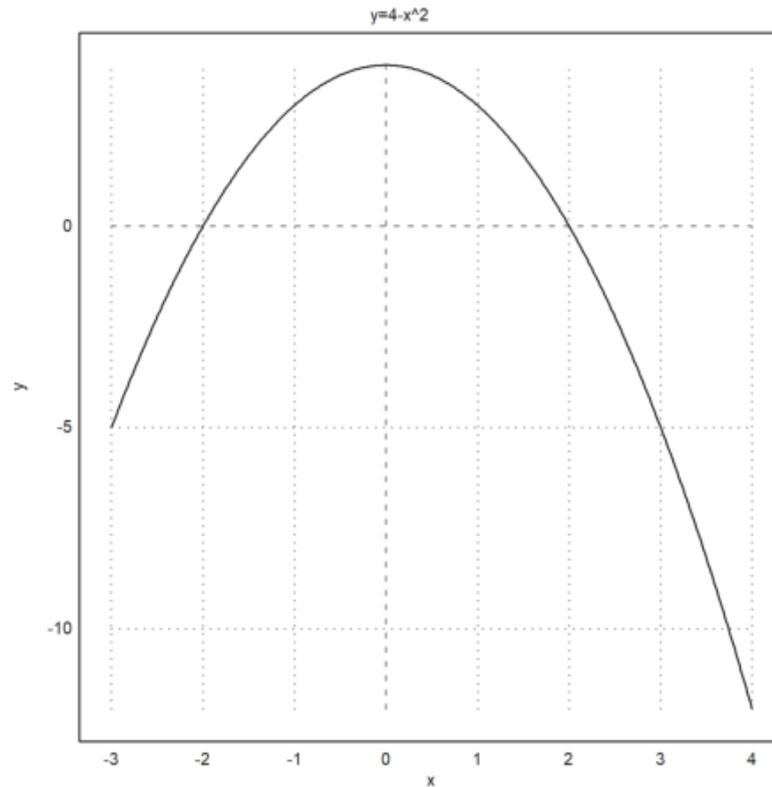


Contoh penggunaan label pada sumbu koordinat

Akan digambarkan kurva dengan fungsi:

$$y = 4 - x^2$$

```
>aspect(1); plot2d("4-x^2",-3,4,title="y=4-x^2",yl="y",xl="x"):
```



Kurva diatas merupakan grafik dari fungsi $y=f(x)=4-x^2$. Pada plot diatas menggunakan perintah `xl` dan `yl`, sumbu x diberi nama "x" dan sumbu y diberi nama "y". Plot diatas juga diberi judul dengan fungsinya tapi judul tersebut belum menggunakan latex.

Selain memberikan label pada sumbu koordinatnya, grafik tersebut juga memiliki batasan x yang diatur dari -3 sampai 4.

Contoh penggunaan label kurva

Gambarkan grafik dari fungsi

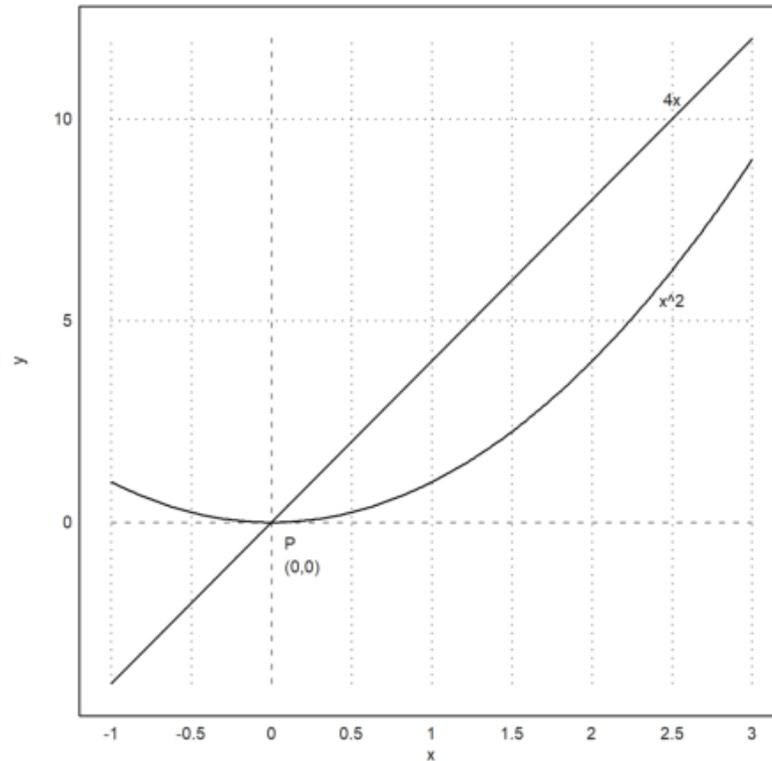
$$y = 4x$$

dan

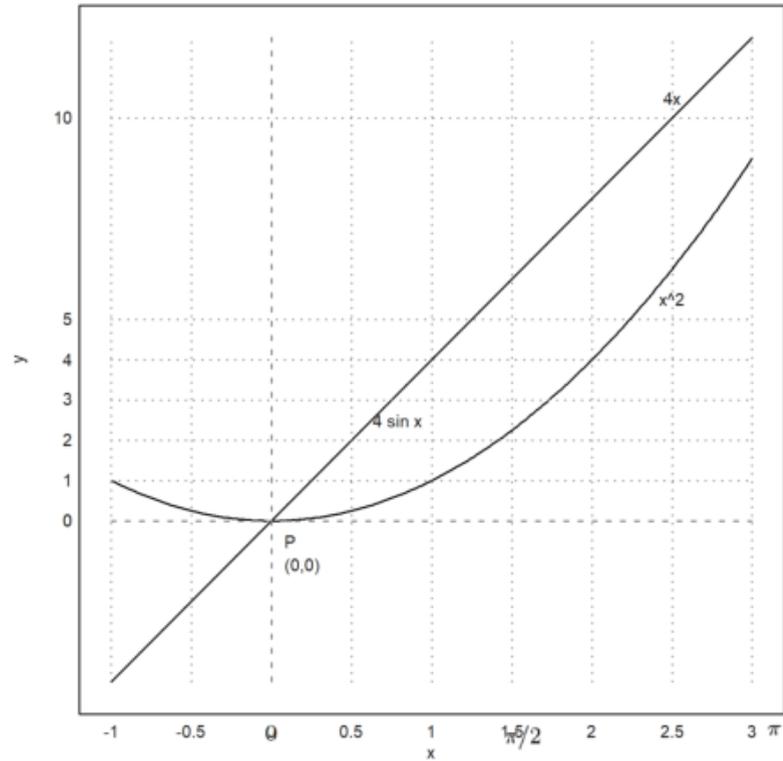
$$y = x^2$$

Dengan interval $[-1,3]$, tentukan titik perpotongannya.

```
>reset;  
>plot2d(["4*x", "x^2"], -1, 3, xl="x", yl="y"); label(["P", "(0,0)"], 0, 0); ...  
>label("4x", 2.5, 10, pos="uc"); label("x^2", 2.5, 6, pos="lc");
```



```
>label("4 sin x",pi/4,3,pos="lc"); ygrid(0:4); xgrid([0:1]*pi,<ticks); ...  
>xtick([0,pi/2,pi],["0","\pi/2","\pi"],>latex):
```



Berikut adalah plot 2d dari fungsi $4 \sin x$. Telah diberikan judul plot, label kurva menggunakan perintah `title=“...”` dan `label(“...”,x,y)` dan mengubah nilai pada sumbu x yang awalnya menggunakan nilai (angka) karena fungsinya merupakan sin maka diubah menggunakan pi.

Contoh penggunaan keterangan kurva (legend)

Akan dibuat grafik dengan fungsi

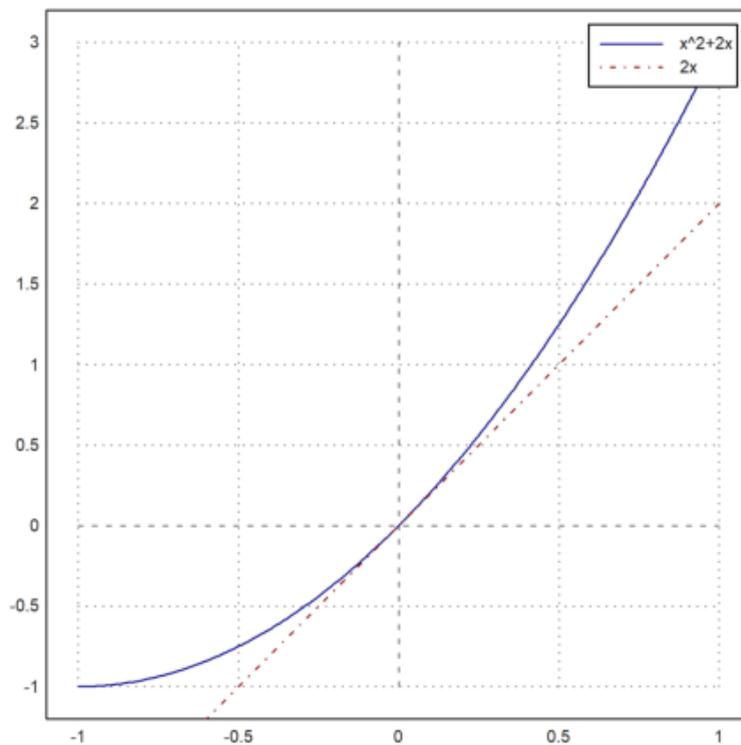
$$f(x) = x^2 + 2x$$

dan

$$g(x) = 2x$$

lalu diberikan keterangan kurjanya (legend)

```
>reset;  
>plot2d("x^2+2*x",-1,1,color=blue); plot2d("2*x",color=red,style="-.",>add);  
>labelbox(["x^2+2x","2x"],colors=[blue,red],styles=["-", "-."]):
```



Plot 2d diatas telah ditambahkan keterangan kurva (legend) menggunakan perintah label(box). Setiap kurva juga memiliki warna yang berbeda sehingga menambah kemudahan untuk menjelaskan fungsi dari masing-masing kurva.

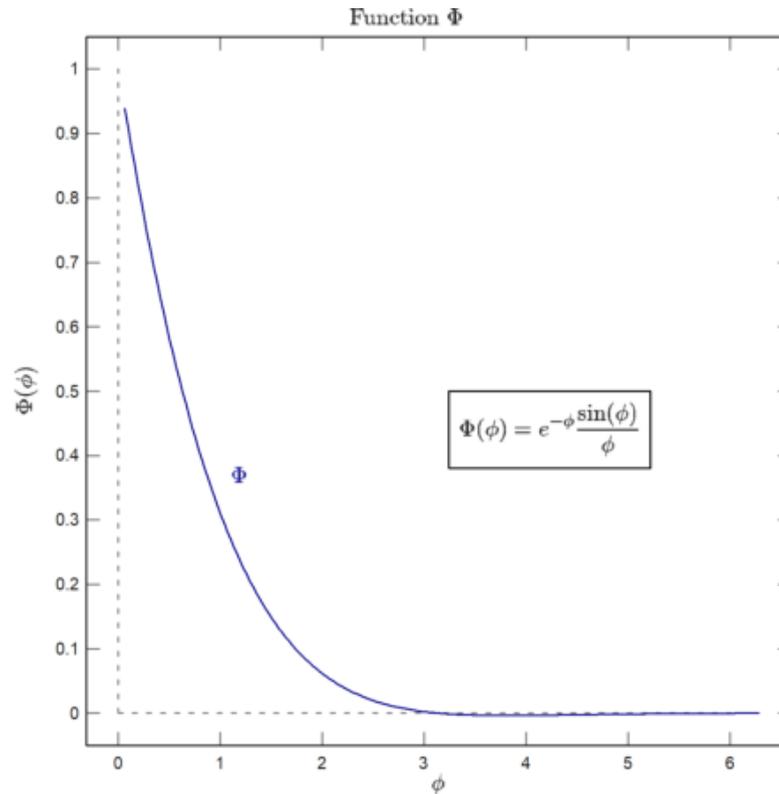
LaTeX

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

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

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

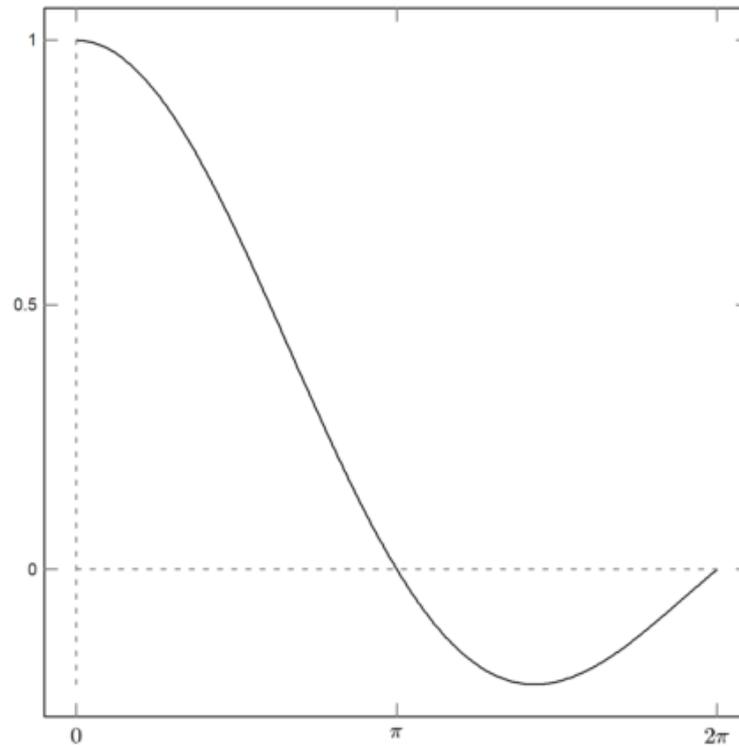
```
>plot2d("exp(-x)*sin(x)/x",a=0,b=2pi,c=0,d=1,grid=6,color=blue, ...
> title=latex("\text{Function } \$\Phi\$"), ...
> xl=latex("\phi"),yl=latex("\Phi(\phi)"); ...
>textbox( ...
> latex("\Phi(\phi) = e^{-\phi} \frac{\sin(\phi)}{\phi}"),x=0.8,y=0.5); ...
>label(latex("\Phi",color=blue),1,0.4):
```



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

Cara termudah adalah dengan membuat plot kosong dengan bingkai menggunakan `grid=4`, lalu menambahkan grid dengan `ygrid()` dan `xgrid()`. Dalam contoh berikut, kami menggunakan tiga string LaTeX untuk label pada sumbu x dengan `xtick()`.

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



Tentu saja, fungsi juga dapat digunakan.

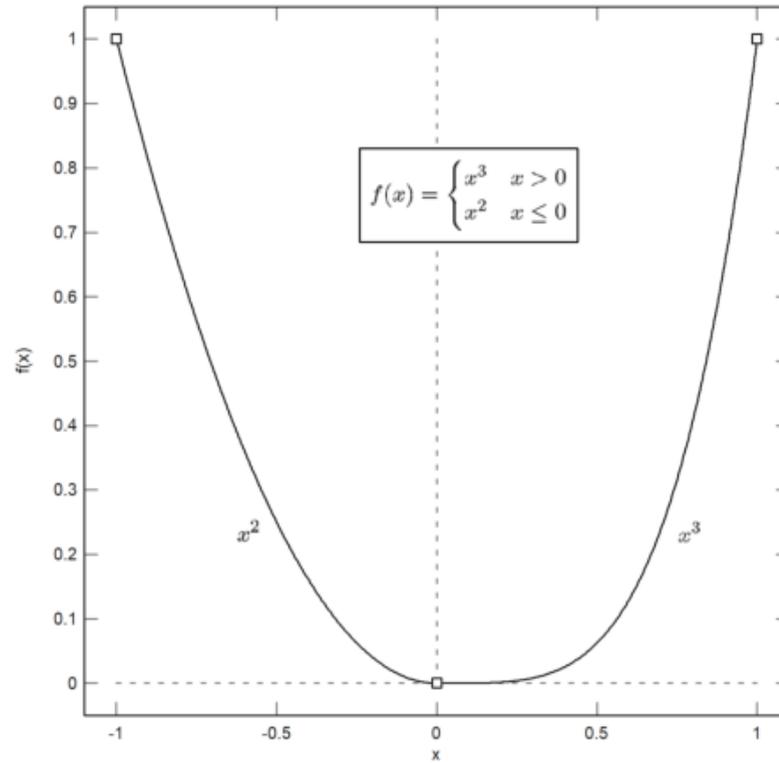
```
>function map f(x) ...
```

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

Parameter "peta" membantu menggunakan fungsi untuk vektor. Untuk lot, itu tidak perlu. Tetapi untuk menunjukkan vektorisasi itu berguna, kami menambahkan beberapa poin kunci ke plot di $x=-1$, $x=0$ dan $x=1$.

Pada plot berikut, kami juga memasukkan beberapa kode LaTeX. Kami menggunakannya untuk label dan kotak teks. Tentu saja, Anda hanya akan dapat menggunakan aTeX 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

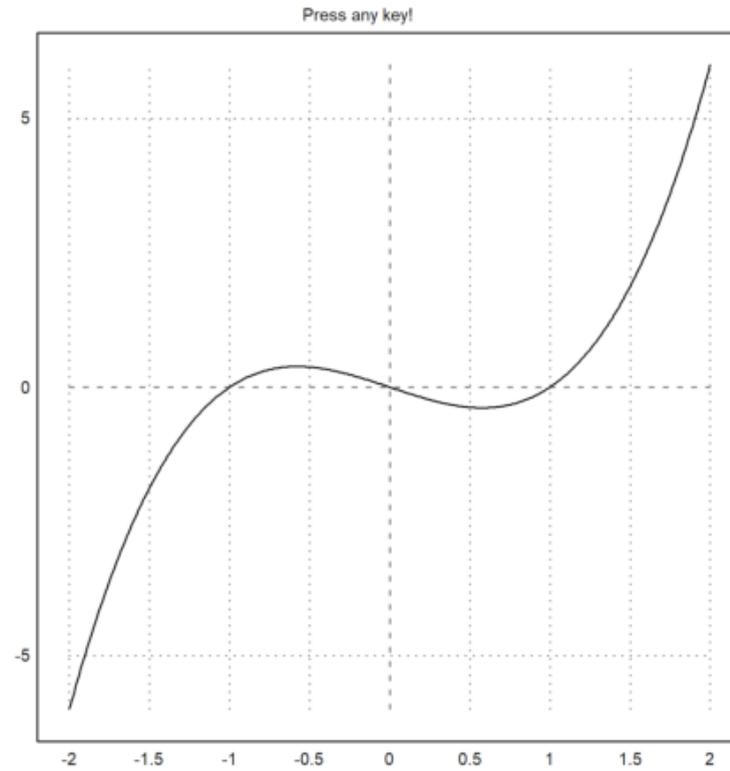
Saat memplot fungsi atau ekspresi, parameter `>user` memungkinkan pengguna untuk memperbesar dan menggeser plot dengan tombol kursor atau mouse. Pengguna dapat

- perbesar dengan `+` atau `-`
- pindahkan plot dengan tombol kursor
- pilih jendela plot dengan mouse
- atur ulang tampilan dengan spasi
- keluar dengan kembali

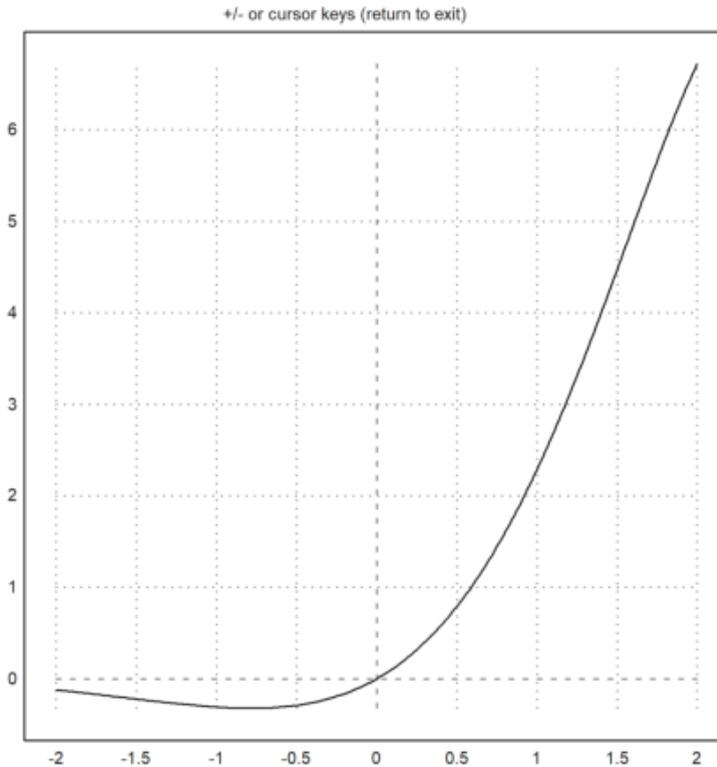
Tombol spasi akan mengatur ulang plot ke jendela plot asli.

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

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



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



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

Fungsi bawaan `mousedrag()` menunggu event mouse atau keyboard. Ini melaporkan mouse ke bawah, mouse dipindahkan atau mouse ke atas, dan penekanan tombol. Fungsi `dragpoints()` memanfaatkan ini, dan memungkinkan pengguna menyeret titik mana pun dalam plot.

Kita membutuhkan fungsi `plot` terlebih dahulu. Sebagai contoh, kita interpolasi dalam 5 titik dengan polinomial. Fungsi harus diplot ke area plot tetap.

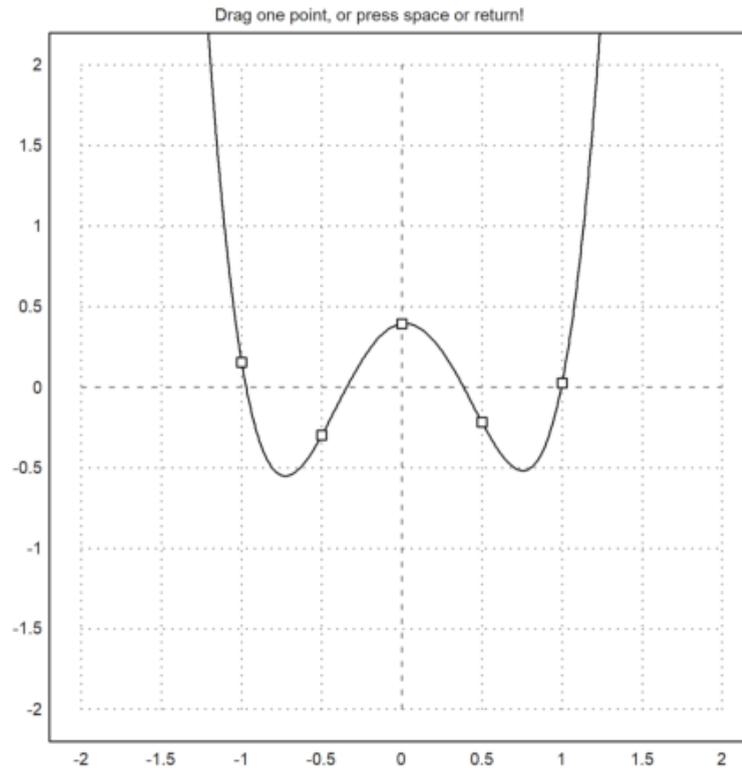
```
>function plotf(xp,yp,select) ...
```

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

Sekarang kita menghasilkan beberapa nilai acak, dan membiarkan pengguna menyeret poin.

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



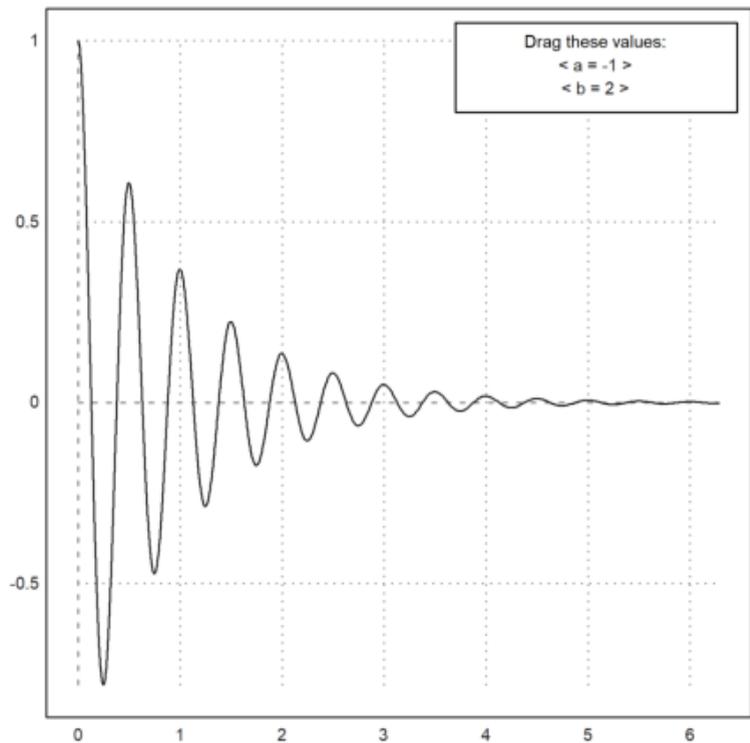
Ada juga fungsi, yang memplot fungsi lain tergantung pada vektor parameter, dan memungkinkan pengguna menyesuaikan parameter ini.

Pertama kita membutuhkan fungsi plot.

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

Kemudian kita membutuhkan nama untuk parameter, nilai awal dan matriks rentang $n \times 2$, opsional baris judul. Ada slider interaktif, yang dapat mengatur nilai oleh pengguna. Fungsi `dragvalues()` menyediakan ini.

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



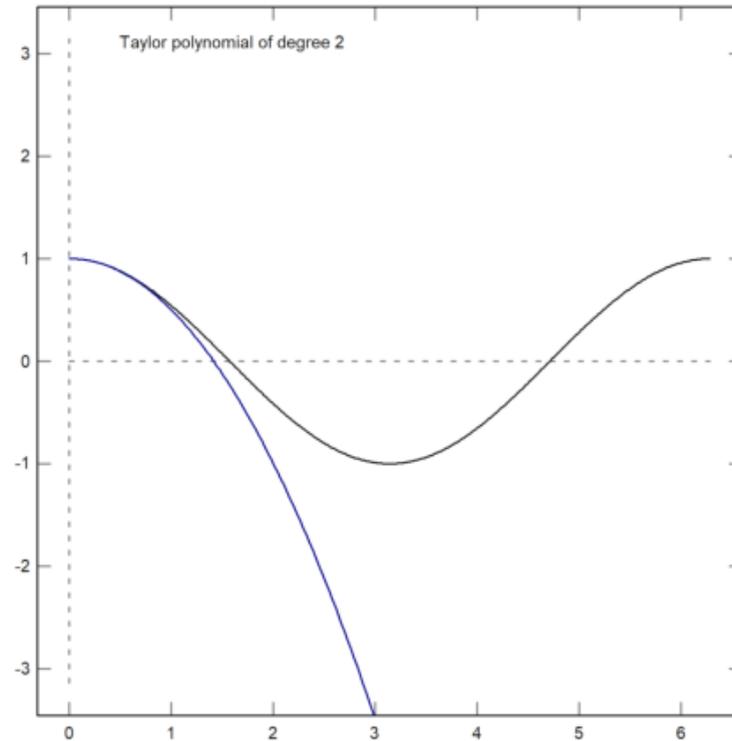
Dimungkinkan untuk membatasi nilai yang diseret ke bilangan bulat. Sebagai contoh, kita menulis fungsi plot, yang memplot polinomial Taylor derajat n ke fungsi kosinus.

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

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

Sekarang kami mengizinkan derajat n bervariasi dari 0 hingga 20 dalam 20 pemberhentian. Hasil drag-values() digunakan untuk memplot sketsa dengan n ini, dan untuk memasukkan plot ke dalam buku catatan.

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



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

```
>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

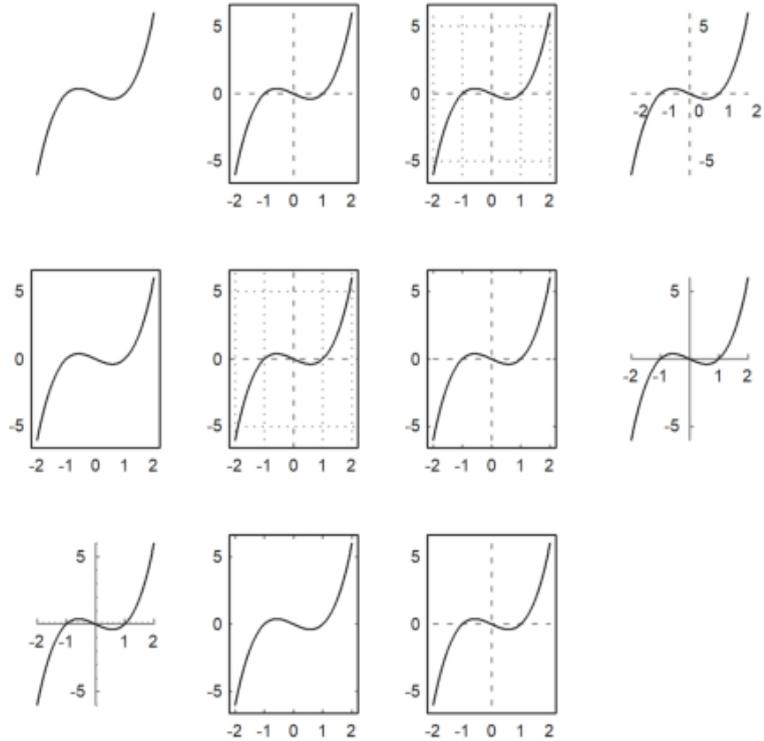
Secara default, EMT menghitung tick sumbu otomatis dan menambahkan label ke setiap tick. Ini dapat diubah dengan parameter grid. Gaya default sumbu dan label dapat dimodifikasi. Selain itu, label dan judul dapat ditambahkan secara manual. Untuk mengatur ulang ke gaya default, gunakan reset().

```
>aspect();
>figure(3,4); ...
> figure(1); plot2d("x^3-x",grid=0); ... // 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 centang 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 centang, hanya label
```

```

> figure(6); plot2d("x^3-x",grid=5); ... // default, tetapi tanpa margin
> figure(7); plot2d("x^3-x",grid=6); ... // hanya sumbu
> figure(8); plot2d("x^3-x",grid=7); ... // hanya sumbu, tanda centang pada sumbu
> figure(9); plot2d("x^3-x",grid=8); ... // hanya sumbu, tanda centang yang lebih halus pada sumbu
> figure(10); plot2d("x^3-x",grid=9); ... // default, tanda centang kecil di dalam
> figure(11); plot2d("x^3-x",grid=10); ...// tanpa tanda centang, hanya sumbu
> figure(0):

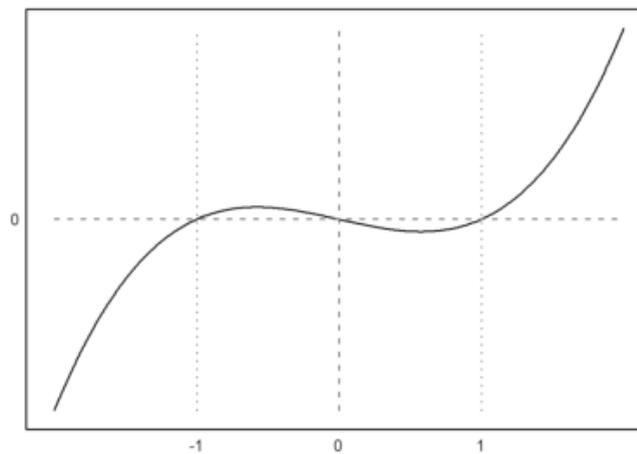
```



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

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

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

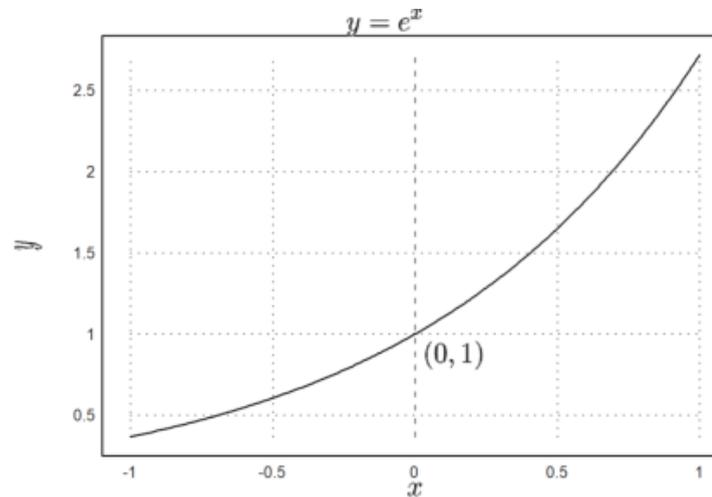


Untuk judul plot dan label sumbu, lihat contoh berikut.

```

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

```

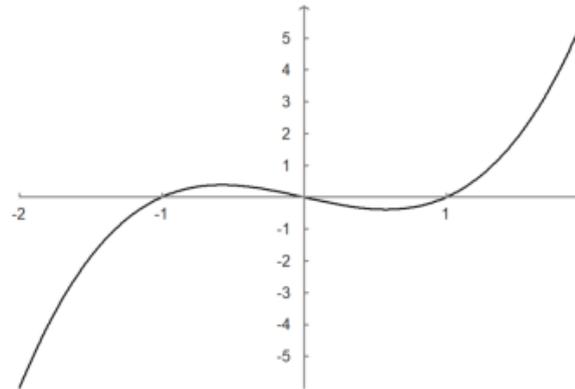


Sumbu dapat digambar secara terpisah dengan `xaxis()` dan `yaxis()`.

```

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

```

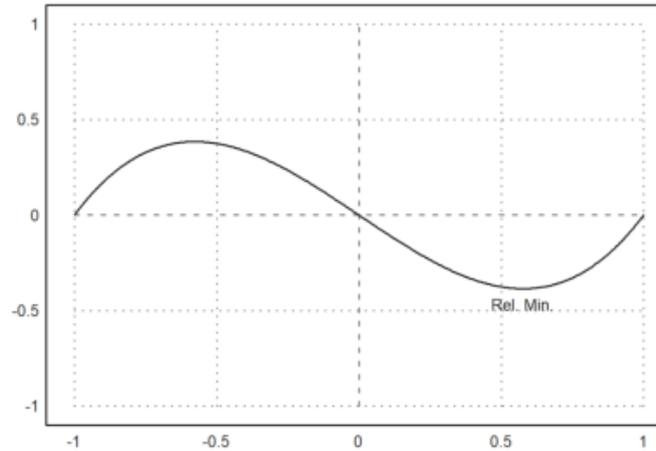


Teks pada plot dapat diatur dengan `label()`. Dalam contoh berikut, "lc" berarti tengah bawah. Ini mengatur posisi label relatif terhadap koordinat plot.

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

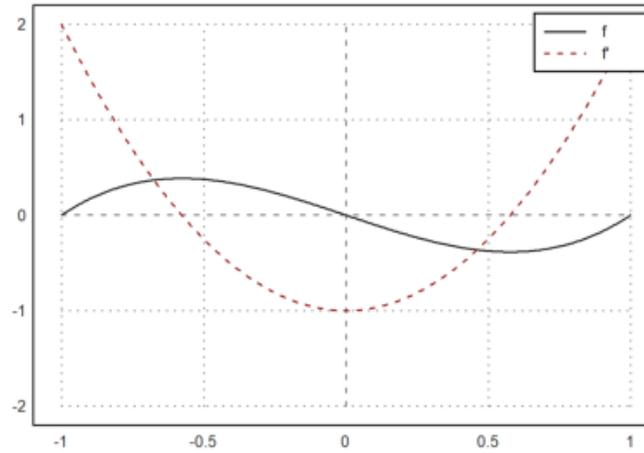
```
3  
x - x
```

```
>plot2d(f,-1,1,>square);  
>x0=fmin(f,0,1); // menghitung titik minimum  
>label("Rel. Min.",x0,f(x0),pos="lc"): // menambahkan label di sana
```

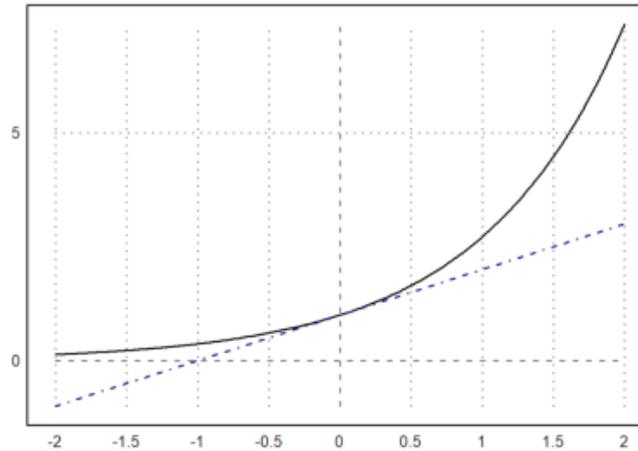


Ada juga kotak teks.

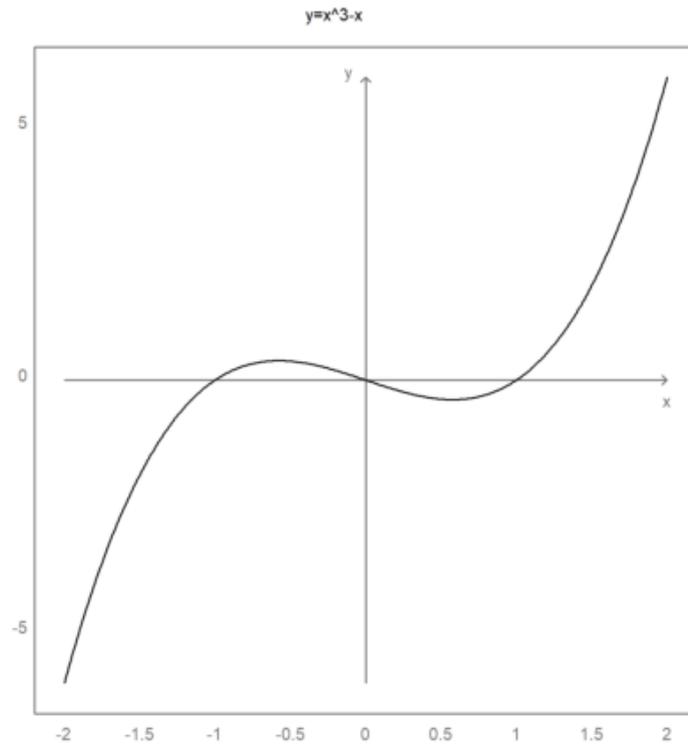
```
>plot2d(&f(x),-1,1,-2,2); // fungsi  
>plot2d(&diff(f(x),x),>add,style="--",color=red); // turunan  
>labelbox(["f","f'"],["-","--"],[black,red]): // kotak label
```



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



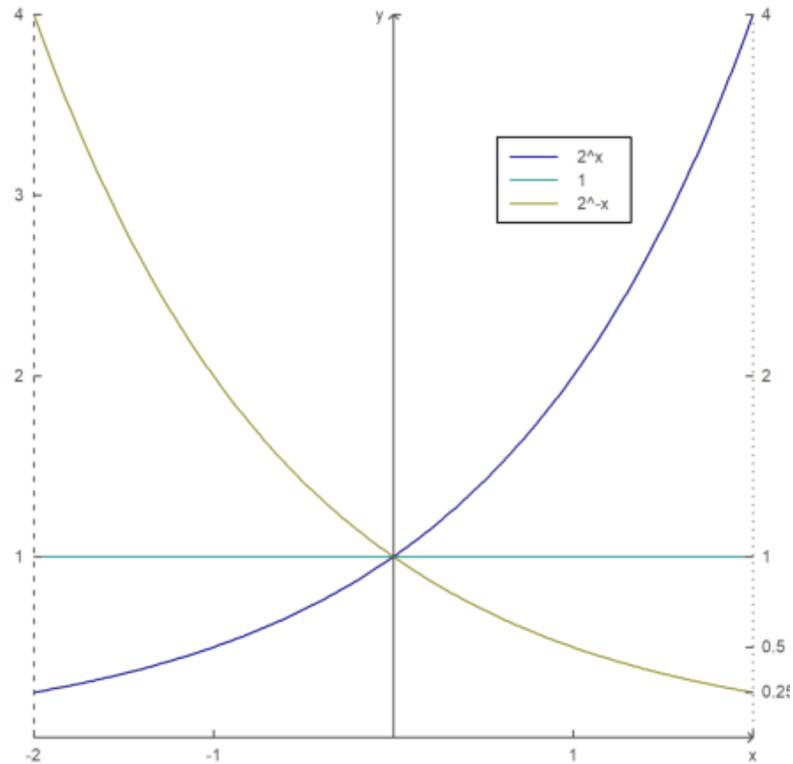
```
> gridstyle("->", color=gray, textcolor=gray, framecolor=gray); ...  
> plot2d("x^3-x", grid=1); ...  
> settitle("y=x^3-x", color=black); ...  
> label("x", 2, 0, pos="bc", color=gray); ...  
> label("y", 0, 6, pos="cl", color=gray); ...  
> reset():
```



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

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

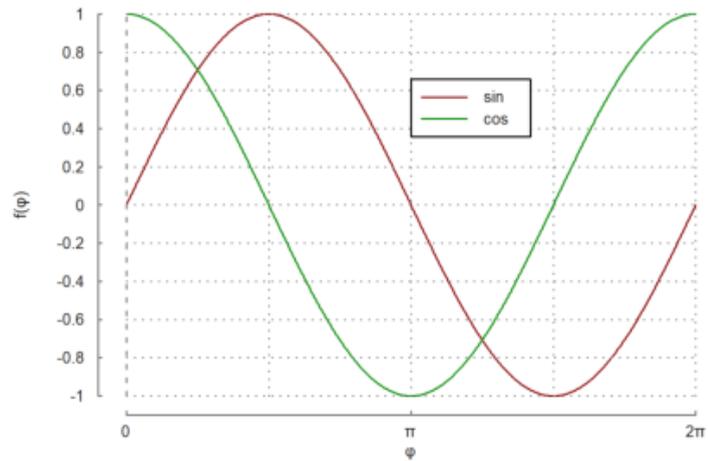
```
>fullwindow; ...
> gridstyle(color=darkgray,textcolor=darkgray); ...
> plot2d(["2^x","1","2^(-x)"],a=-2,b=2,c=0,d=4,<grid,color=4:6,<frame); ...
> xaxis(0,-2:1,style="->"); xaxis(0,2,"x",<axis); ...
> yaxis(0,4,"y",style="->"); ...
> yaxis(-2,1:4,>left); ...
> yaxis(2,2^(-2:2),style=".",<left); ...
> labelbox(["2^x","1","2^-x"],colors=4:6,x=0.8,y=0.2); ...
> reset:
```



Berikut adalah contoh lain, di mana string Unicode digunakan dan sumbu di luar area plot.

```
>aspect(1.5);
>plot2d(["sin(x)", "cos(x)"], 0, 2pi, color=[red, green], <grid, <frame); ...
> xaxis(-1.1, (0:2)*pi, xt=["0", u"&pi;", u"2&pi;"], style="-", >ticks, >zero); ...
> xgrid((0:0.5:2)*pi, <ticks); ...
```

```
> yaxis(-0.1*pi,-1:0.2:1,style="-",>zero,>grid); ...  
> labelbox(["sin","cos"],colors=[red,green],x=0.5,y=0.2,>left); ...  
> xlabel(u"&phi;"); ylabel(u"f(&phi;)"):
```



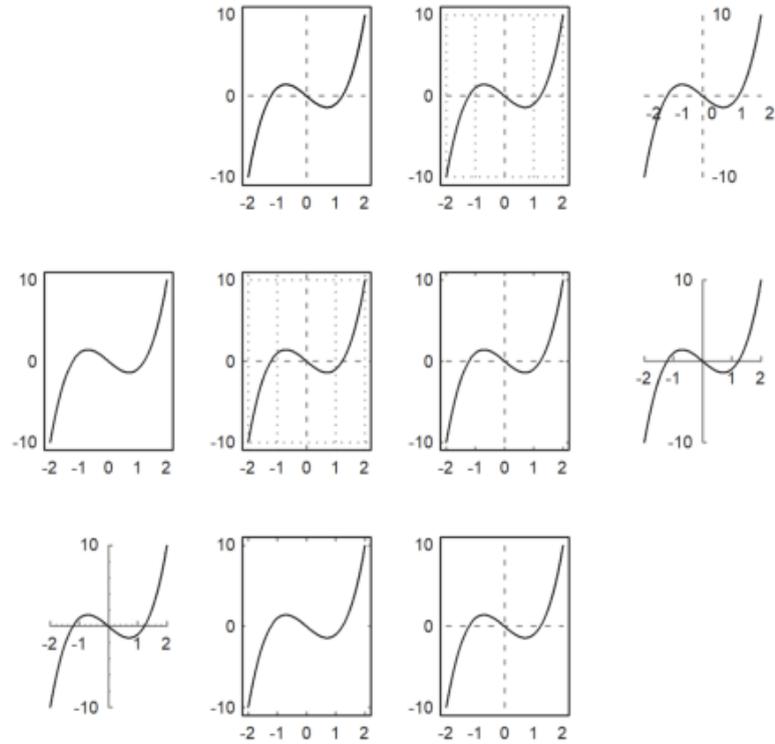
CONTOH SOAL:

Soal 1

Gambarkan plot fungsi berikut :

$$g(x) = 2x^3 - 3x$$

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



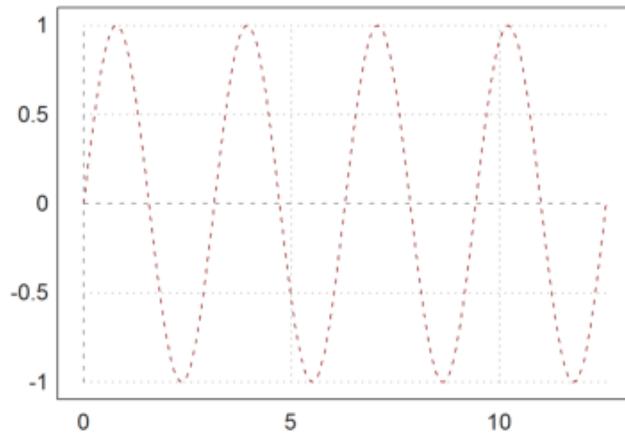
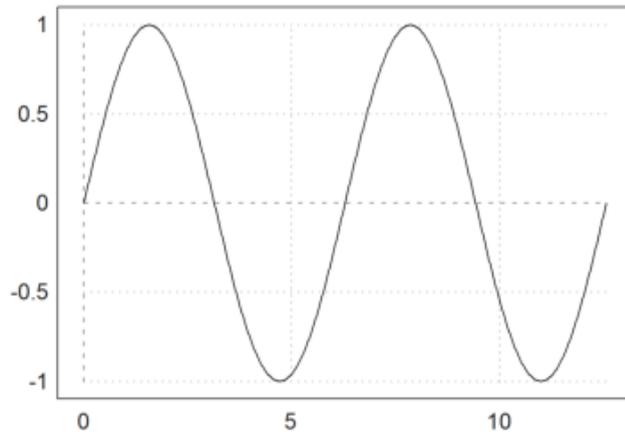
Soal 2

```

>reset;
>aspect(2,3);
>figure(2,1);...
>figure(1); plot2d("sin(x)",0,4pi, grid=2);...

```

```
>figure(2); plot2d("sin(2x)",0,4pi, grid=2, color=red, style="--",);  
>figure(0):
```



Merencanakan Data 2D

Jika x dan y adalah vektor data, data ini akan digunakan sebagai koordinat x dan y dari suatu kurva. Dalam hal ini, a , b , c , dan d , atau radius r dapat ditentukan, atau jendela plot akan menyesuaikan secara otomatis dengan data. Atau, `>persegi` dapat diatur untuk menjaga rasio aspek persegi.

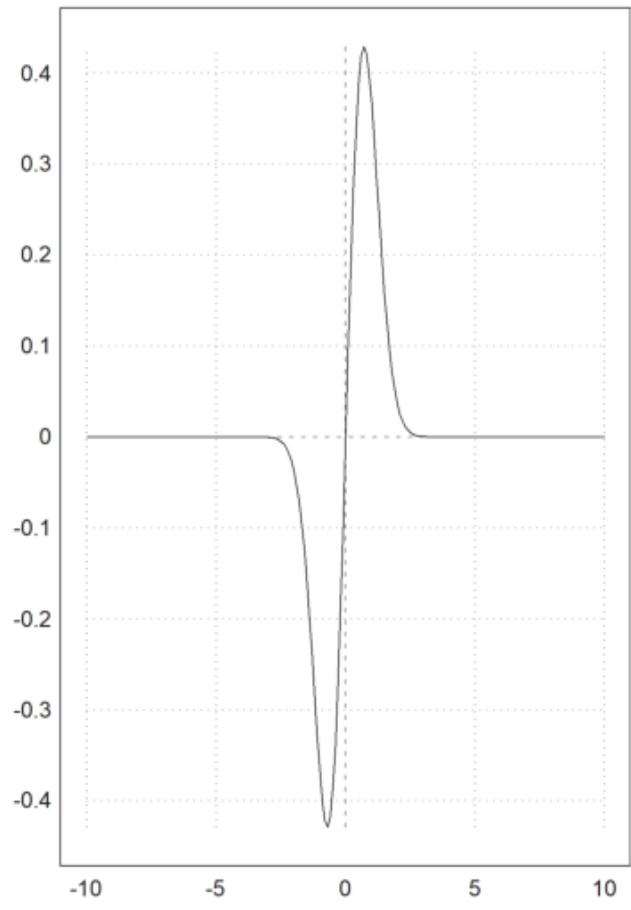
Memplot ekspresi hanyalah singkatan untuk plot data. Untuk plot data, Anda memerlukan satu atau beberapa baris nilai x , dan satu atau beberapa baris nilai y . Dari rentang dan nilai- x , fungsi `plot2d` akan menghitung data yang akan diplot, secara default dengan evaluasi fungsi yang adaptif. Untuk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

Tapi Anda bisa memasukkan data secara langsung.

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

Berikut adalah contoh dengan satu baris untuk x dan y .

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



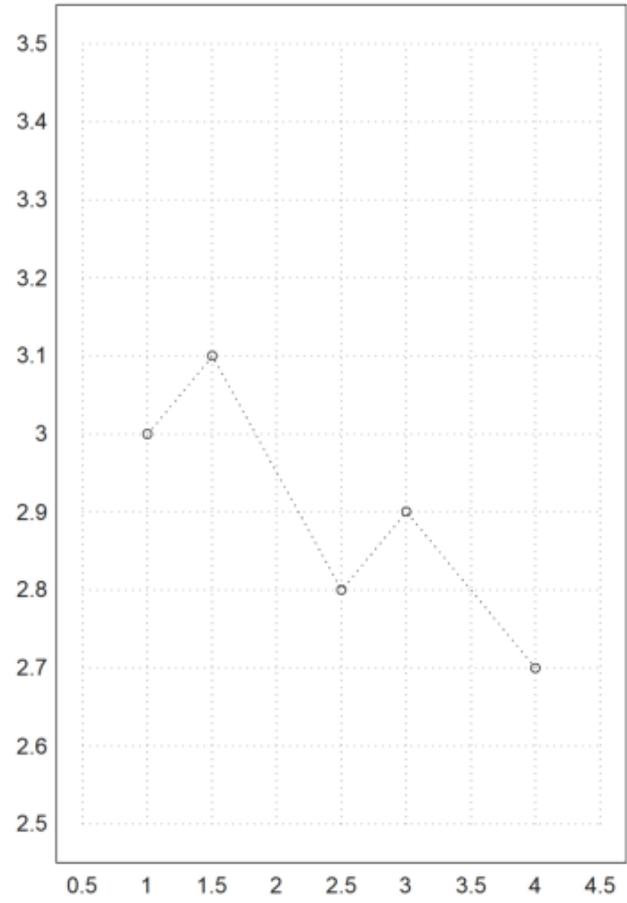
Data juga dapat diplot sebagai titik. Gunakan `poin=true` untuk ini. Plotnya bekerja seperti poligon, tetapi hanya menggambar sudut-sudutnya.

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

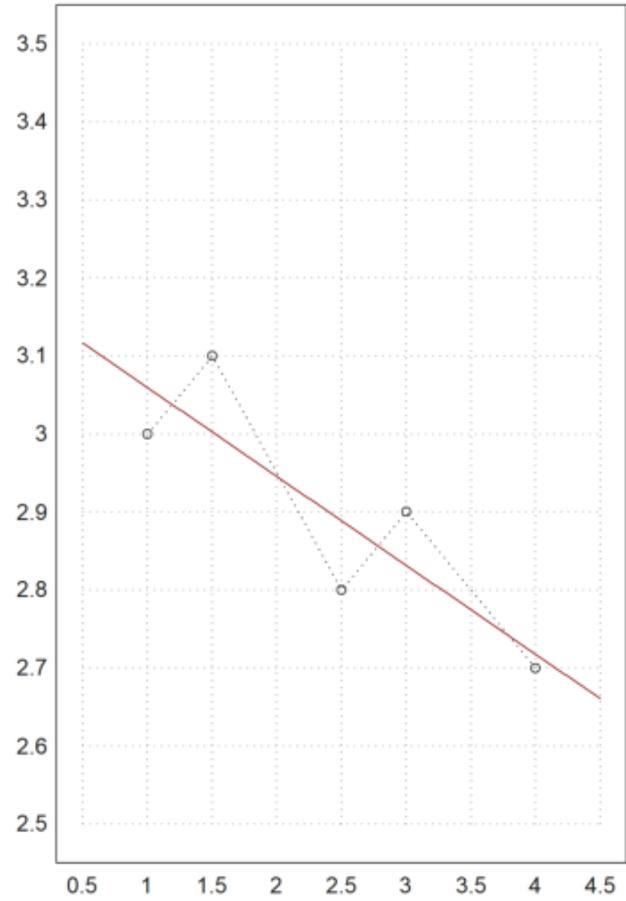
Untuk memplot set poin gunakan `>points`. Jika warna adalah vektor warna, setiap titik endapat warna yang berbeda. Untuk matriks koordinat dan vektor kolom, warna berlaku untuk baris matriks.

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

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

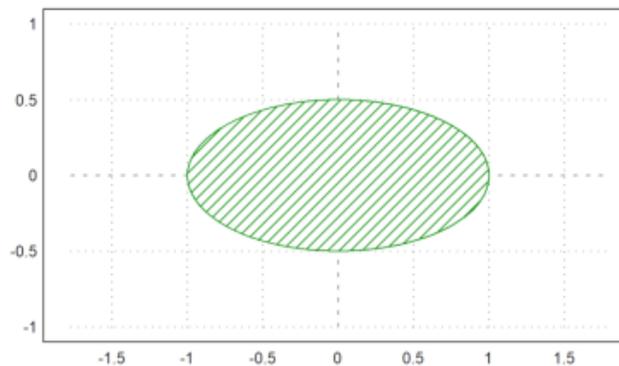


```
>p=polyfit(xdata,ydata,1); // mendapatkan garis regresi  
>plot2d("polyval(p,x)",>add,color=red): // meambahkan plot garis
```

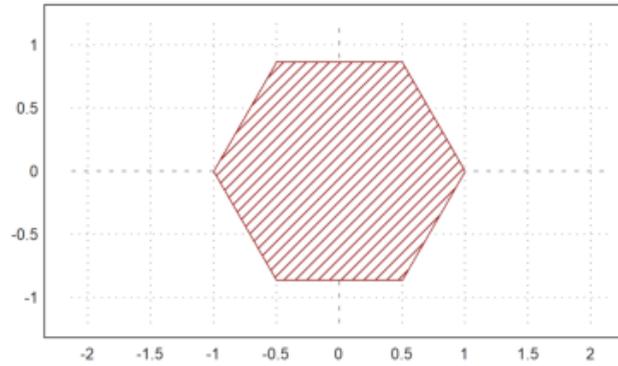


Dalam contoh berikut kami memplot elips terisi dan dua segi enam terisi menggunakan kurva tertutup dengan 6 titik dengan gaya isian berbeda.

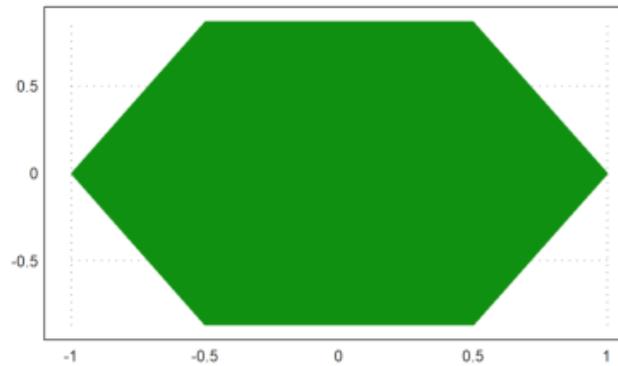
```
>x=linspace(0,2pi,1000); plot2d(sin(x),cos(x)*0.5,r=1,>filled,style="/"):
```



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

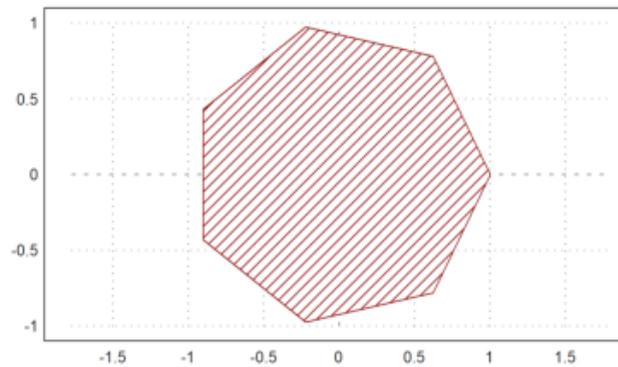


```
>t=linspace(0,2pi,6); plot2d(cos(t),sin(t),>filled,style="#"):
```



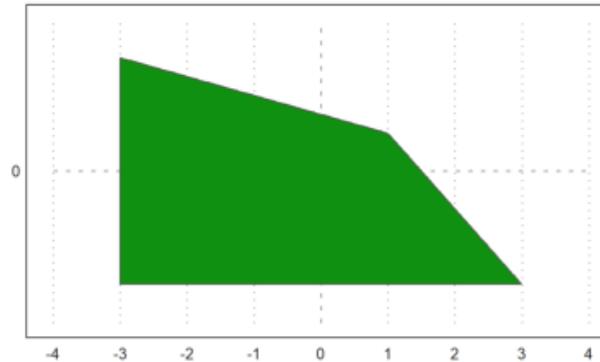
Contoh lainnya adalah segi empat, yang kita buat dengan 7 titik pada lingkaran satuan.

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



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

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

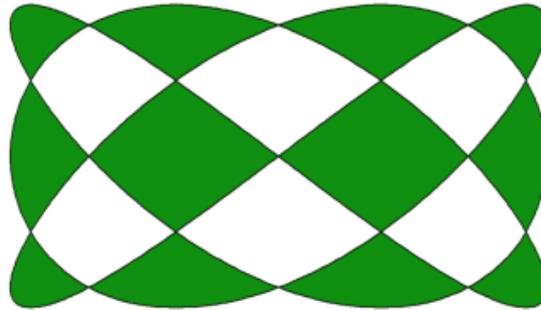


Poin utama dari bahasa matriks adalah memungkinkan untuk menghasilkan tabel fungsi dengan mudah.

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

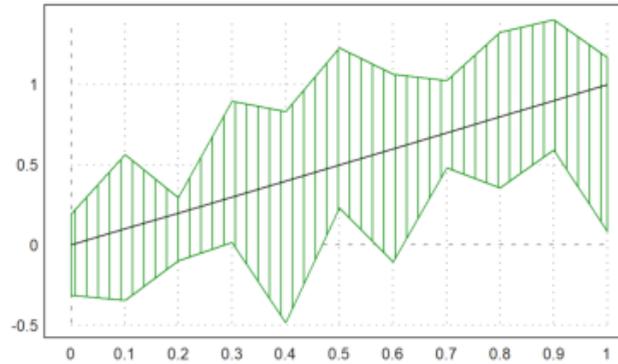
Kami sekarang memiliki vektor x dan y nilai. `plot2d()` dapat memplot nilai-nilai ini sebagai kurva yang menghubungkan titik-titik. Plotnya bisa diisi. Pada kasus ini menghasilkan hasil yang bagus karena aturan lilitan, yang digunakan untuk isi.

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



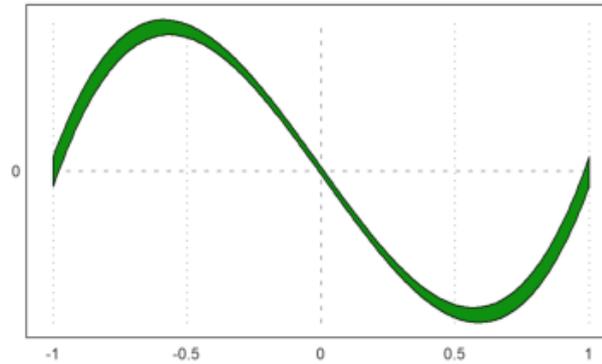
Sebuah vektor interval diplot terhadap nilai x sebagai daerah terisi antara nilai interval bawah dan atas. Hal ini dapat berguna untuk memplot kesalahan perhitungan. Tapi itu bisa juga digunakan untuk memplot kesalahan statistik.

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



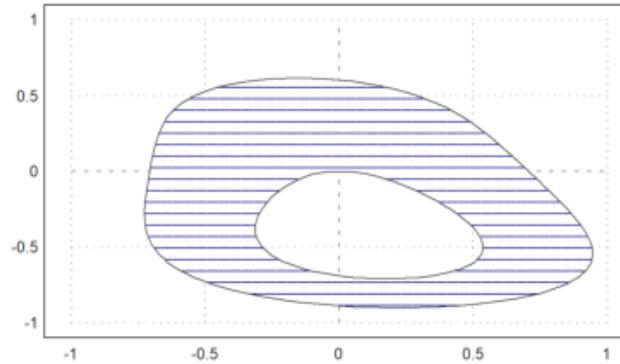
Jika x adalah vektor yang diurutkan, dan y adalah vektor interval, maka `plot2d` akan memplot rentang interval yang terisi dalam bidang. Gaya isian sama dengan gaya poligon.

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



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

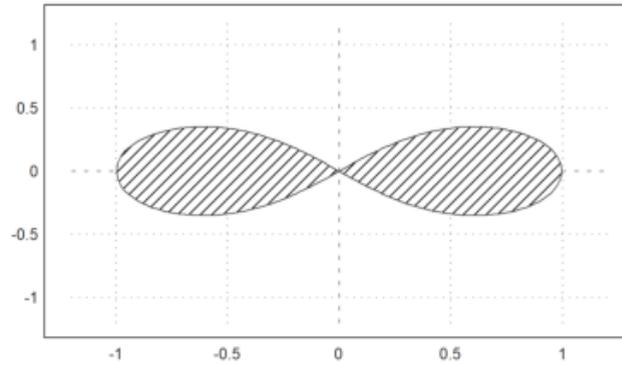
```
>expr := "2*x^2+x*y+3*y^4+y"; // mendefinisikan ekspresi f(x,y)
>plot2d(expr,level=[0;1],style="-",color=blue): // 0 <= f(x,y) <= 1
```



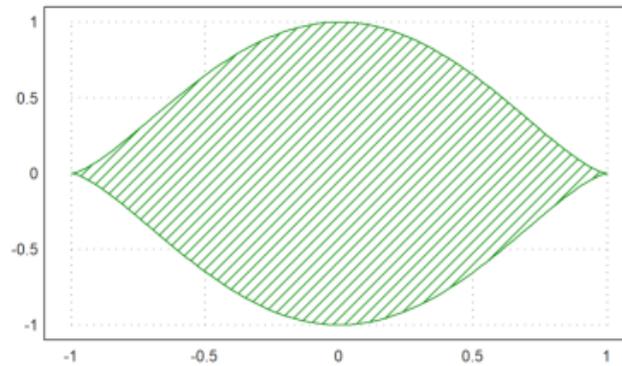
Kami juga dapat mengisi rentang nilai seperti

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

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

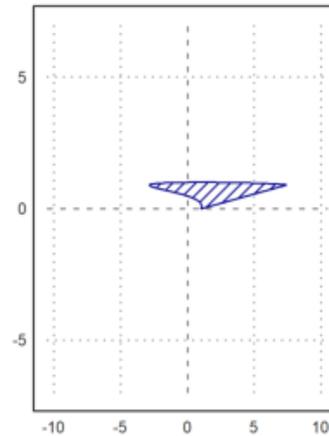
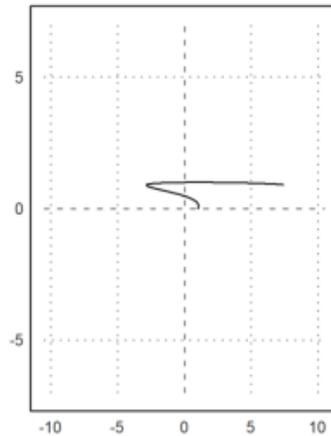


```
>plot2d("cos(x)", "sin(x)^3", xmin=0, xmax=2pi, >filled, style="/"):
```



CONTOH LAIN:

```
>t=linspace(0, 2pi, 1000); // parameter untuk kurva
>x=cos(t)*exp(t/pi); y=sin(t/pi); // x(t) dan y(t)
>figure(1,2); aspect(3/2)
>figure(1); plot2d(x,y,r=7); // plot kurva
>figure(2); plot2d(x,y,r=7,>filled,style="/",fillcolor=blue); // mengisi kurva
>figure(0):
```



Penjelasan

- `t=linspace(0,2pi,1000);`

Pada langkah pertama yaitu mendefinisikan parameter t sebagai serangkaian 1000 titik antara 0 dan 2pi. Parameter t ini akan digunakan sebagai parameter untuk menggambar kurva.

- `cos(t)*exp(t/pi); y=sin(t)*exp(t/pi);`

Didefinisikan dua vektor x dan y yang merupakan koordinat x dan y dari kurva yang akan digambar.

Fungsi :

$$\cos(t) * \exp(t/\pi)$$

digunakan untuk menghitung komponen x(x(t)), dan

$$\sin(t) * \exp(t/\pi)$$

digunakan untuk menghitung komponen y (y(t)) dari kurva.

- `figure(1,2); aspect(3/2)`

Perintah ini digunakan untuk mengatur tampilan gambar. Perintah `figure(1,2)` digunakan membuat dua gambar (1 dan 2) dalam satu jendela gambar. Dan perintah `aspect(3/2)` mengatur rasio aspek gambar menjadi 3:2, yang mempengaruhi bentuk dan ukuran gambar yang akan digambar.

- `figure(1); plot2d(x,y,r=7);`

Perintah ini memilih gambar pertama (1) dan menggunakan perintah `plot2d` untuk menggambar kurva yang dihitung sebelumnya. Parameter `r=10` mengatur lebar garis plot. Ini menghasilkan kurva tanpa adanya isi atau arsiran di dalamnya.

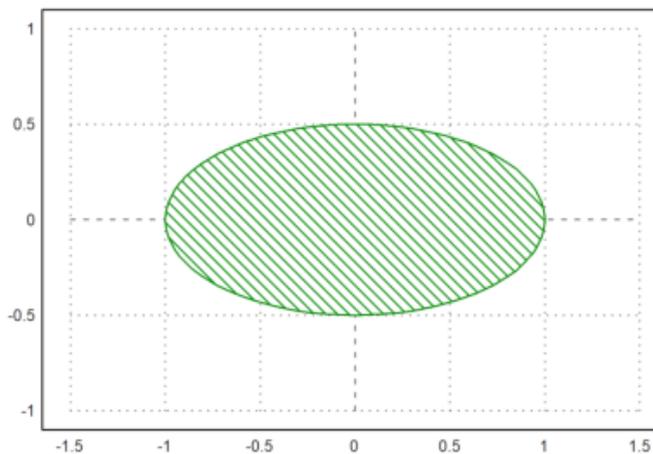
- `figure(2); plot2d(x,y,r=7,>filled,style='/',fillcolor=blue);`

Selanjutnya pada perintah ini beralih ke gambar kedua (2) dan menggunakan perintah `plot2d` lagi untuk menggambar kurva yang sama dengan pengisian area di bawahnya. Perintah `>filled` digunakan untuk mengisi area di bawah kurva, `style='/'` digunakan untuk mengatur gaya garis menjadi garis miring, dan `fillcolor=blue` digunakan untuk mengatur warna isian menjadi biru.

- figure(0);

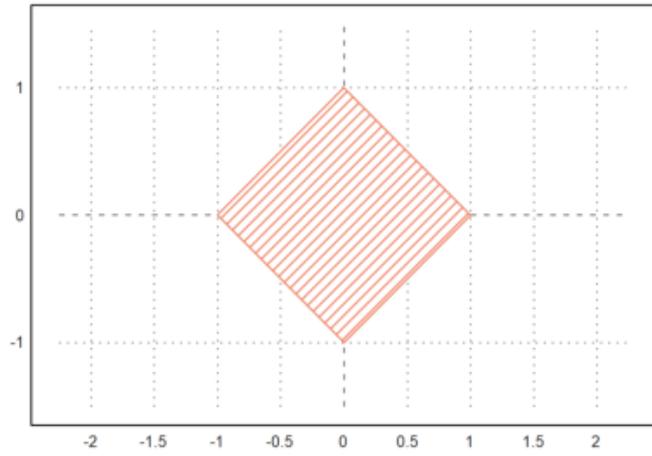
Baris perintah ini digunakan untuk mengakhiri gambar dan kembali ke tampilan biasa tanpa gambar. Perintah ini berfungsi untuk menyelesaikan proses penggambaran.

```
>x=linspace(0,2pi,100); plot2d(cos(x),sin(x)*0.5,r=1,>filled,style="\"):
```



```
>t=linspace(0,2pi,4);
```

```
>plot2d(cos(t),sin(t),>filled, style="/",fillcolor=orange,r=1.5):
```



Grafik Fungsi Parametrik

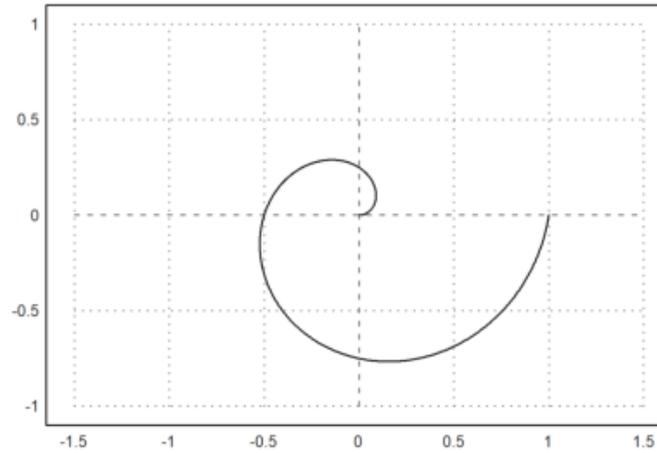
Nilai-x tidak perlu diurutkan. (x,y) hanya menggambarkan kurva. Jika x diurutkan, kurva tersebut merupakan grafik fungsi.

Dalam contoh berikut, kami memplot spiral

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

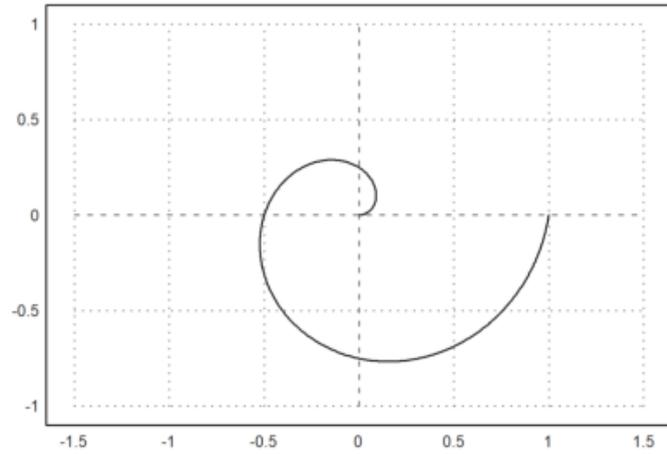
Kita perlu menggunakan banyak titik untuk tampilan yang halus atau fungsi adaptif() untuk mengevaluasi ekspresi (lihat fungsi adaptif() untuk lebih jelasnya).

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

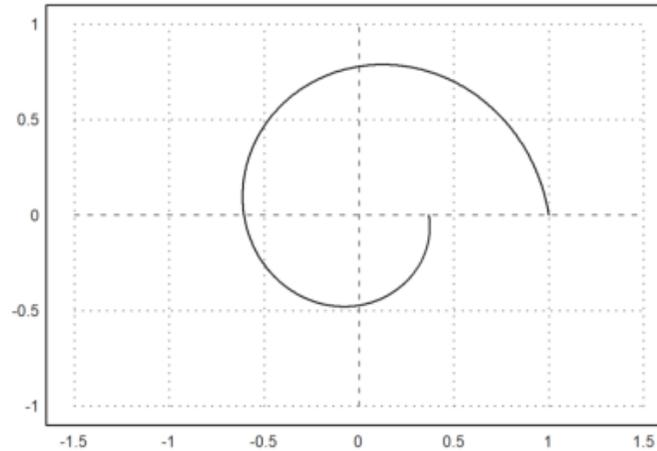


Atau, dimungkinkan untuk menggunakan dua ekspresi untuk kurva. Berikut ini plot kurva yang sama seperti di atas.

```
>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):
```



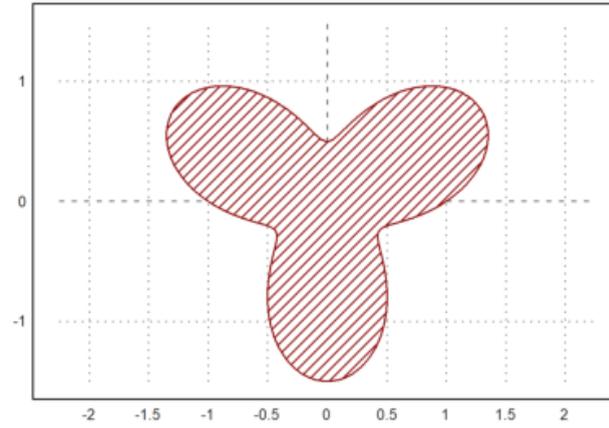
Dalam contoh berikutnya, kami memplot kurva

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

dengan

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

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



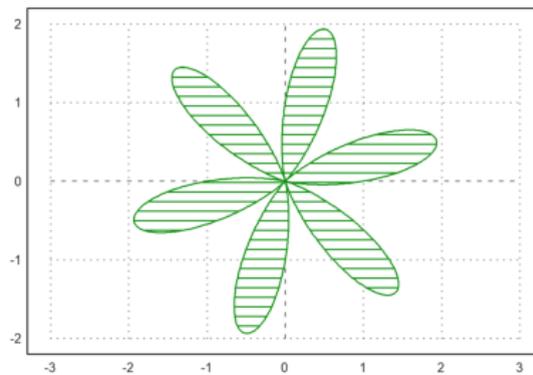
CONTOH LAIN:
Contoh 1

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

dengan

$$r(t) = 1 + \sin(6t)$$

```
>t=linspace(0,2pi,1000); r=1+sin(6*t); x=r*sin(t); y=r*cos(t); ...  
>plot2d(x,y,>filled,fillcolor=green,style="-",r=2):
```



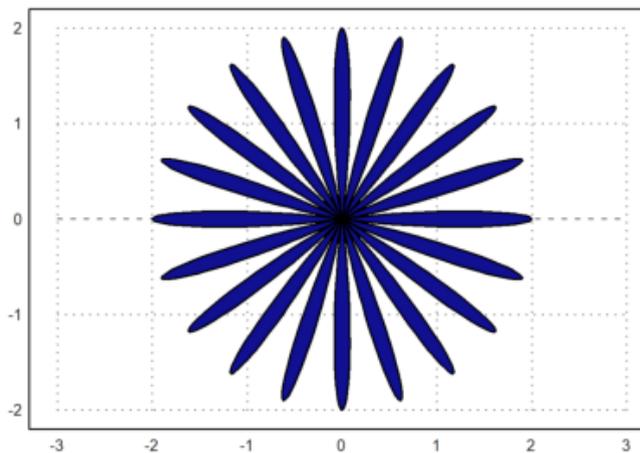
Contoh 2

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

dengan

$$r(t) = 1 + \cos(20t)$$

```
>t=linspace(0,2pi,1000); r=1+cos(20*t); x=r*cos(t); y=r*sin(t); ...  
>plot2d(x,y,>filled,fillcolor=blue,style="--",r=2):
```



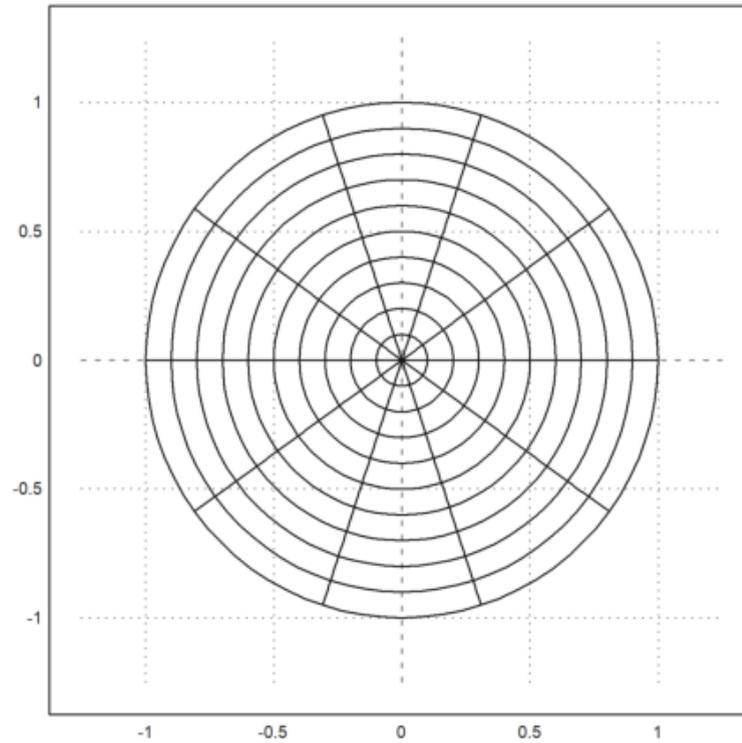
Menggambar Grafik Bilangan Kompleks

Array bilangan kompleks juga dapat diplot. Kemudian titik-titik grid akan terhubung. Jika sejumlah garis kisi ditentukan (atau vektor garis kisi 1×2) dalam argumen `cgrid`, hanya garis kisi tersebut yang terlihat.

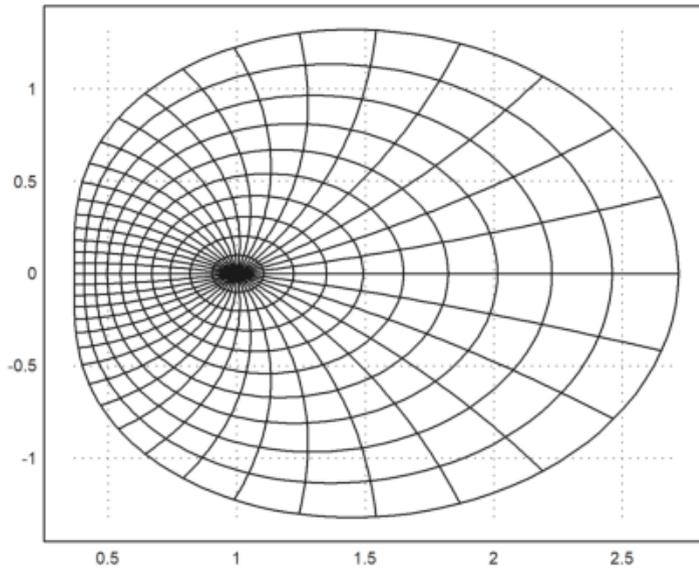
Matriks bilangan kompleks akan secara otomatis diplot sebagai kisi di bidang kompleks.

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

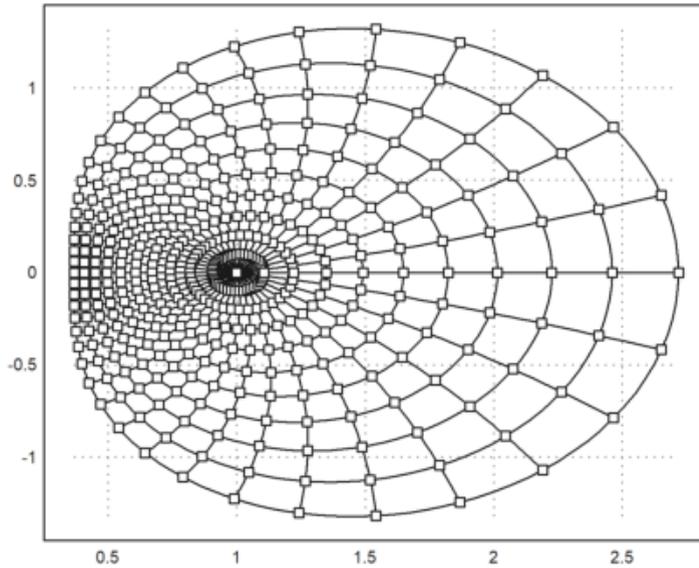
```
>aspect(); r=linspace(0,1,50); a=linspace(0,2pi,80)'; z=r*exp(I*a);...  
>plot2d(z,a=-1.25,b=1.25,c=-1.25,d=1.25,cgrid=10):
```



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



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

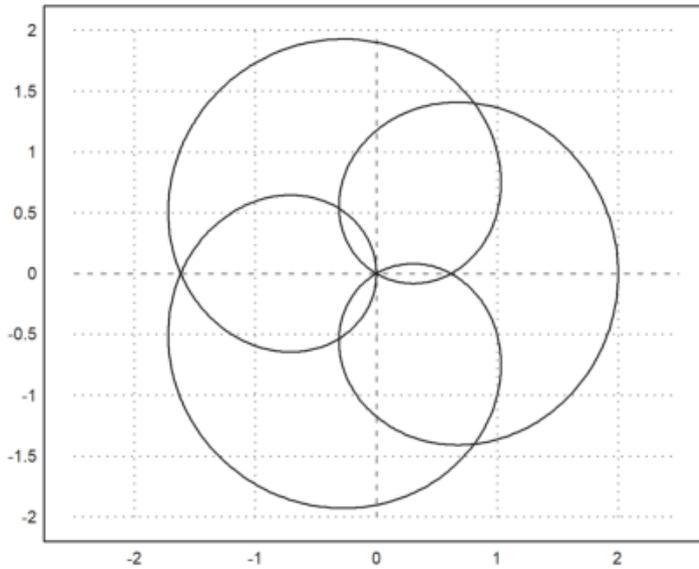


Sebuah vektor bilangan kompleks secara otomatis diplot sebagai kurva pada bidang kompleks dengan bagian real dan bagian imajiner.

Dalam contoh, kami memplot lingkaran satuan dengan

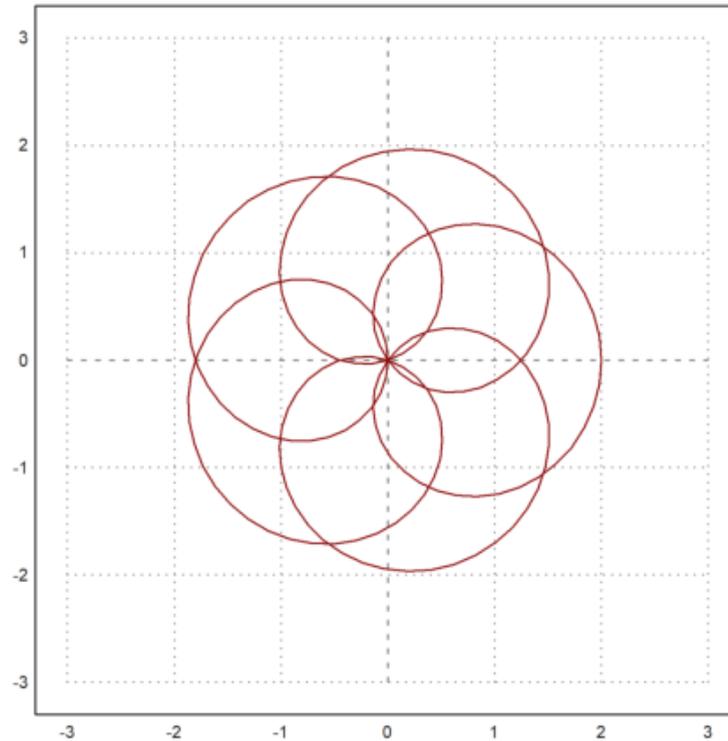
lateks: $\gamma(t) = e^{it}$

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



CONTOH LAIN:
Contoh 1

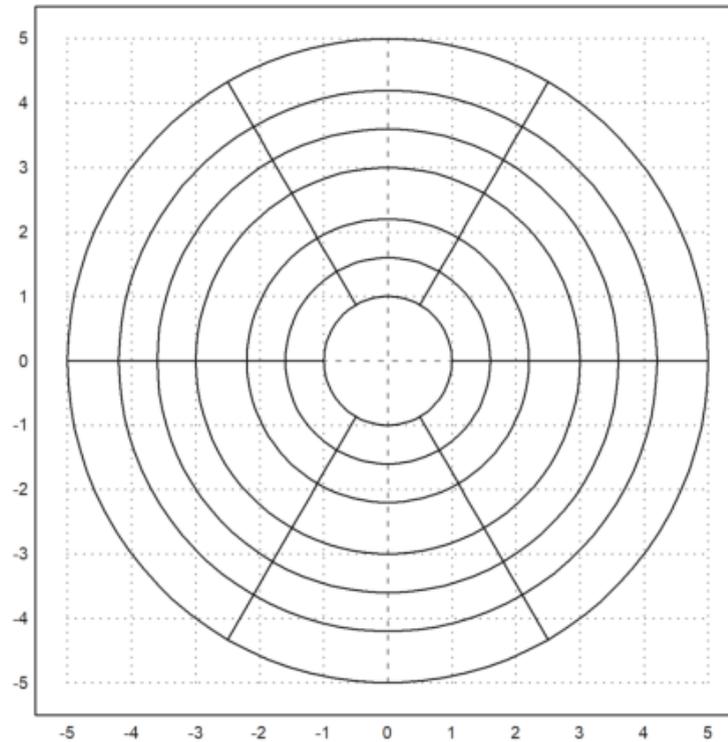
```
>aspect(1); n=linspace(0,2pi,200);  
>plot2d(exp(I*n)+exp(6*I*n),-3,3,-3,3, color=red):
```



Contoh di atas merupakan penjumlahan dua fungsi eksponensial dengan n yang merupakan suatu array kompleks dari 0 sampai 2π hingga 200 titik sehingga dapat membentuk beberapa lingkaran. Gambar di atas menggunakan aspect 1 dengan rentang sumbu x dan y dari -3 sampai 3.

Contoh 2

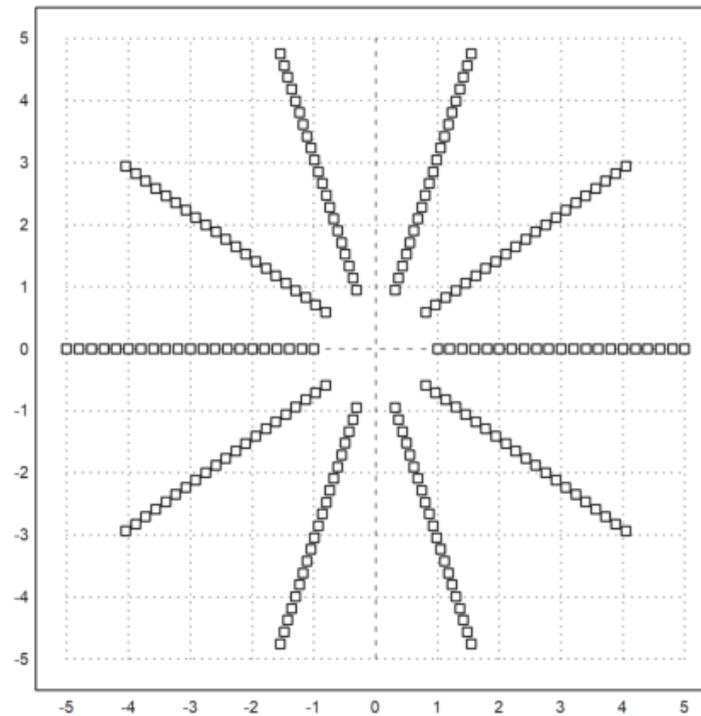
```
>aspect(1); r=linspace(1,5,20); a=linspace(0,2pi,90)'; z=r*(cos(a) + I*sin(a));  
>plot2d(z,-5,5,-5,5, cgrid=6):
```



Perintah di atas merupakan eksponensial kompleks yang melibatkan fungsi trigonometri dengan r yaitu array dari 1 sampai 5 yang berisi 20 elemen di dalamnya

Contoh 3

```
>aspect(1); r=linspace(1,5,20); a=linspace(0,2pi,10)'; z=r*(cos(a) + I*sin(a));  
>plot2d(z,-5,5,-5,5, cgrid=6, >points):
```



Contoh gambar di atas adalah kurva fungsi kompleks yang sama dengan sebelumnya, namun ada penambahan sintaks `<points` ke dalam `plot2d`, sehingga gambar yang dihasilkan berupa titik-titik.

Plot Statistik

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

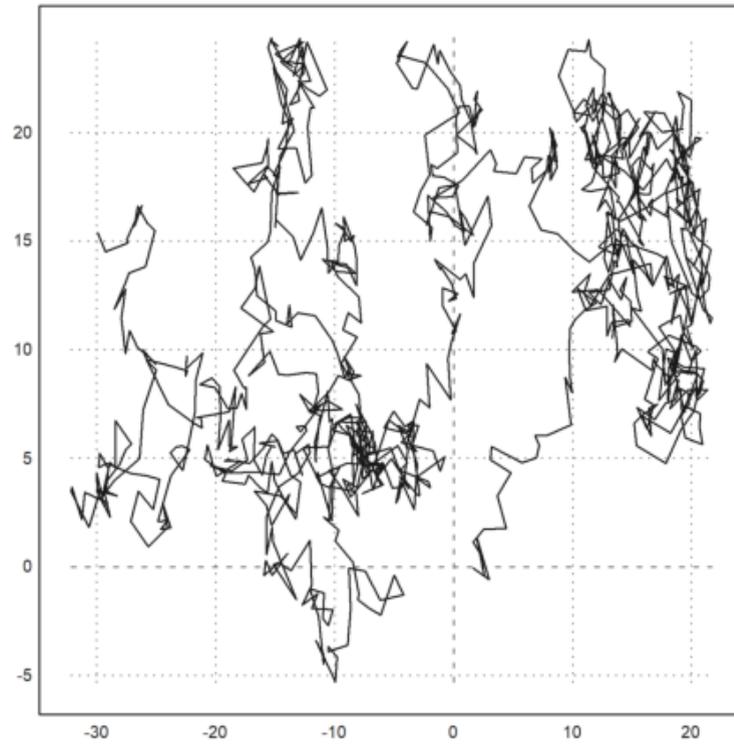
Jumlah kumulatif dari nilai terdistribusi 0-1-normal menghasilkan jalan acak.

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

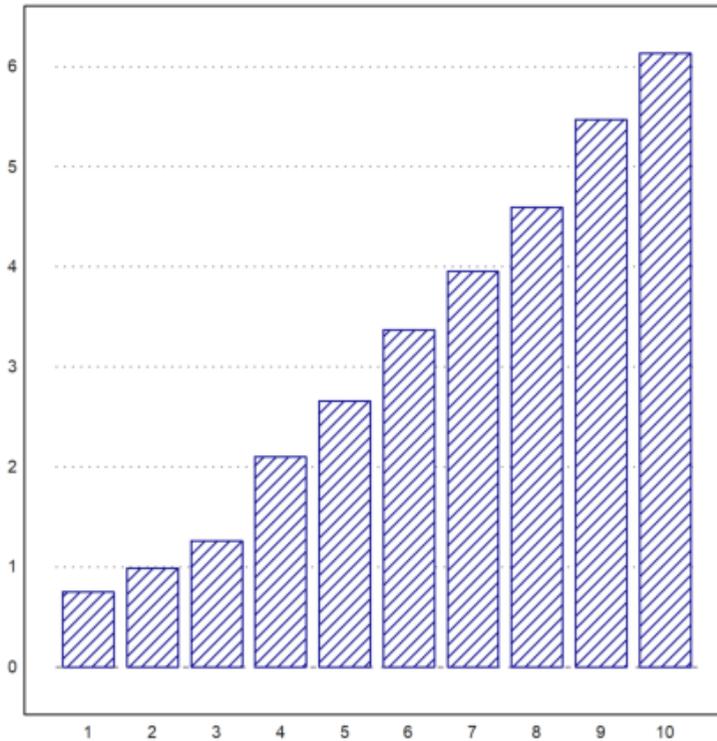


Menggunakan dua baris menunjukkan jalan dalam dua dimensi.

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

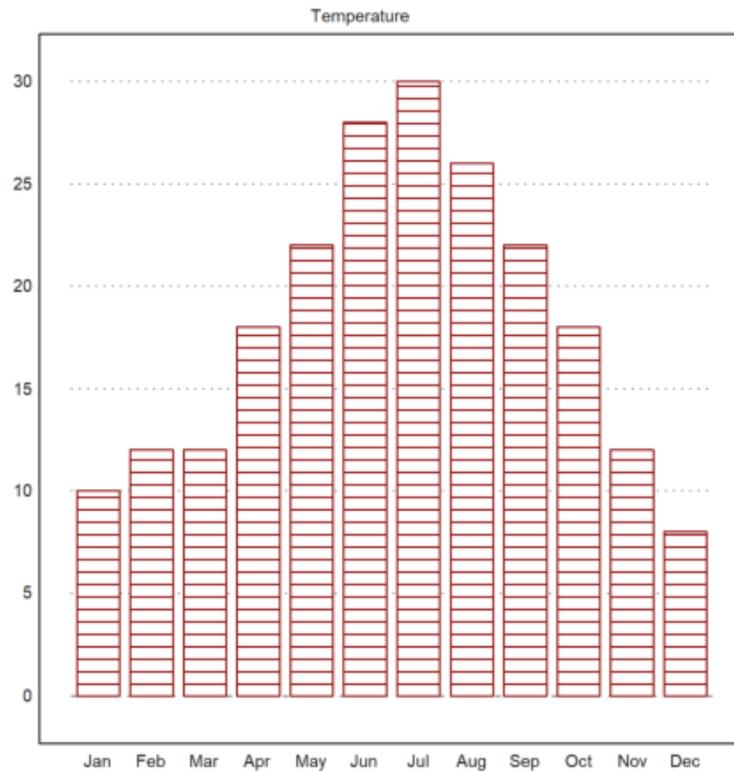


```
>columnplot(cumsum(random(10)),style="/",color=blue):
```

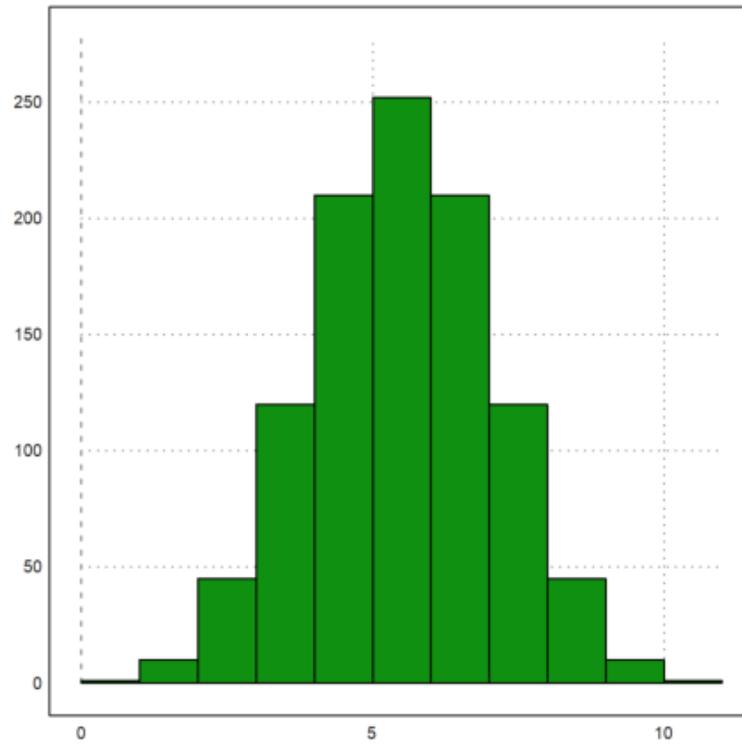


Itu juga dapat menampilkan string sebagai label.

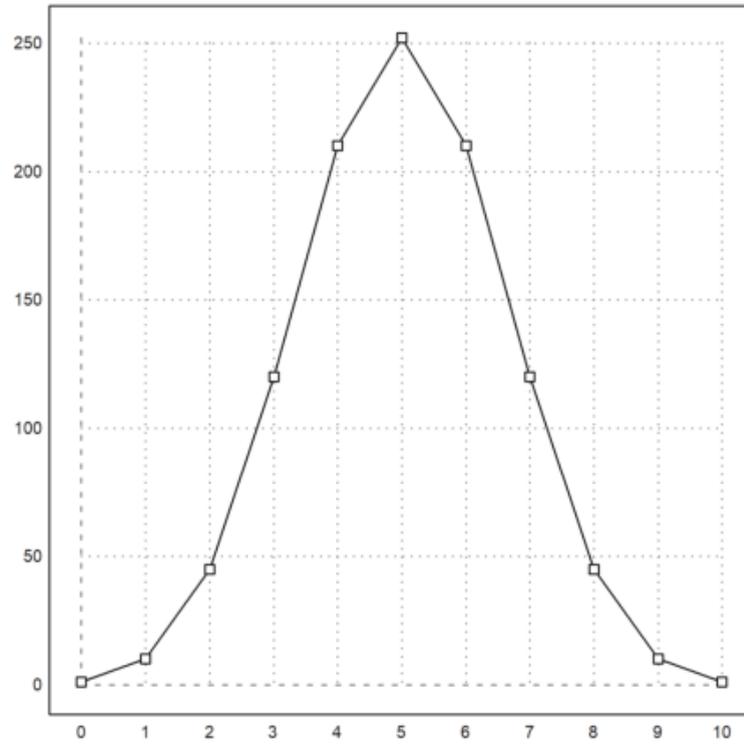
```
>months=["Jan","Feb","Mar","Apr","May","Jun", ...  
> "Jul","Aug","Sep","Oct","Nov","Dec"];  
>values=[10,12,12,18,22,28,30,26,22,18,12,8];  
>columnsplot(values,lab=months,color=red,style="-");  
>title("Temperature");
```



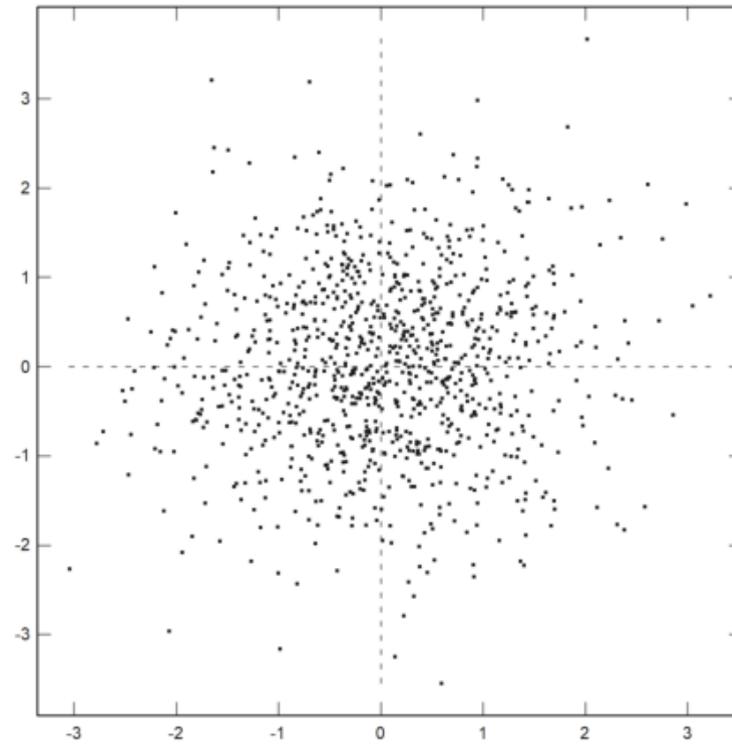
```
>k=0:10;  
>plot2d(k,bin(10,k),>bar):
```



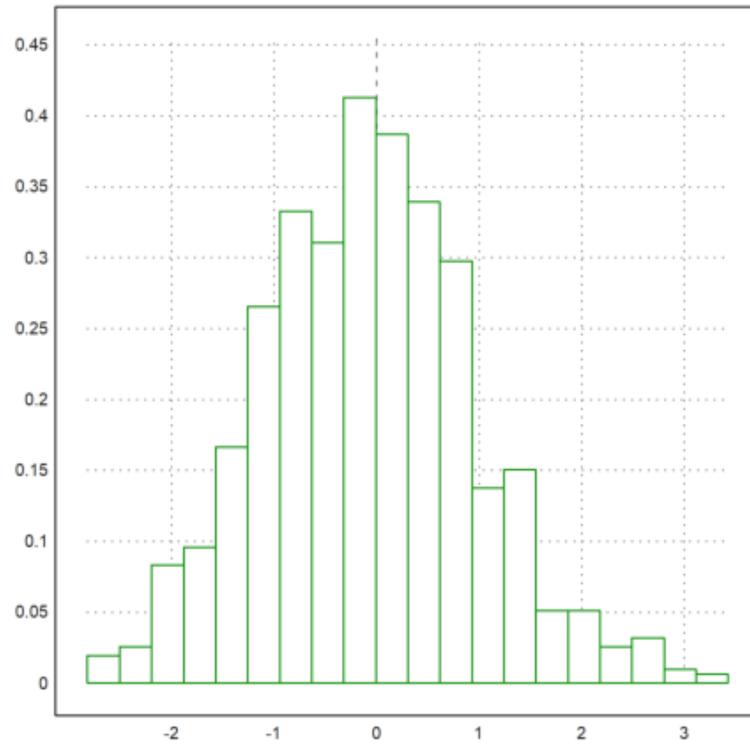
```
>plot2d(k,bin(10,k)); plot2d(k,bin(10,k),>points,>add):
```



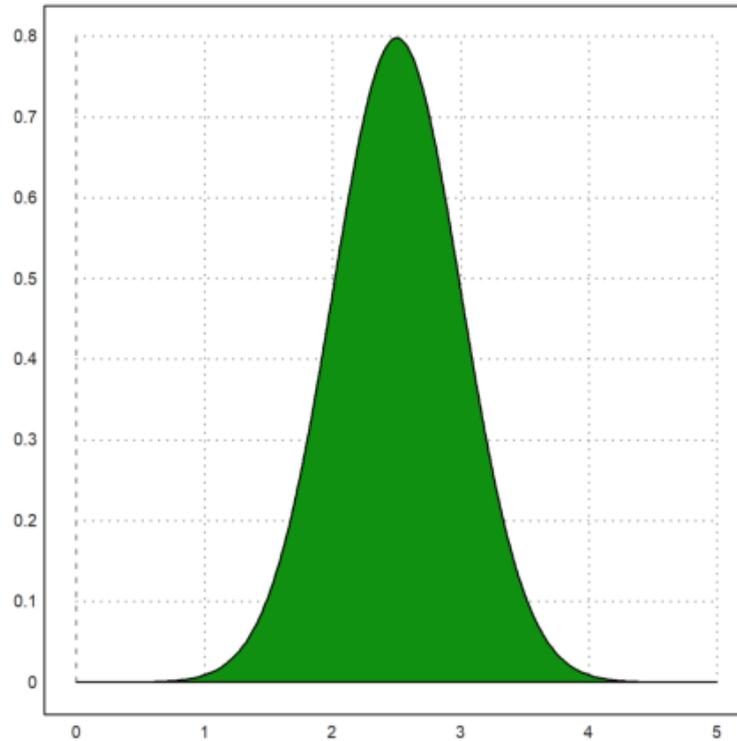
```
>plot2d(normal(1000),normal(1000),>points,grid=6,style=".."):
```



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

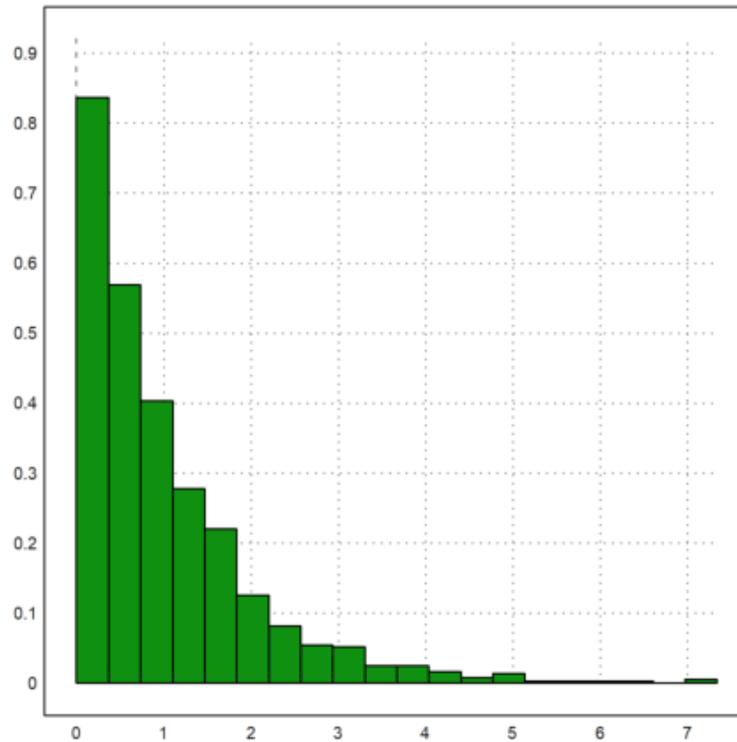


```
>plot2d("qnormal",0,5;2.5,0.5,>filled):
```



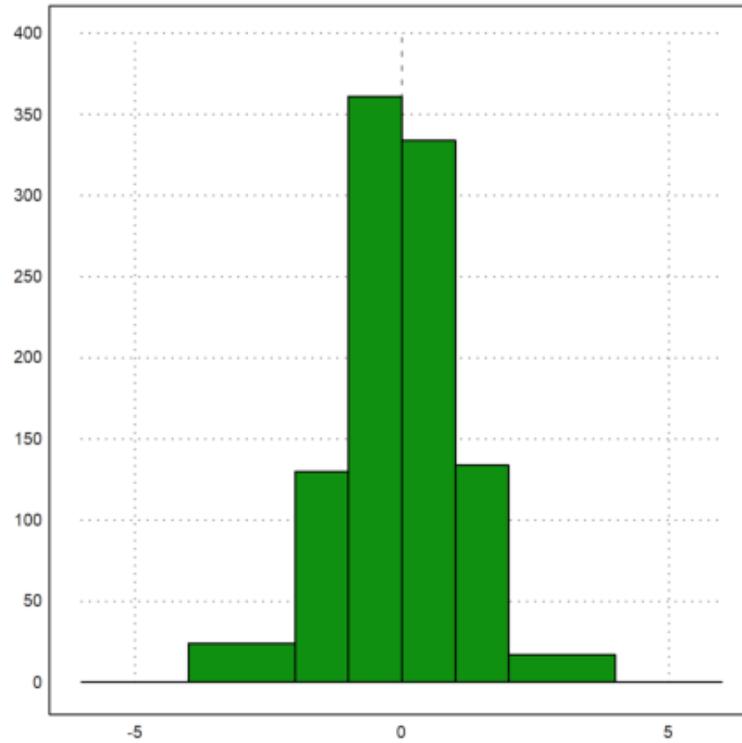
Untuk memplot distribusi statistik eksperimental, Anda dapat menggunakan `distribution=n` dengan `plot2d`.

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



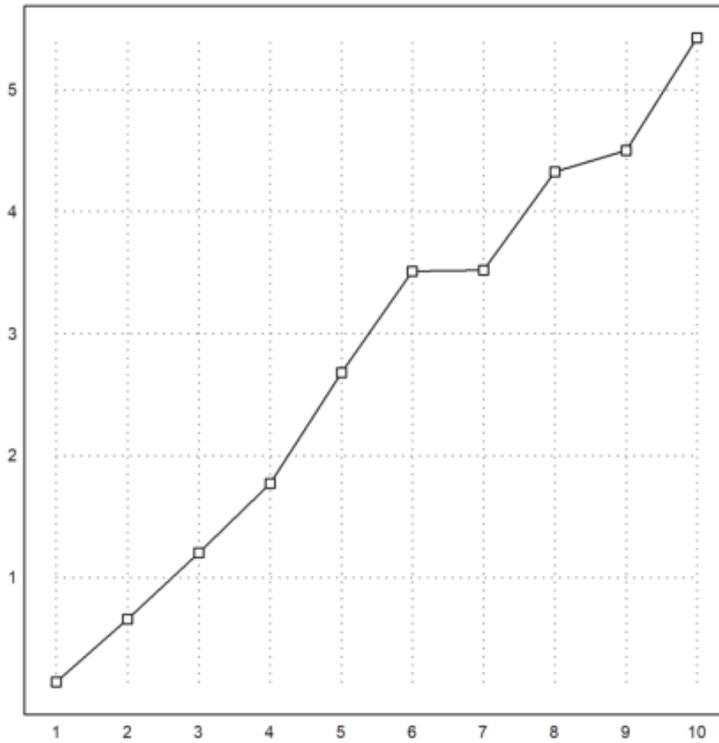
Atau Anda dapat menghitung distribusi dari data dan memplot hasilnya dengan `>bar` di `plot3d`, atau dengan `plot.kolom`.

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

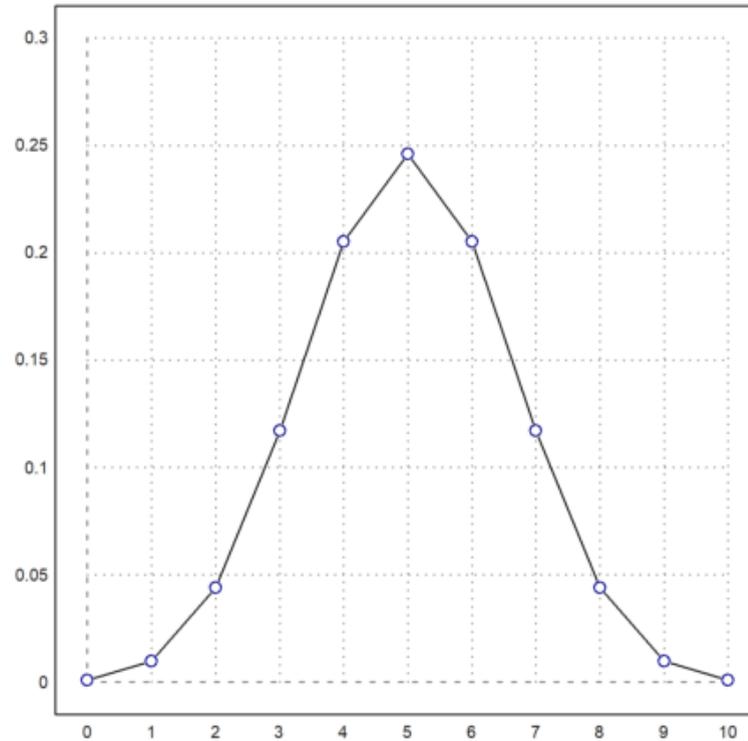


Fungsi `statplot()` menyetel gaya dengan string sederhana.

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



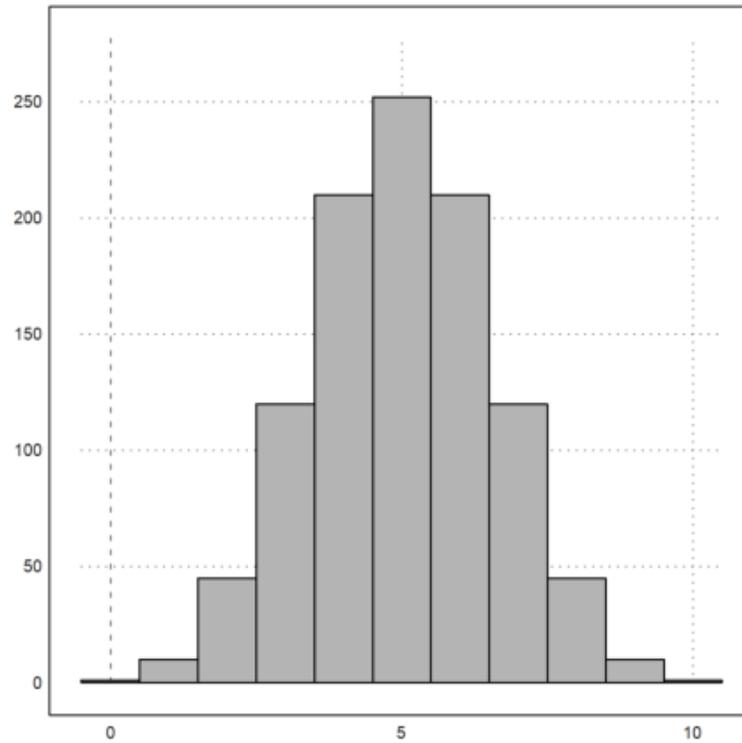
```
>n=10; i=0:n; ...  
>plot2d(i,bin(n,i)/2^n,a=0,b=10,c=0,d=0.3); ...  
>plot2d(i,bin(n,i)/2^n,points=true,style="ow",add=true,color=blue):
```



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

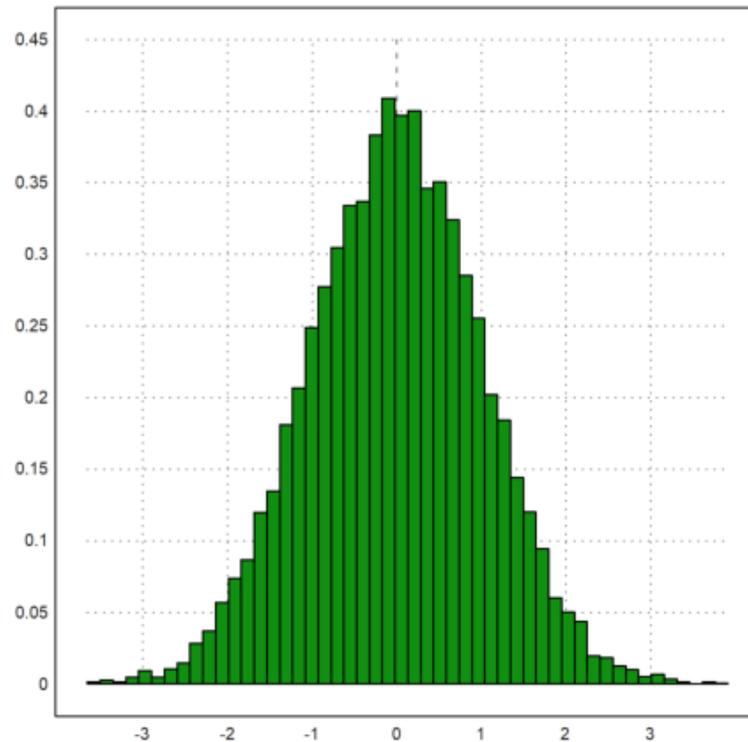
Gaya isian dapat digunakan seperti di atas.

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

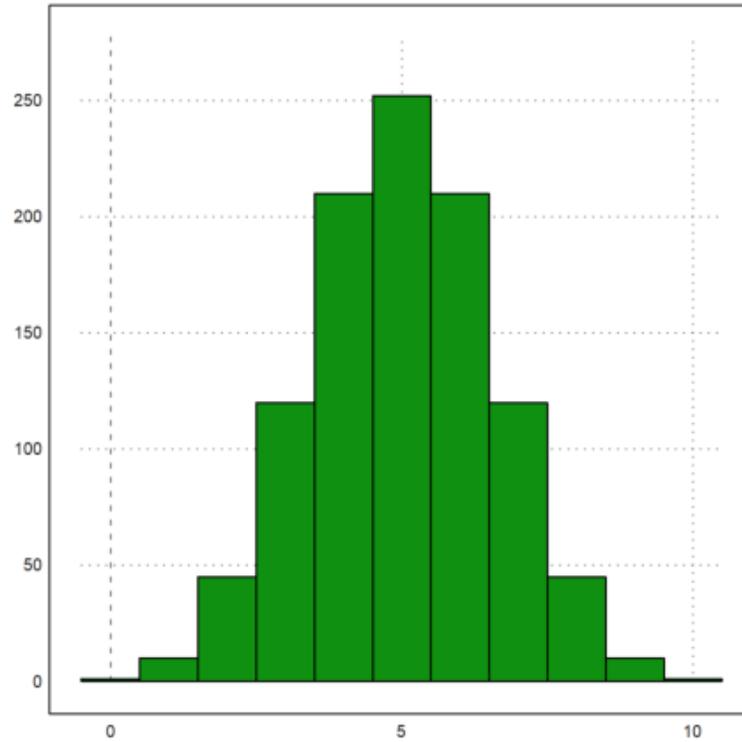


Data untuk plot batang (`bar=1`) dan histogram (`histogram=1`) dapat dinyatakan secara eksplisit dalam `xv` dan `yv`, atau dapat dihitung dari distribusi empiris dalam `xv` dengan `>distribusi` (atau `distribusi=n`). Histogram nilai `xv` akan dihitung secara otomatis dengan `>histogram`. Jika `>genap`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

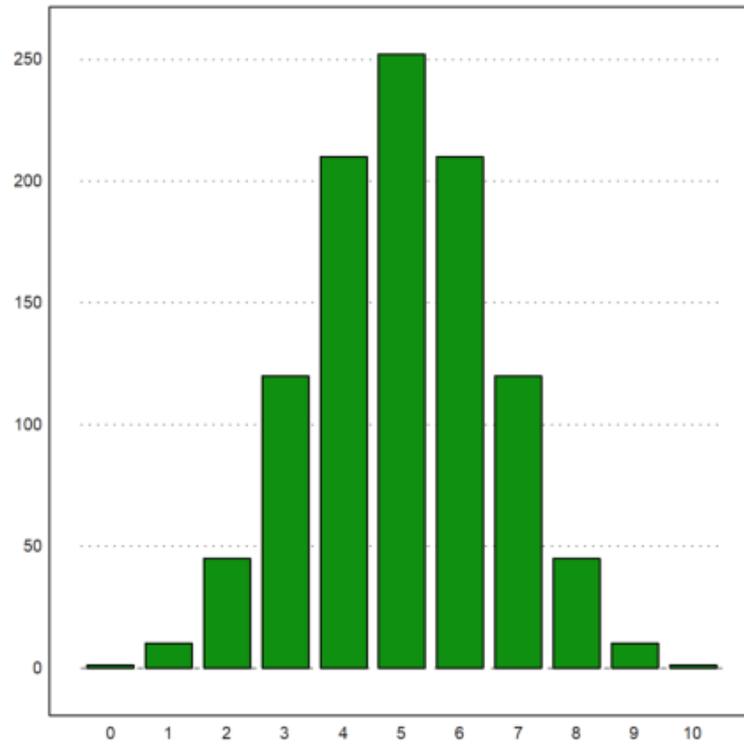
```
>plot2d(normal(10000),distribution=50):
```



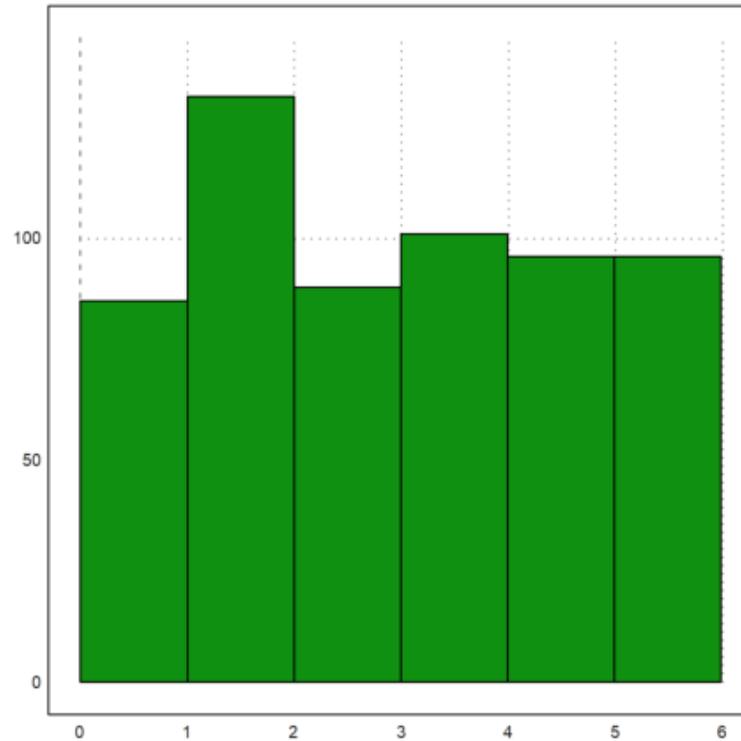
```
>k=0:10; m=bin(10,k); x=(0:11)-0.5; plot2d(x,m,>bar):
```



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

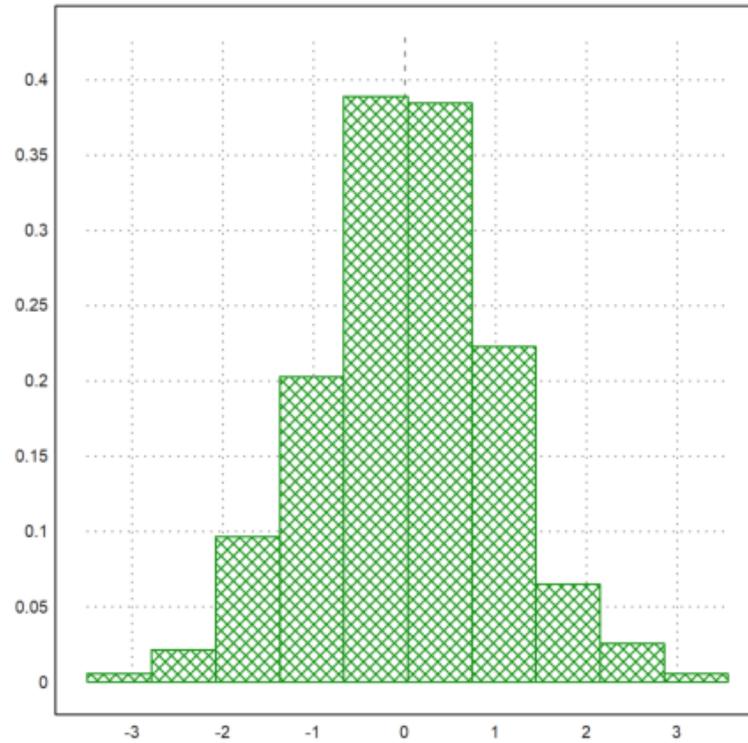


```
>plot2d(random(600)*6,histogram=6):
```



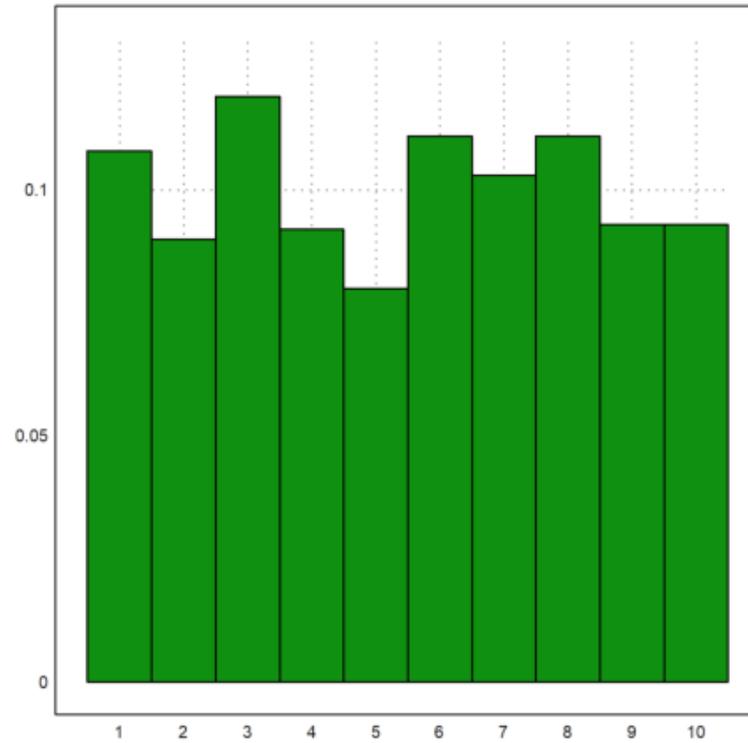
Untuk distribusi, ada parameter `distribusi=n`, yang menghitung nilai secara otomatis dan mencetak distribusi relatif dengan `n` sub-interval.

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



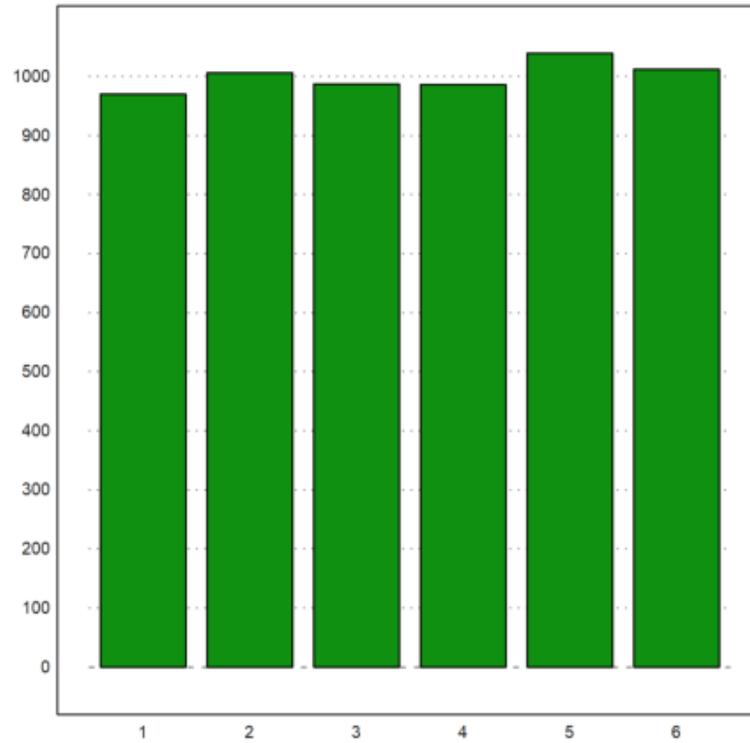
Dengan parameter `even=true`, ini akan menggunakan interval integer.

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

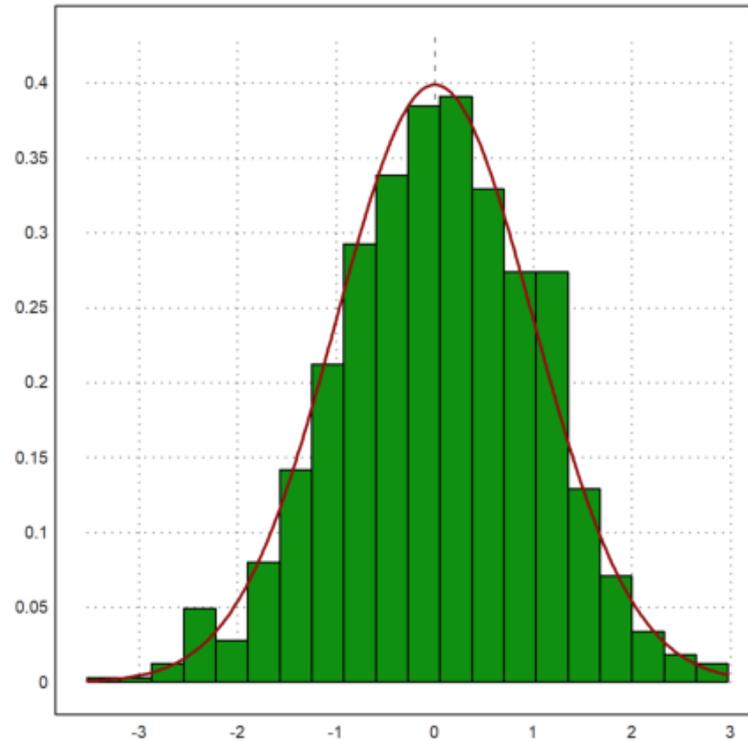


Perhatikan bahwa ada banyak plot statistik, yang mungkin berguna. Silahkan lihat tutorial tentang statistik.

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

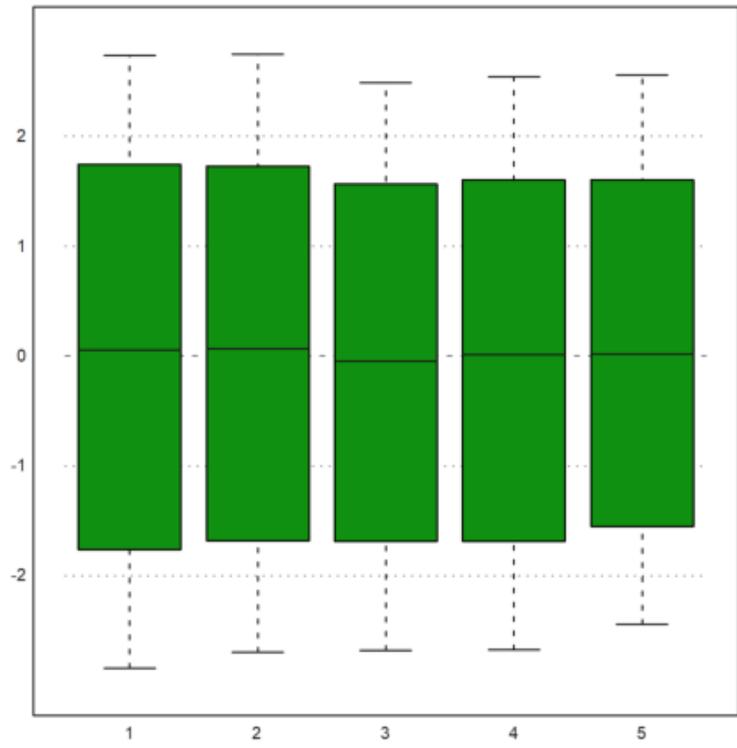


```
>plot2d(normal(1,1000),>distribution); ...  
> plot2d("qnormal(x)",color=red,thickness=2,>add):
```



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

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



Fungsi Implisit

Plot implisit menunjukkan garis level yang menyelesaikan $f(x,y)=\text{level}$, di mana "level" dapat berupa nilai tunggal atau vektor nilai. Jika $\text{level}=\text{"auto"}$, akan ada garis level n_c , yang akan menyebar antara fungsi minimum dan maksimum secara merata. 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 dapat berupa matriks nilai.

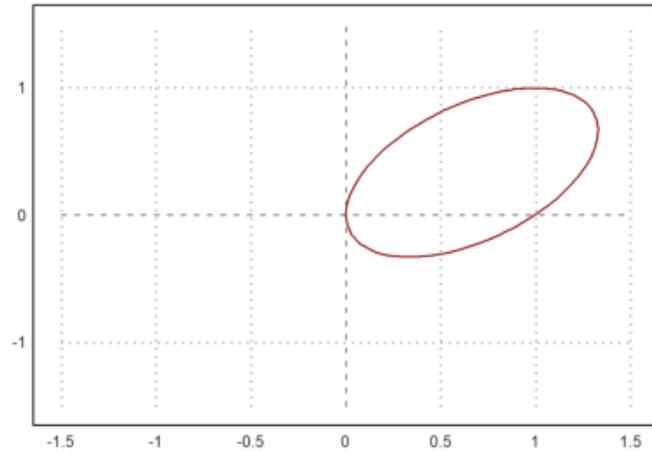
Euler dapat menandai garis level

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

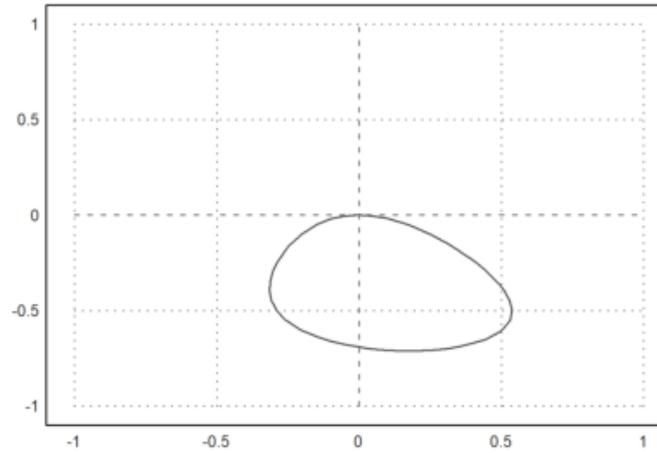
dari fungsi apapun.

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

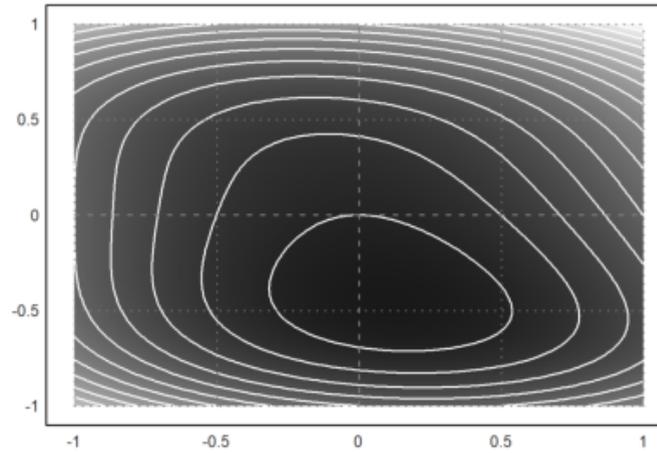
```
>aspect(1.5);  
>plot2d("x^2+y^2-x*y-x",r=1.5,level=0,contourcolor=red):
```



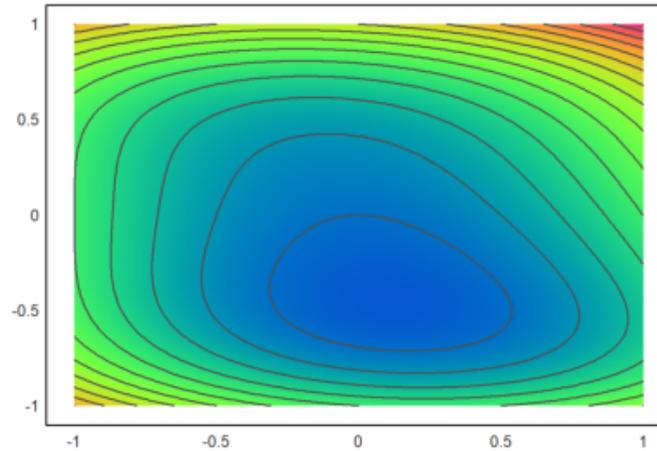
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=0): // Solutions of f(x,y)=0
```



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

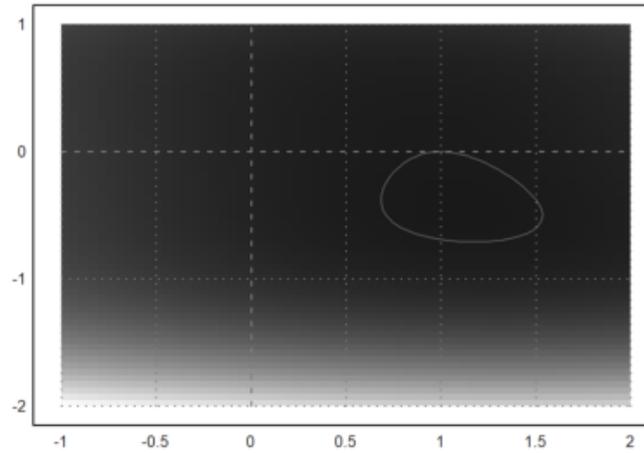


```
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=200,grid=4): // nicer
```

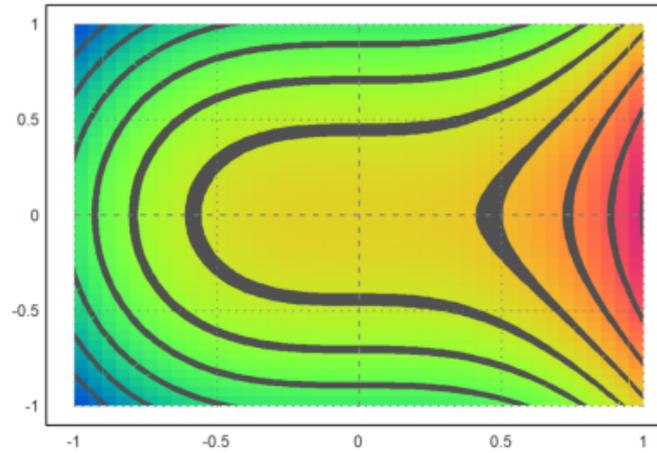


Ini berfungsi untuk plot data juga. Tetapi Anda harus menentukan rentangnya untuk label sumbu.

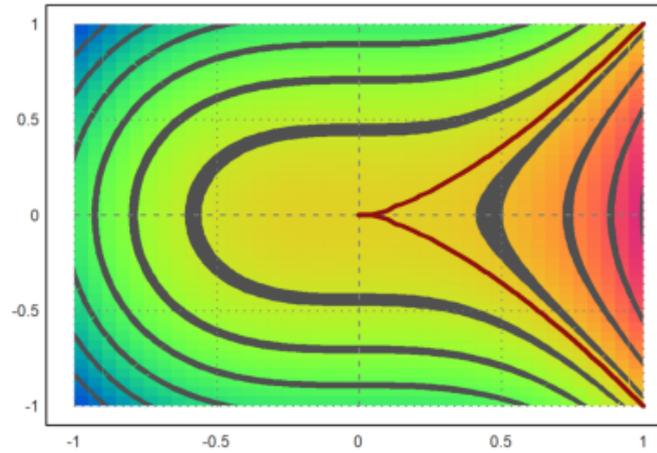
```
>x=-2:0.05:1; y=x'; z=exp(x,y);  
>plot2d(z,level=0,a=-1,b=2,c=-2,d=1,>hue):
```



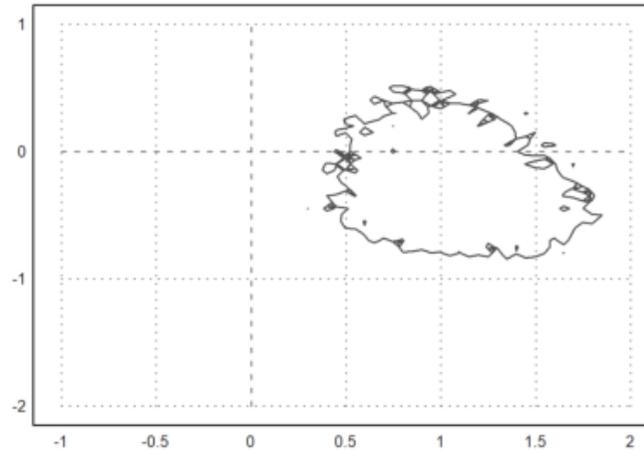
```
>plot2d("x^3-y^2",>contour,>hue,>spectral):
```



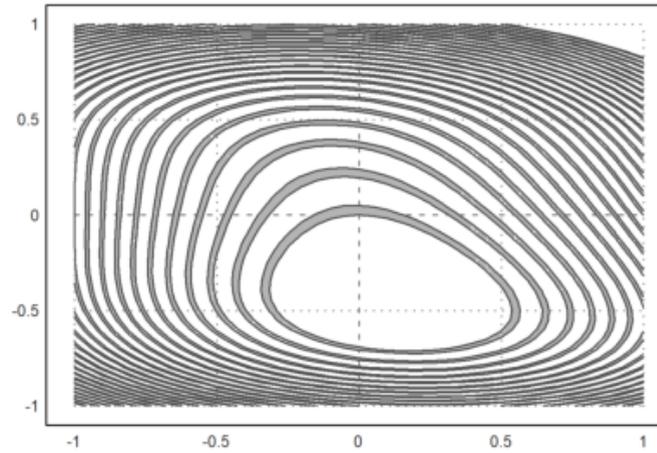
```
>plot2d("x^3-y^2",level=0,contourwidth=3,>add,contourcolor=red):
```



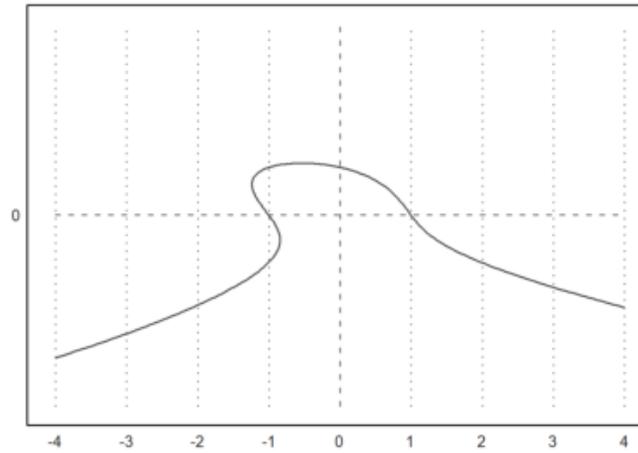
```
>z=z+normal(size(z))*0.2;  
>plot2d(z,level=0.5,a=-1,b=2,c=-2,d=1):
```



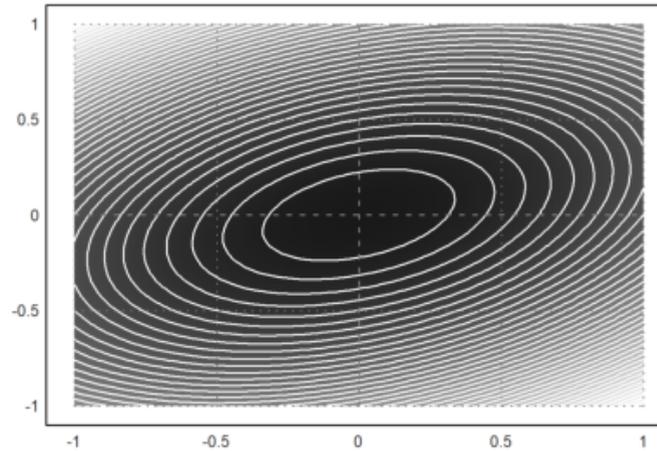
```
>plot2d(expr,level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
```



```
>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):
```



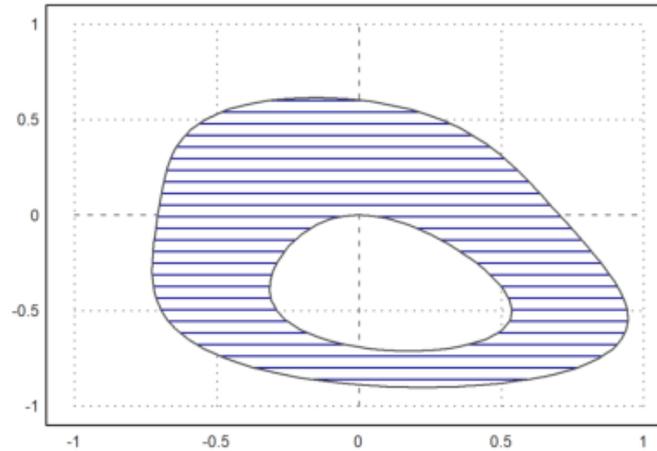
Juga dimungkinkan untuk mengisi set

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

dengan rentang tingkat.

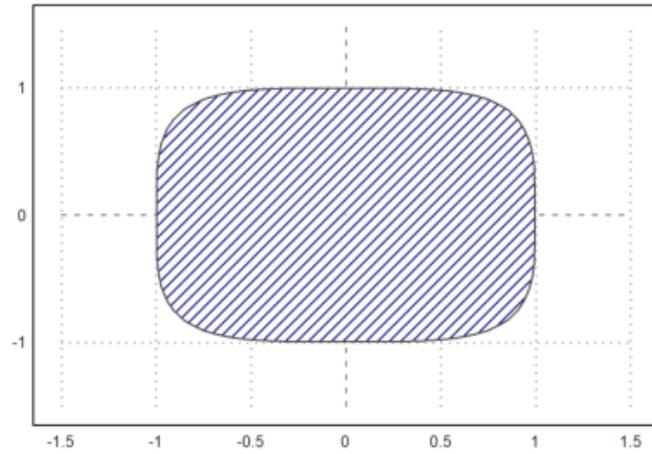
Dimungkinkan untuk mengisi wilayah nilai untuk fungsi tertentu. Untuk ini, level harus berupa matriks $2 \times n$. Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

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

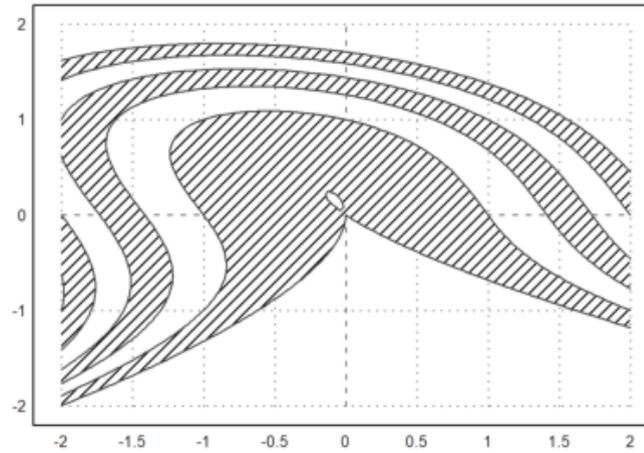


Plot implisit juga dapat menunjukkan rentang level. Kemudian level harus berupa matriks $2 \times n$ dari interval level, di mana baris pertama berisi awal dan baris kedua adalah akhir dari setiap interval. Atau, vektor baris sederhana dapat digunakan untuk level, dan parameter `dl` memperluas nilai level ke interval.

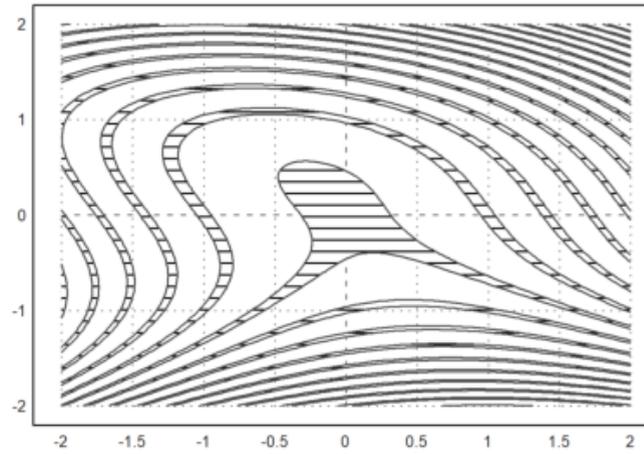
```
>plot2d("x^4+y^4",r=1.5,level=[0;1],color=blue,style="/"):
```



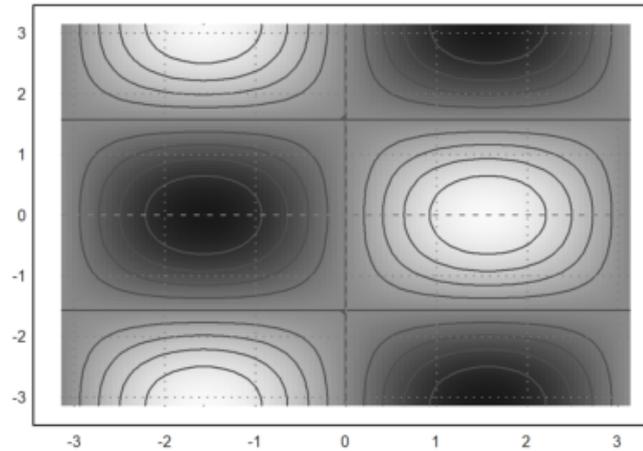
```
>plot2d("x^2+y^3+x*y",level=[0,2,4;1,3,5],style="/",r=2,n=100):
```



```
>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):
```

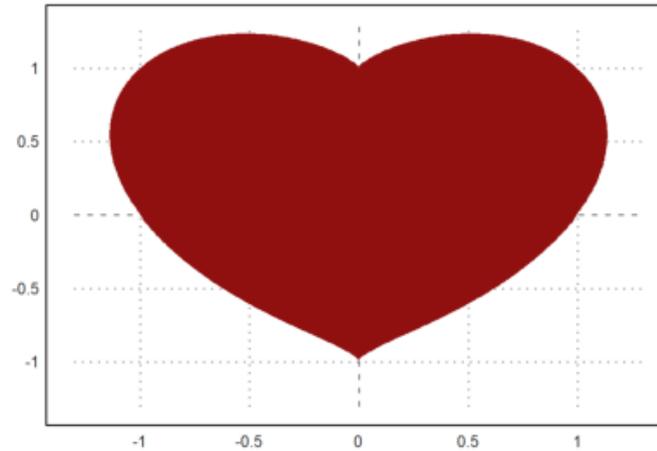


Dimungkinkan juga untuk menandai suatu wilayah

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

Ini dilakukan dengan menambahkan level dengan dua baris.

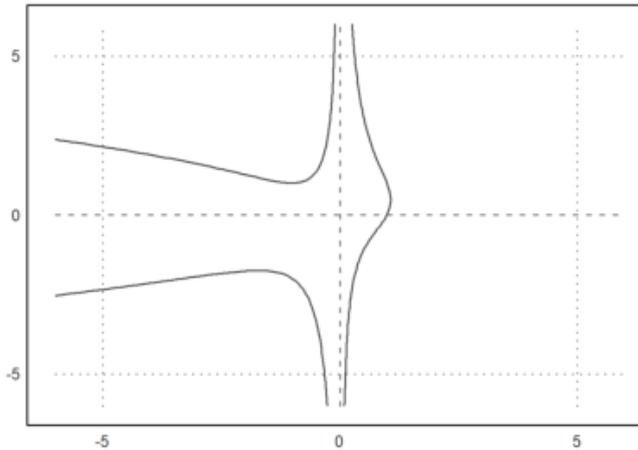
```
> plot2d("(x^2+y^2-1)^3-x^2*y^3",r=1.3, ...
> style="#",color=red,<outline, ...
> level=[-2;0],n=100):
```



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

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

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



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

```

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

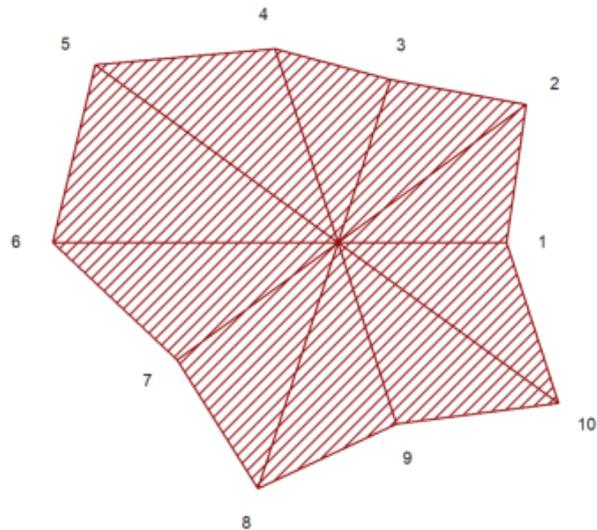
```

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

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

Kami memanggil reset sebelum kami menguji plot ini untuk mengembalikan default grafis. Ini tidak perlu, jika Anda yakin plot Anda berhasil.

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



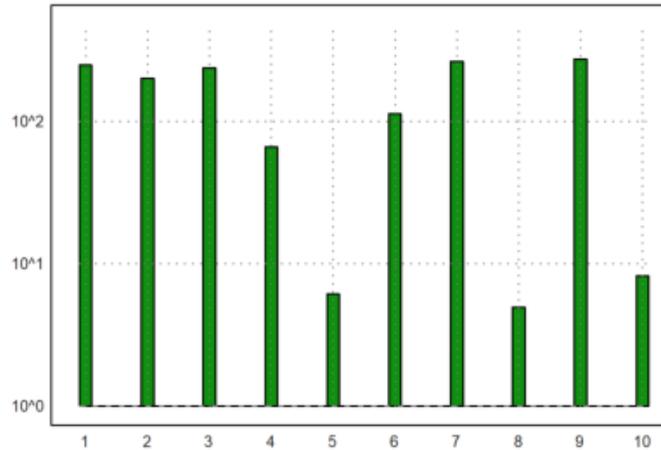
Terkadang, Anda mungkin ingin merencanakan sesuatu yang tidak dapat dilakukan plot2d, tetapi hampir. Dalam fungsi berikut, kami melakukan plot impuls logaritmik. plot2d dapat melakukan plot logaritmik, tetapi tidak untuk batang impuls.

```
>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 yang terdistribusi secara eksponensial.

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



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

Fungsi `wait(0)` memaksa plot untuk muncul di jendela grafik. Jika tidak, menggambar ulang 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;

```

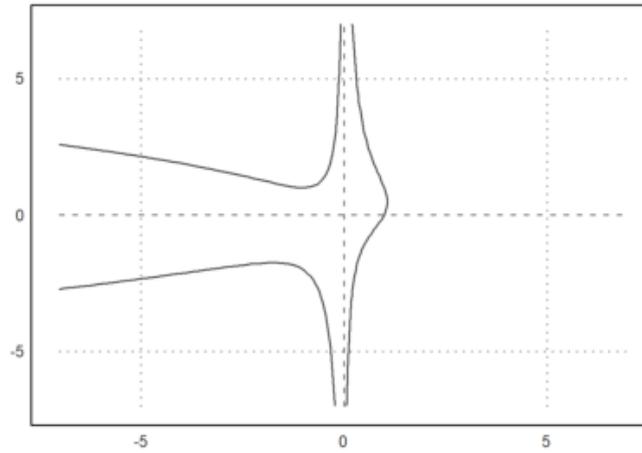
```
plot(sin(n*t),cos(m*t+f));  
wait(0);  
if testkey() then break; endif;  
f=f+0.02;  
end;  
framecolor(c);  
linewidth(1);  
endfunction
```

Tekan sembarang tombol untuk menghentikan animasi ini.

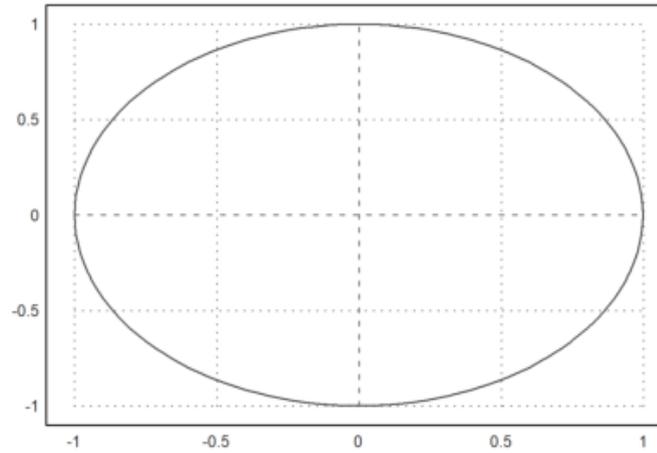
```
>animliss(2,3); // lihat hasilnya, jika sudah puas, tekan ENTER
```

CONTOH LAIN:

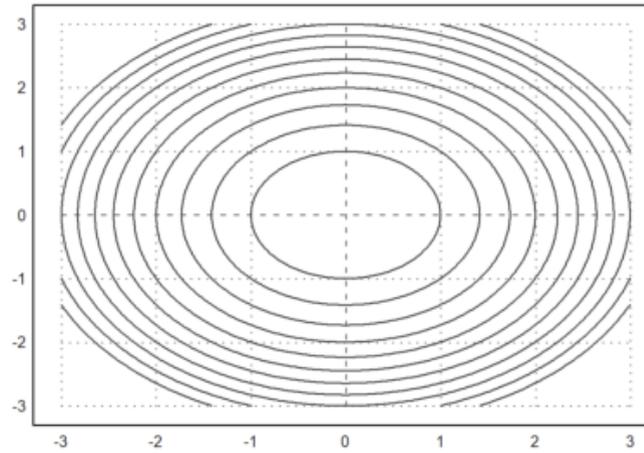
```
>plot2d("x^3-x*y+x^2*y^2", r=7, level=1, n=100):
```



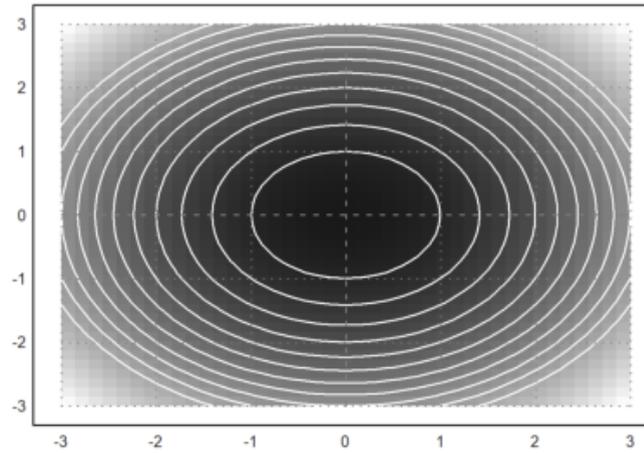
```
>plot2d("x^2+y^2-1", level=0):
```



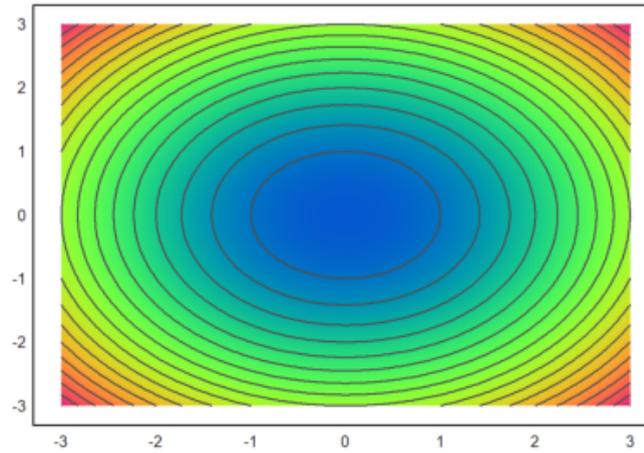
```
>plot2d("x^2+y^2-1", r=3, level=0:1:10, n=200):
```



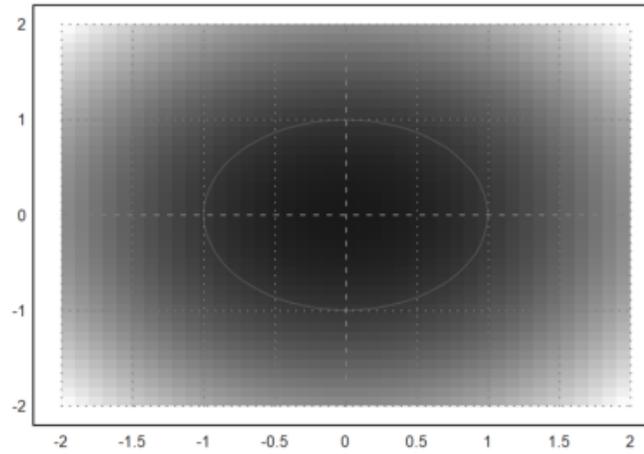
```
>plot2d("x^2+y^2-1", r=3, level=0:1:10, >hue, contourcolor=white):
```



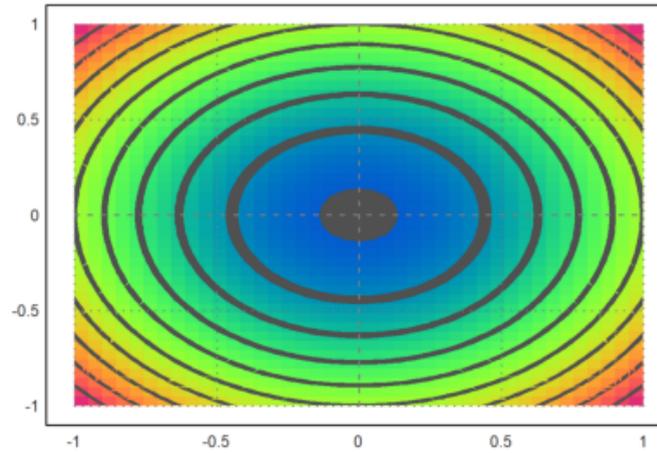
```
>plot2d("x^2+y^2-1", r=3, level=0:1:20, >hue, >spectral, n=200, grid=4):
```



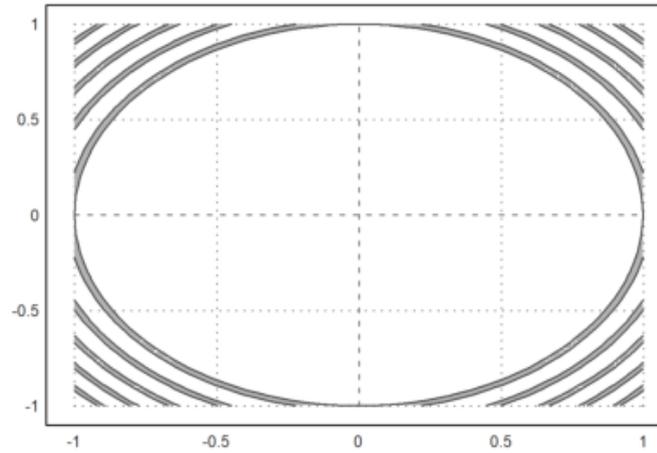
```
>plot2d("x^2+y^2-1",level=0,a=-2,b=2,c=-2,d=2,>hue):
```



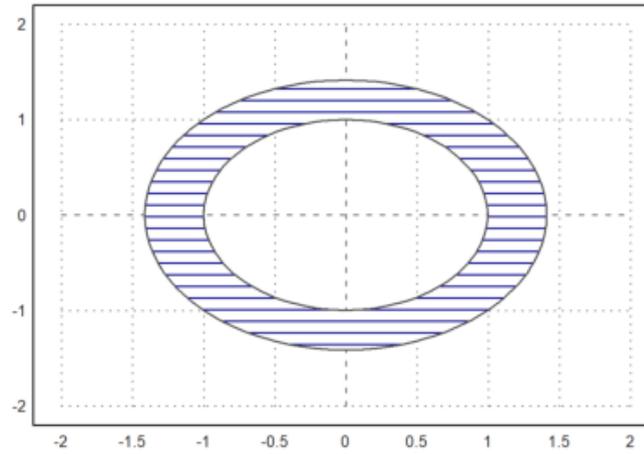
```
>plot2d("x^2+y^2-1",>contour,>hue,>spectral):
```



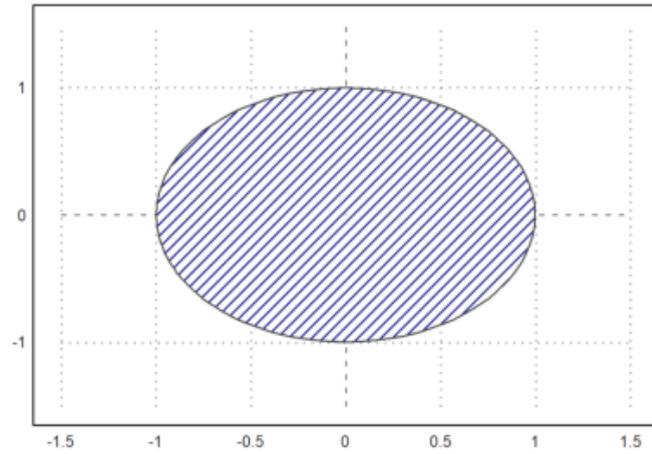
```
>plot2d("x^2+y^2-1",level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
```



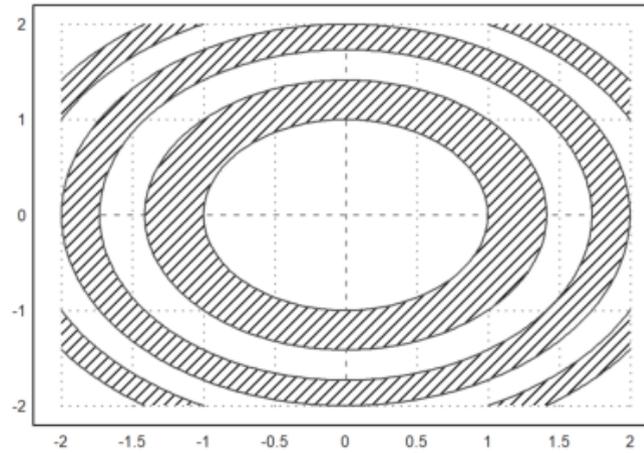
```
>plot2d("x^2+y^2-1",r=2,level=[0;1],style="-",color=blue): // 0 <= f(x,y) <= 1
```



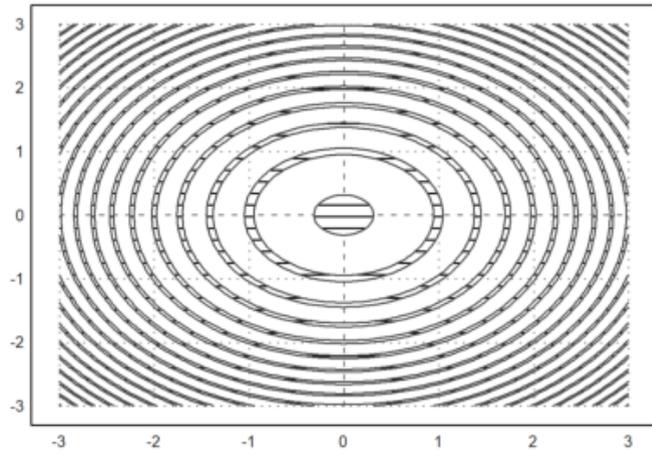
```
>plot2d("x^2+y^2",r=1.5,level=[0;1],color=blue,style="/"):
```



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



```
>plot2d("x^2+y^2-1",level=-10:20,r=3,style="-",dl=0.1,n=100):
```



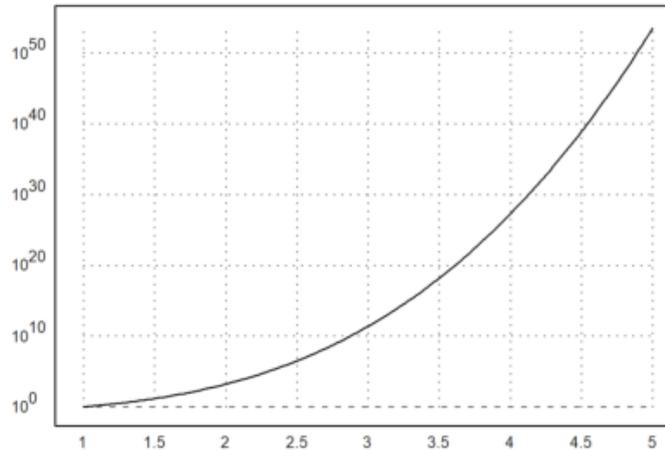
Plot Logaritmik

EMT menggunakan parameter "logplot" untuk skala logaritmik.

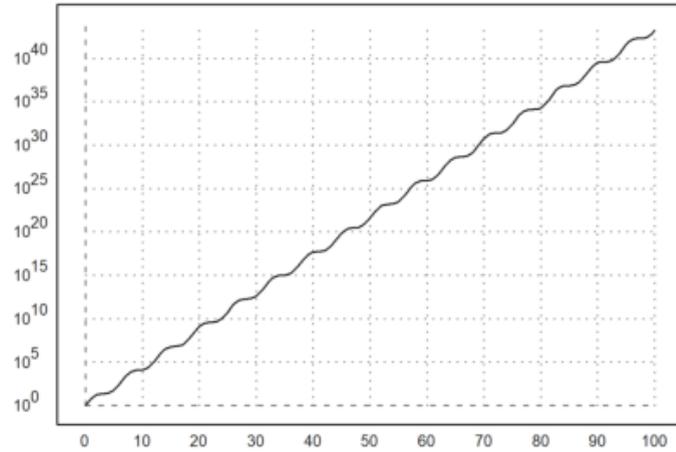
lot logaritma dapat diplot baik menggunakan skala logaritma dalam y dengan logplot=1, atau menggunakan skala logaritma dalam x dan y dengan logplot=2, atau dalam x dengan logplot=3.

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

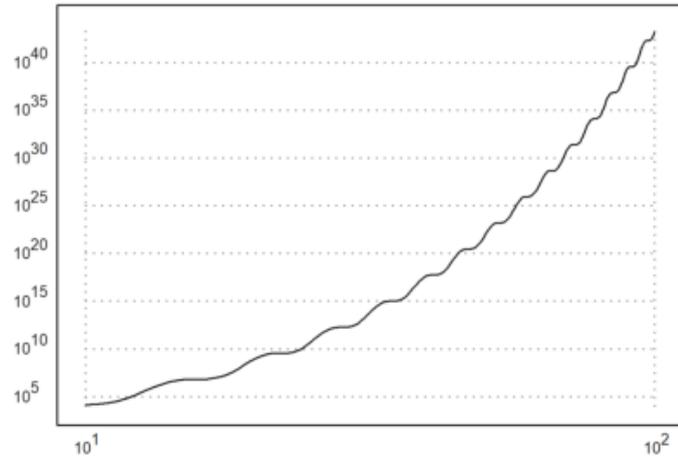
```
>plot2d("exp(x^3-x)*x^2",1,5,logplot=1):
```



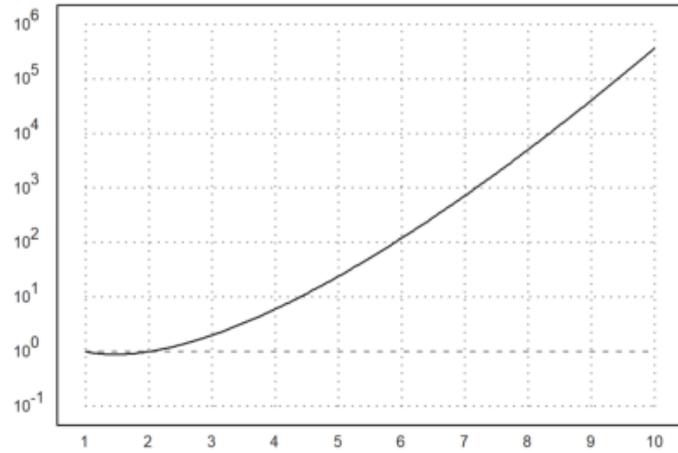
```
>plot2d("exp(x+sin(x))",0,100,logplot=1):
```



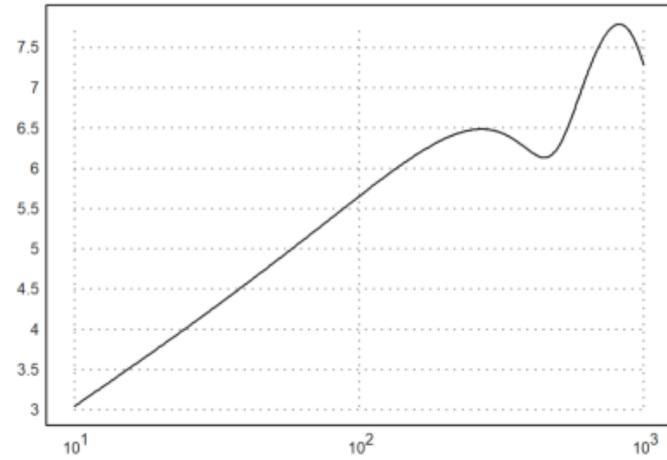
```
>plot2d("exp(x+sin(x))",10,100,logplot=2):
```



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

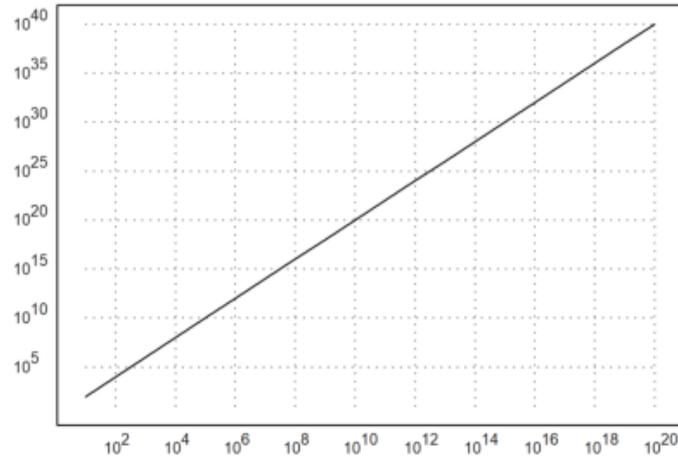


```
>plot2d("log(x*(2+sin(x/100)))",10,1000,logplot=3):
```



Ini juga berfungsi dengan plot data.

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



MATERI TAMBAHAN

MATERI TAMBAHAN

Rujukan Lengkap Fungsi plot2d()

fungsi 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)

Fungsi plot serbaguna untuk plot pada bidang (plot 2D). Fungsi ini dapat membuat plot fungsi satu variabel, plot data, kurva pada bidang, plot batang, kisi bilangan kompleks, dan plot implisit fungsi dua variabel.

Parameter

x,y : persamaan, fungsi atau vektor data

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

r : jika r ditetapkan, maka $a=cx-r$, $b=cx+r$, $c=cy-r$, $d=cy+r$

dapat berupa vektor $[rx,ry]$ atau vektor $[rx1,rx2,ry1,ry2]$.

$xmin,xmax$: rentang parameter untuk kurva

$auto$: Tentukan rentang y secara otomatis (default)

$square$: jika benar, cobalah untuk mempertahankan rentang $x-y$ yang persegi

n : jumlah interval (default adalah adaptif)

$grid$: 0 = tidak ada grid dan label,

1 = hanya sumbu,

2 = grid normal (lihat di bawah untuk jumlah garis grid)

3 = di dalam sumbu

4 = tidak ada kisi

5 = kisi penuh termasuk margin

6 = anda centang pada bingkai

7 = hanya sumbu

8 = hanya sumbu, sub-centang

frame : 0 = tanpa bingkai
framecolor: warna bingkai dan kisi
margin : angka antara 0 dan 0,4 untuk margin di sekitar plot
color : Warna kurva. Jika ini adalah vektor warna,
ni akan digunakan untuk setiap baris matriks plot. Dalam kasus
lot titik, ini harus berupa vektor kolom. Jika vektor baris atau
atriks penuh warna digunakan untuk plot titik, ini akan digunakan untuk
etiap titik data.
thickness : ketebalan garis untuk kurva

Nilai ini dapat lebih kecil dari 1 untuk garis yang sangat

tipis
style : Gaya plot untuk garis, penanda, dan isian.

Untuk titik gunakan
"[]", "<>", ".", "..", "...",
"*", "+", "|", "-", "o"
"[]#", "<>#", "o#" (bentuk yang terisi)
"[]w", "<>w", "ow" (tidak transparan)
Untuk garis gunakan
"-", "--", "-.", ".", "-.", "-.-", "->"
Untuk poligon atau diagram batang yang terisi gunakan
"#", "#0", "0", "/", "\", "\/",
"+", "|", "-", "t"

points : plot titik tunggal alih-alih segmen garis
addpoints : jika benar, plot segmen garis dan titik
add : menambahkan plot ke plot yang sudah ada
user : mengaktifkan interaksi pengguna untuk fungsi
delta : ukuran langkah untuk interaksi pengguna
bar : plot batang (x adalah batas interval, y adalah nilai interval)
histogram : plot frekuensi x dalam n subinterval
distribution=n : plot distribusi x dengan n subinterval
even : menggunakan nilai inter untuk histogram otomatis.
steps : plot fungsi sebagai fungsi langkah (steps=1,2)
adaptive : menggunakan plot adaptif (n adalah jumlah langkah minimal)
level : plot garis level dari fungsi implisit dua variabel
outline : menggambar batas rentang level.
Jika nilai level adalah matriks $2 \times n$, rentang level akan digambar
alam warna menggunakan gaya isian yang diberikan. Jika outline benar, maka akan digambar
alam warna kontur. Dengan menggunakan fitur ini, daerah
(x,y) di antara batas dapat ditandai.
hue : menambahkan warna hue ke plot level untuk menunjukkan fungsi
ilai
contour : menggunakan plot level dengan level otomatis
nc : jumlah garis level otomatis
title : judul plot (default "")
xl, yl : label untuk sumbu x dan y
smaller : jika >0 , akan ada lebih banyak ruang di sebelah kiri untuk label.
vertical :

Mengaktifkan atau menonaktifkan label vertikal. Ini mengubah

variabel global

verticallabels secara lokal untuk satu plot. Nilai 1 hanya menetapkan teks vertikal, nilai 2 menggunakan label numerik vertikal pada sumbu y.

filled : mengisi plot kurva

fillcolor : mengisi warna untuk kurva batang dan isi kurva batang

outline : batas untuk poligon terisi

logplot : mengatur plot logaritmik

```
1 = logplot dalam y,  
2 = logplot dalam xy,  
3 = logplot dalam x
```

own :

Sebuah string, yang menunjuk ke rutinitas plot sendiri. Dengan

>user, Anda mendapatkan

interaksi pengguna yang sama seperti di plot2d. Rentang akan ditetapkan sebelum setiap panggilan ke fungsi Anda.

maps : ekspresi peta (0 lebih cepat), fungsi selalu dipetakan.

contourcolor : warna garis kontur

contourwidth : lebar garis kontur

clipping : mengaktifkan kliping (default adalah benar)

title :

Ini dapat digunakan untuk mendeskripsikan plot. Judul akan muncul di

atas

lot. Selain itu, label untuk sumbu x dan y dapat ditambahkan dengan `xl="string"` atau `yl="string"`. Label lain dapat ditambahkan dengan fungsi `label()` atau `labelbox()`. Judul dapat berupa string unicode atau gambar rumus Latex.

`cgrid` :

Menentukan jumlah garis grid untuk plot grid kompleks.

Harus berupa pembagi ukuran matriks dikurangi 1 (jumlah subinterval). `cgrid` dapat berupa vektor `[cx,cy]`.

Ringkasan

Fungsi dapat memplot

- ekspresi, memanggil koleksi atau fungsi dari satu variabel,
- kurva parametrik,
- data x terhadap data y,
- fungsi implisit,
- plot batang,
- kisi kompleks,
- poligon.

Jika fungsi atau ekspresi untuk xv diberikan, `plot2d()` akan menghitung nilai dalam rentang yang diberikan menggunakan fungsi atau ekspresi tersebut. Ekspresi tersebut harus berupa ekspresi dalam variabel x . Rentang tersebut harus didefinisikan dalam parameter a dan b kecuali rentang default `[-2,2]` harus digunakan. Rentang y akan dihitung secara otomatis, kecuali c dan d ditentukan, atau radius r , yang menghasilkan rentang `[-r,r]` untuk x dan y . Untuk plot fungsi, `plot2d` akan menggunakan evaluasi adaptif fungsi secara default. Untuk mempercepat plot untuk fungsi yang rumit, matikan ini dengan `<adaptive`, dan secara opsional kurangi jumlah interval n . Selain itu, `plot2d()` secara default akan menggunakan pemetaan. Yaitu, ia akan menghitung elemen plot untuk elemen. Jika ekspresi atau fungsi Anda dapat menangani vektor x , Anda dapat mematakannya dengan `<maps` untuk evaluasi yang lebih cepat.

Perhatikan bahwa plot adaptif selalu dihitung elemen per elemen.

Jika fungsi atau ekspresi untuk xv dan yv ditentukan, `plot2d()` akan menghitung kurva dengan nilai

xv sebagai koordinat x dan nilai yv sebagai koordinat y. Dalam kasus ini, rentang harus ditentukan untuk parameter menggunakan xmin, xmax. Ekspresi yang terkandung dalam string harus selalu berupa ekspresi dalam variabel parameter x.

MATERI TAMBAHAN

Mengatur Ukuran Gambar, Format (Style), dan Warna Kurva

1. Mengatur Ukuran Gambar

Dalam mengatur ukuran gambar akan dibahas cara menyesuaikan aspek rasio yang sesuai dengan data yang disajikan. Plot di EMT dapat muncul di jendela terpisah atau jendela notebook, untuk yang muncul di jendela terpisah ukurannya dapat diubah-ubah.

Untuk plot kuadrat memiliki ukuran default. Namun, dapat diubah dengan menu "Set Aspect" ke rasio aspek tertentu. Selain itu, dapat juga hanya mengubah satu plot. Misalkan `aspect(2)` maka rasio dari gambar tersebut adalah 4:2.

2. Mengatur Format (Style)

Dalam EMT format (style) yang digunakan pada garis kurva ada beberapa pilihan seperti,

1. "_"
2. "-"
3. "-."
4. "."
5. "-."
6. "-."

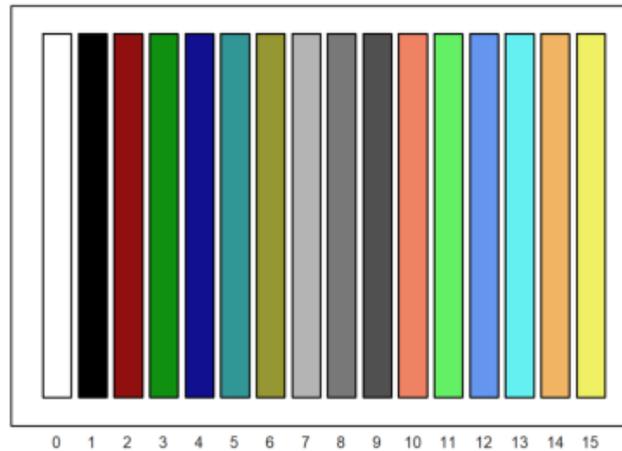
Untuk menggunakan style tersebut digunakan perintah style="..."

3. Mengatur Warna Kurva

EMT memiliki indeks warna default dengan skala 0-15. Tidak semua warna ada di EMT ini.

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

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



berikut warna konstan yang ada di EMT.

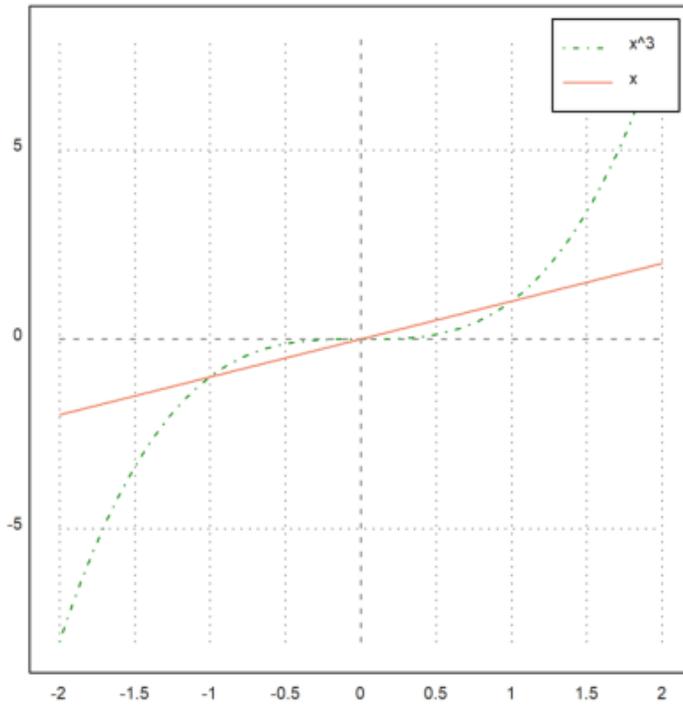
Contoh mengatur ukuran gambar, style, dan warna kurva

Buatkan kurva dengan fungsi berikut:

1. $y = x^3$ dengan $g = x$

Kemudian berikan ukuran, format (style), dan warna kurvanya.

```
>plot2d(["x^3","x"], color=[green, orange], style=[".-","-"]);  
>labelbox(["x^3","x"], colors=[green, orange], styles=[".-","-"]);  
>aspect(1): // rasio 2:1
```

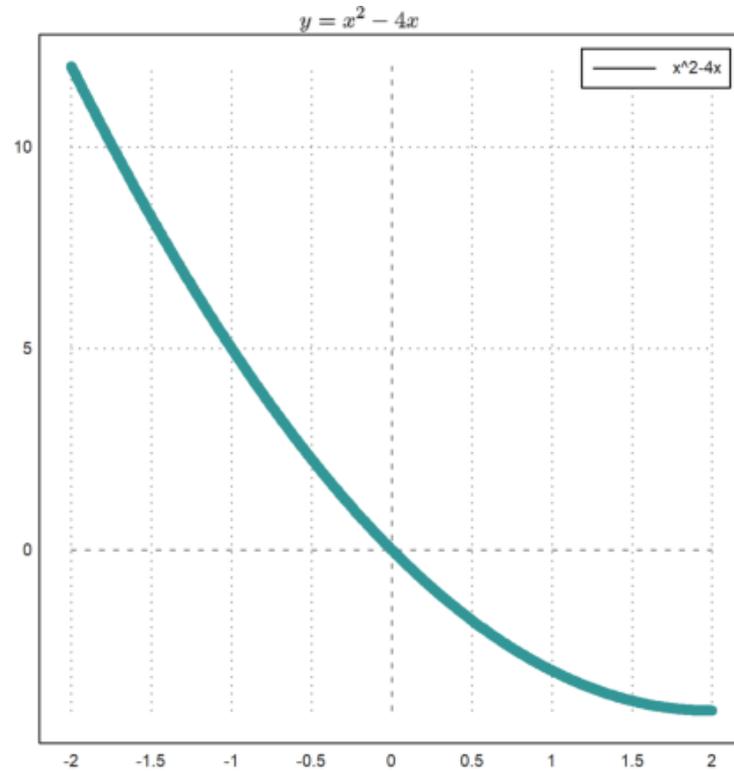


Plot dengan rasio 2:1, dapat diubah rasionya dengan mengubah aspect().

2. kurva fungsi

$$y = x^2 - 4x$$

```
>aspect(1); plot2d("x^2-4*x",>user, title=latex("y=x^2-4x"), color=5, thickness=7);...  
>labelbox("x^2-4x"); insimg;
```



Menggambar Segi Banyak

Poligon atau segi banyak adalah bangun datar yang digambarkan dengan jumlah terhingga dari garis lurus yang terhubung, sehingga membentuk sebuah rantai poligonal (atau sirkuit poligonal) yang tertutup.

Ruas garis dari sirkuit poligonal disebut sebagai sisi. Perpotongan dari dua sisi pada poligon dikenal sebagai titik sudut.

Segi- n adalah sebuah poligon yang mempunyai n sisi, contohnya, segi-3(segitiga). Segi banyak memiliki paling sedikit tiga sisi.

Terdapat dua macam segi banyak, yaitu:

1. Segi banyak beraturan
2. Segi banyak tidak beraturan

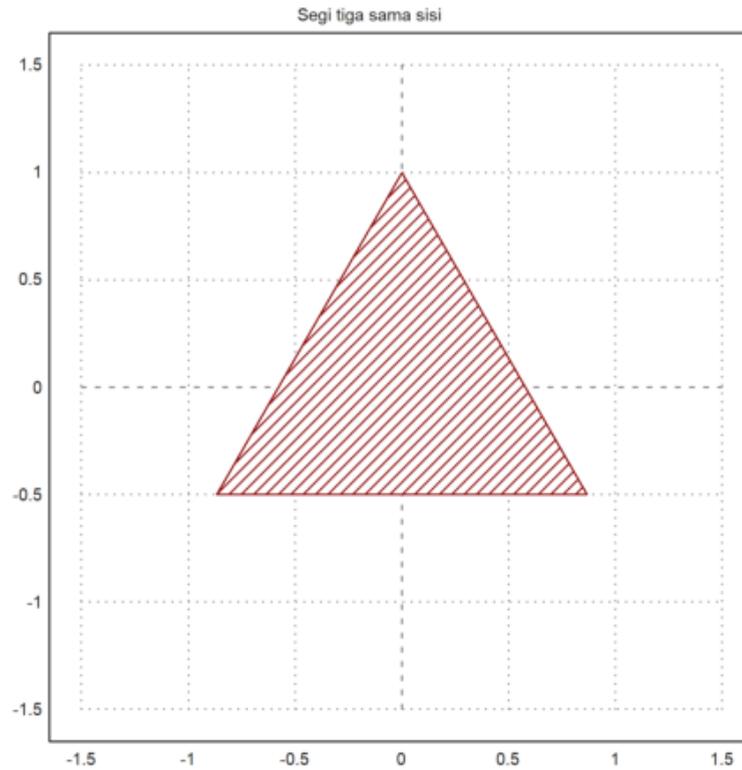
Segi banyak beraturan

Segi banyak beraturan merupakan suatu bangun yang ukuran semua sisinya sama panjang dan besar sudutnya juga sama.

Contoh dari segi banyak beraturan adalah bangun segi tiga sama sisi, persegi, segi lima beraturan, segi enam beraturan, dst.

Contoh gambar

```
>t=linspace(0,2pi,3);...
>plot2d(sin(t),cos(t),title="Segi tiga sama sisi",>filled,style="/",fillcolor=red,r=1.5):
```



Plot di atas membentuk segi tiga sama sisi.

Mengubah style plot:

- `filled=true` mengisi plot.
- `style = "..."`: Pilih dari `" "`, `"O"`, `"o"`, `"/"`, `"\"`, `"\"`, `"+"`, `"|"`, `"-"`, `"t"`. (style untuk isian poligon atau plot bar)
- `fillcolor`: dapat dilihat di bawah untuk warna yang tersedia.

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

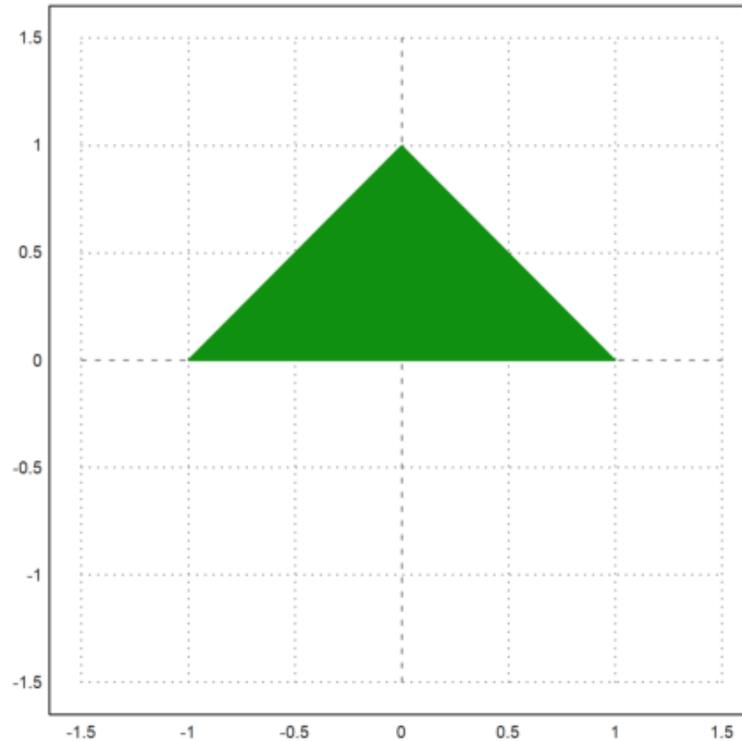
- 0..15: indeks warna default.

- konstanta warna: white, black, red, green, blue, cyan, olive, lightgray, gray, darkgray, orange, lightgreen, turquoise, lightblue, lightorange, yellow

- rgb (red,green,blue): parameter dalam bentuk real dalam [0,1].

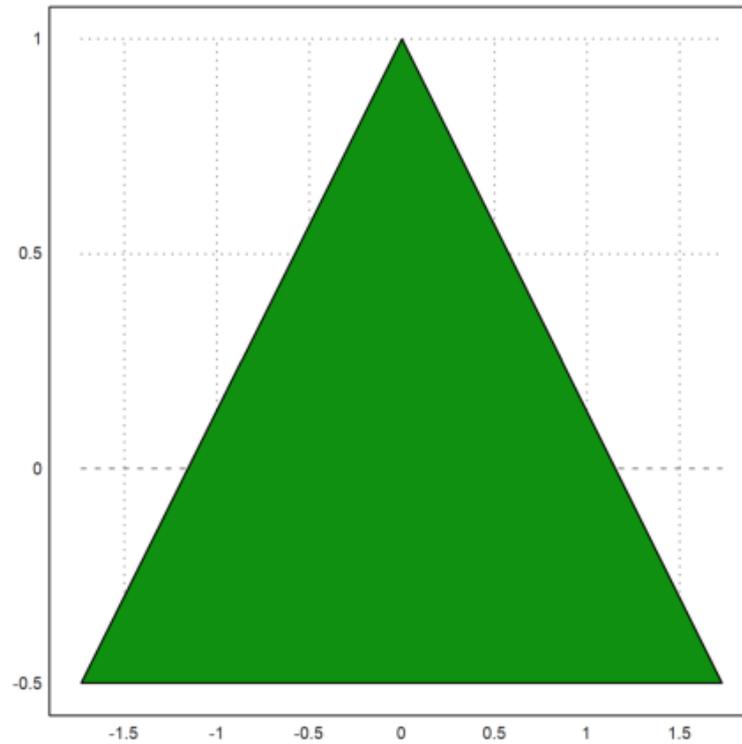
Warna isian ditentukan oleh argumen "fillcolor", jika hanya ditulis "filled" tanpa menambahkan "fillcolor" maka warna defaultnya adalah hijau. Lalu pada fungsi <outline untuk menghilangkan garis batas untuk semua gaya kecuali yang default.

```
>t=linspace(0,pi,2);...  
>plot2d(cos(t),sin(t),>filled,style="#",r=1.5):
```



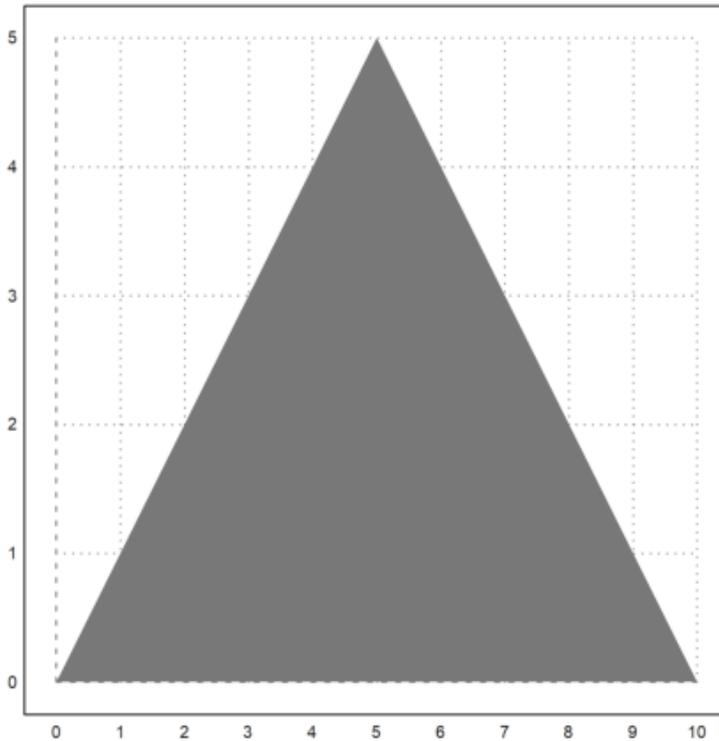
Dengan parameter $t=\text{linspace}(0,\pi,2)$ menunjukkan bahwa terbentuk 2 titik dalam setengah lingkaran (karena batas t dari 0 hingga π) sehingga terbentuk segitiga sama kaki. Namun, segitiga sama kaki bukan termasuk segi banyak beraturan karena hanya 2 sisi yang sama panjang.

```
>t=linspace(0,2pi,3);...  
>plot2d(tan(t),cos(t),>filled,style="#0"):
```



Dihasilkan bentuk segitiga, namun bukan sama sisi melainkan sama kaki.

```
>plot2d([0,5,10],[0,5,0],>filled,fillcolor=gray,<outline):
```



Untuk menggambar segi banyak dapat juga menggunakan titik-titik koordinat yang saling dihubungkan. Plot di atas dengan menghubungkan titik $(0,0)$, $(5,5)$, $(10,0)$ maka terbentuk segi tiga sama kaki. Namun bangun ini tidak termasuk segi banyak beraturan karena ada sisi yang panjangnya berbeda.

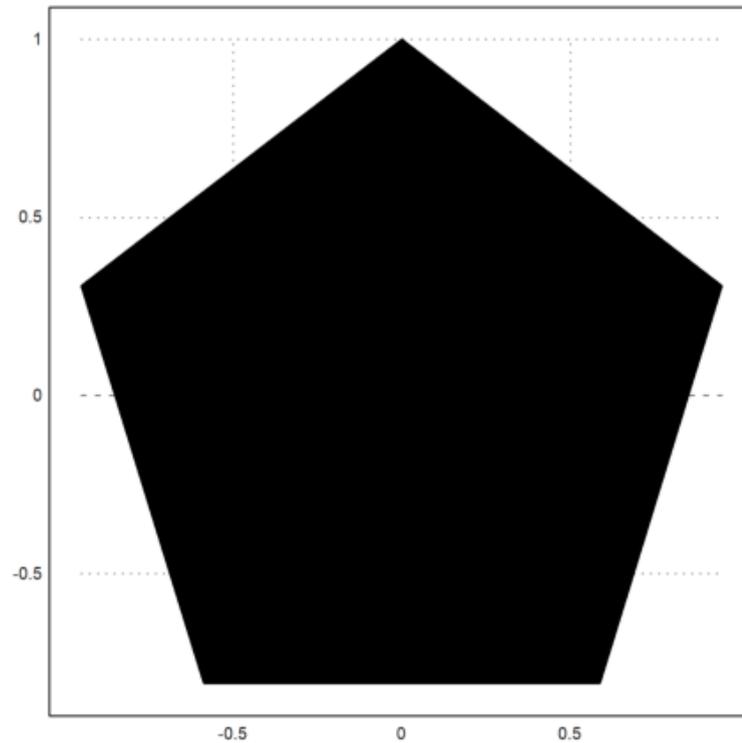
Untuk menghilangkan outline, kita dapat menambahkan "`<outline`" atau menggunakan `style="O"` dengan isian tanpa outline.

```
>plot2d([1,3,3,1],[1,1,3,3],>filled,fillcolor=blue,<outline,grid=0):
```



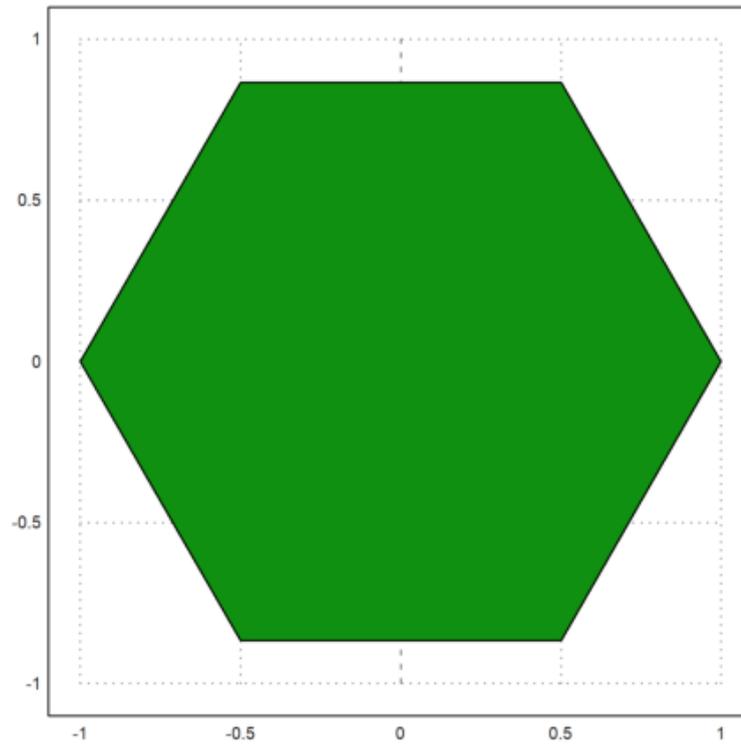
Dengan menghubungkan titik-titik (1,1),(3,1),(3,3),(1,3) maka terbentuk persegi. Pada plot di atas, grid=0 membuat plot tidak menampilkan grid koordinat.

```
>t=linspace(0,2pi,5);...  
>plot2d(sin(t),cos(t),>filled,fillcolor=black):
```



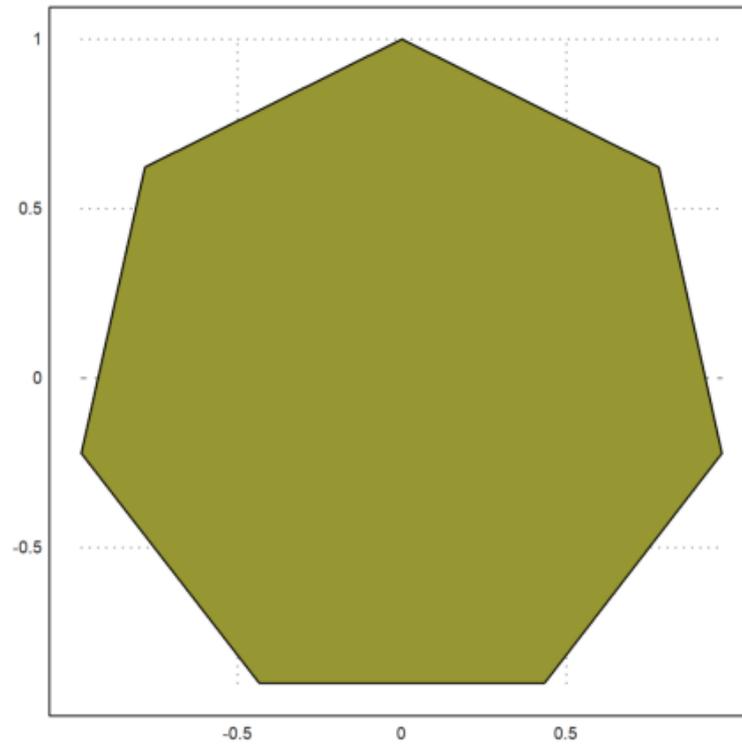
Terbentuk segi lima sama sisi.

```
>t=linspace(0,2pi,6);...  
>plot2d(cos(t),sin(t),>filled,r=1):
```



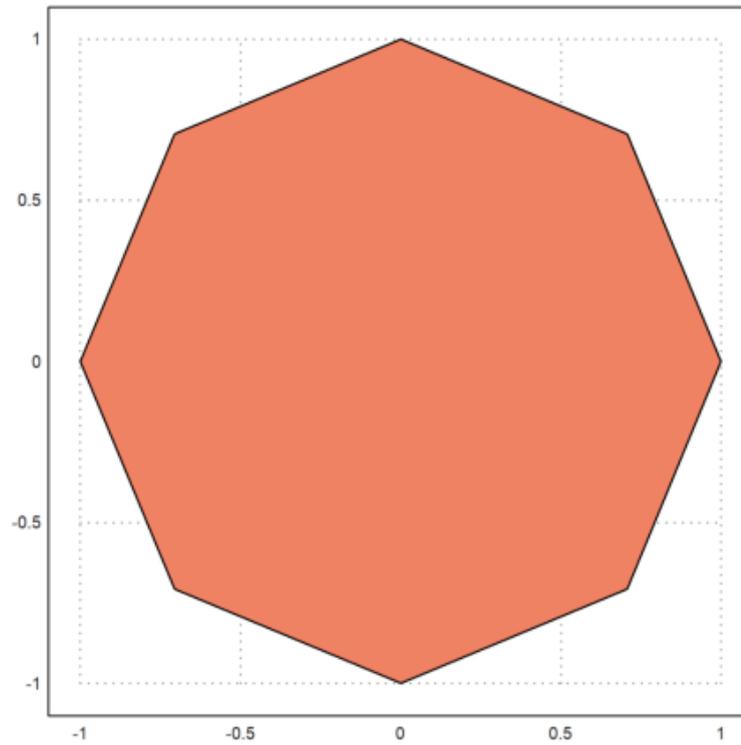
Terbentuk segi enam sama sisi.

```
>t=linspace(0,2pi,7);...  
>plot2d(sin(t),cos(t),>filled,fillcolor=olive):
```



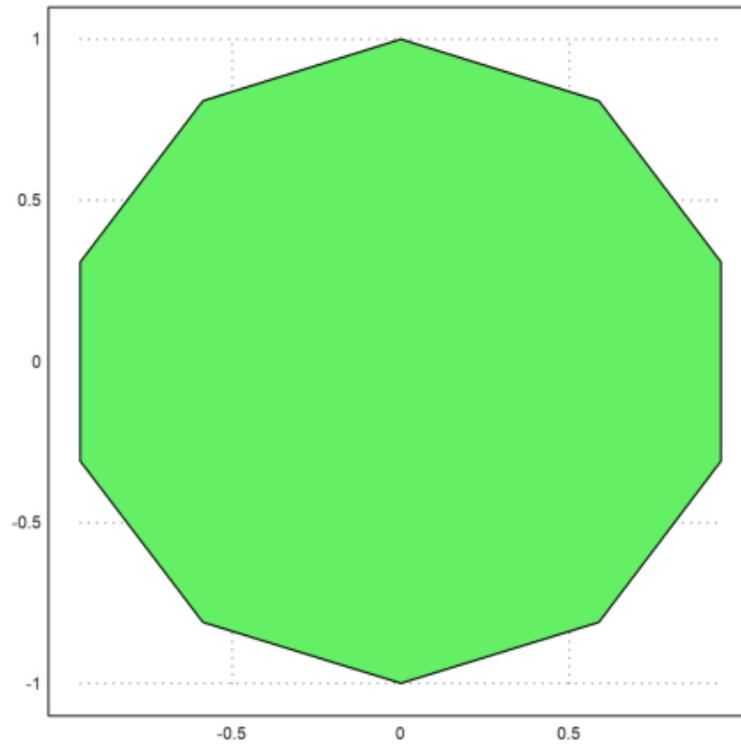
Terbentuk segi tujuh sama sisi.

```
>t=linspace(0,2pi,8);...  
>plot2d(sin(t),cos(t),>filled,fillcolor=orange):
```



Terbentuk segi delapan sama sisi.

```
>t=linspace(0,2pi,10);...  
>plot2d(sin(t),cos(t),>filled,fillcolor=lightgreen):
```



Terbentuk segi sepuluh sama sisi.

Dari beberapa contoh di atas, dapat dilihat bahwa parameter $t=\text{linspace}(a,b,n)$ dengan fungsi $\sin(t),\cos(t)$ menunjukkan

a = sebagai batas bawah t

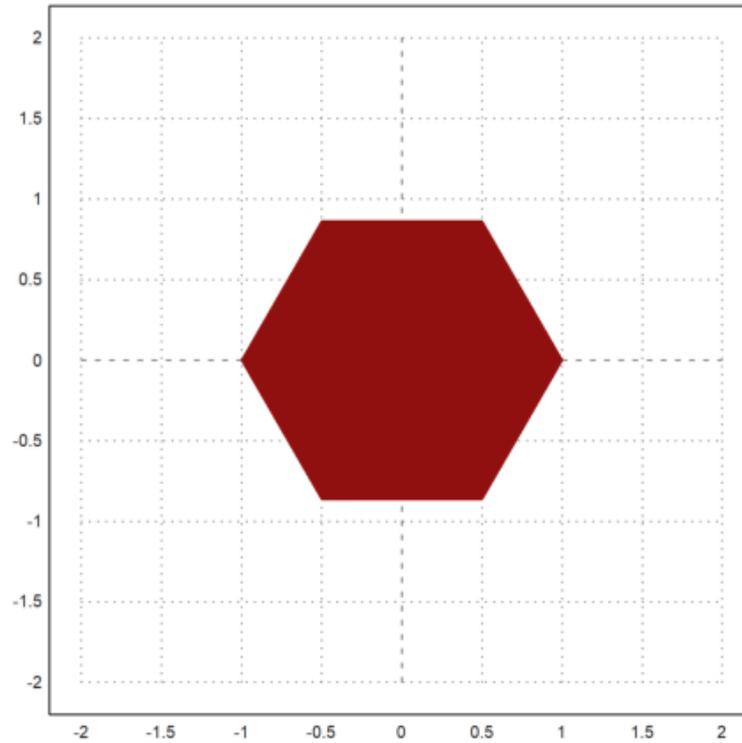
b = batas atas t

n = banyak titik pada lingkaran satuan atau bisa dikatakan n banyak sisi yang terbentuk.

Parameter t dengan $(0,2\pi,n)$ dan fungsi $\sin(t),\cos(t)$ akan menghasilkan segi banyak dengan banyak sisi n dan panjang sisi serta besar sudutnya sama.

Cara lain menggambar segi enam:

```
>t=linspace(0,3pi,9);...  
>plot2d(cos(t),sin(t),>filled,style="#",fillcolor=red,r=2):
```



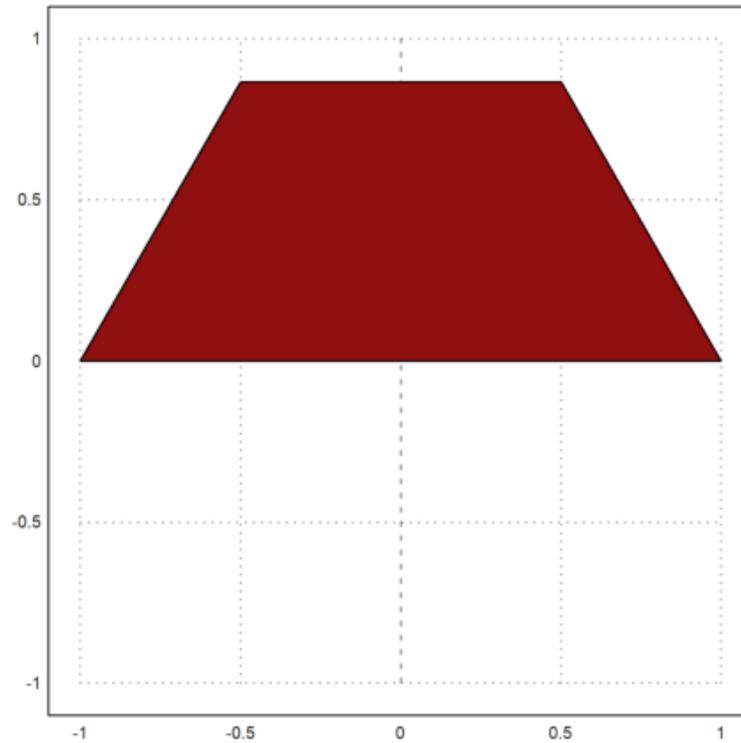
r diartikan sebagai jarak sumbu x positif, sumbu x negatif, sumbu y positif, dan sumbu y negatif dari titik pusat sejauh r .

Segi banyak tidak beraturan

Segi banyak tidak beraturan adalah segi banyak yang sisi-sisinya tidak sama panjang atau sudut-sudutnya tidak sama besar.

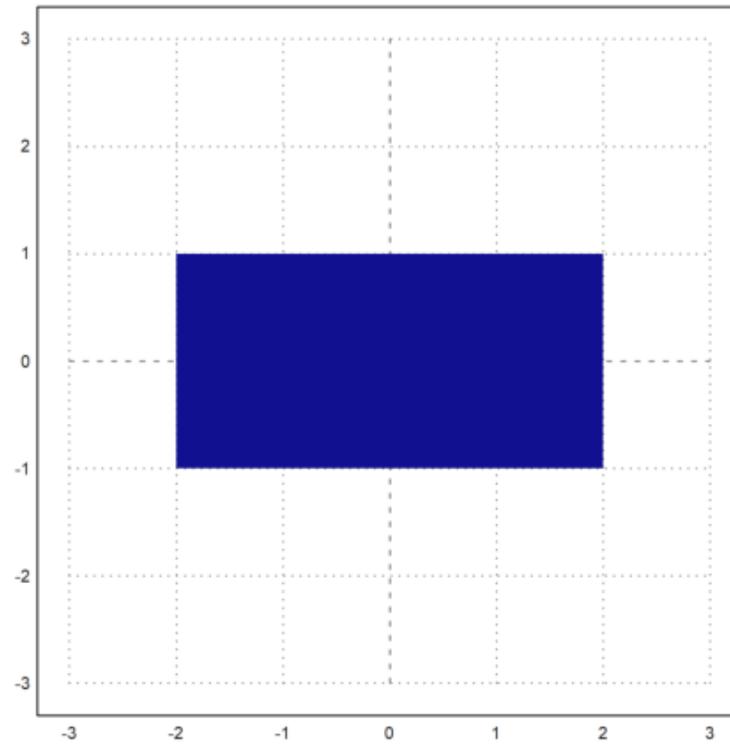
Contoh: segitiga sembarang, persegi panjang, trapesium, dan sebagainya.

```
>t=linspace(0,pi,3);...  
>plot2d(cos(t),sin(t),>filled,fillcolor=red,r=1):
```



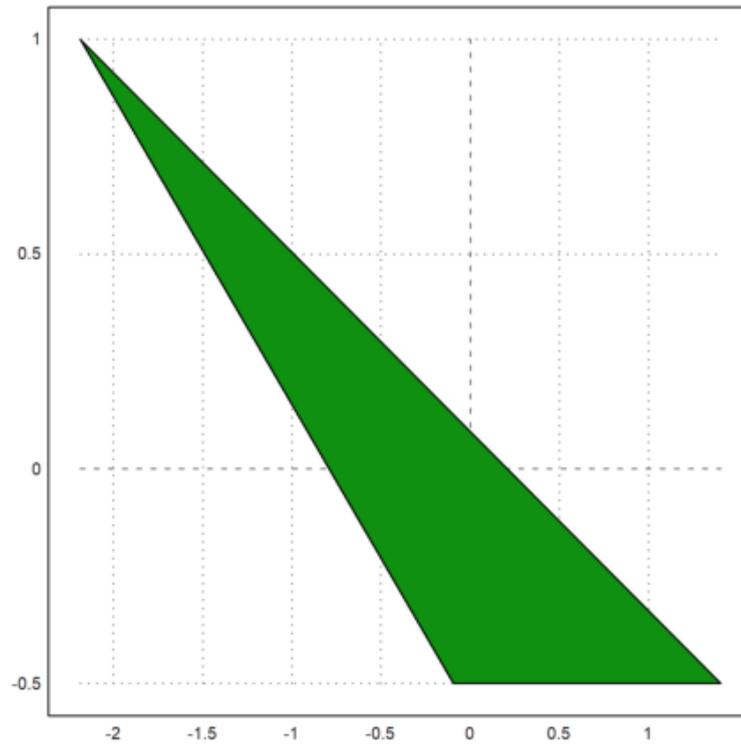
Terbentuk trapesium sama kaki.

```
>plot2d([-2,2,2,-2],[-1,-1,1,1],>filled,fillcolor=blue,<outline,r=3):
```



Dengan menghubungkan titik titik $(-2,-1),(2,-1),(2,1),(-2,1)$ terbentuk bangun persegi panjang.

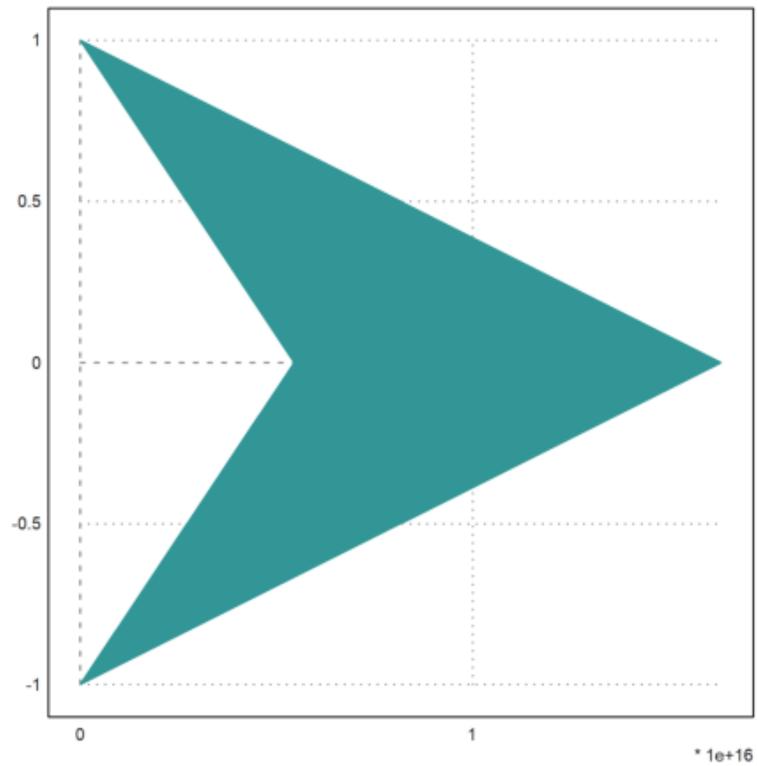
```
>t=linspace(0,2pi,3);...  
>plot2d(tan(t+2),cos(t),>filled,fillcolor=green):
```



Terbentuk segitiga sembarang.

Contoh lain segi banyak tidak beraturan yaitu:

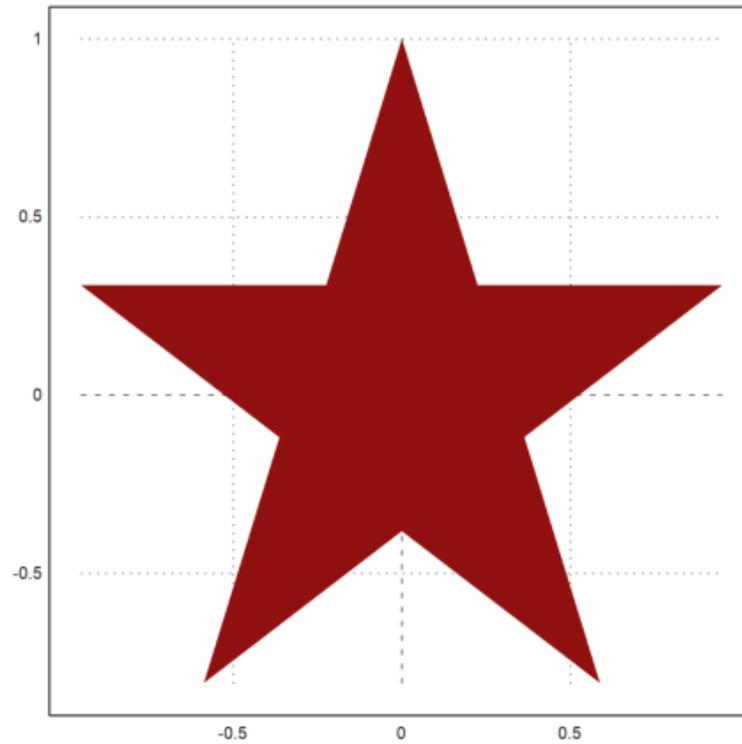
```
>t=linspace(0,2pi,4);...  
>plot2d(tan(t),cos(t),>filled,style="#",fillcolor=cyan):
```



1. Gambarkan bangun segi banyak bintang dalam plot2d

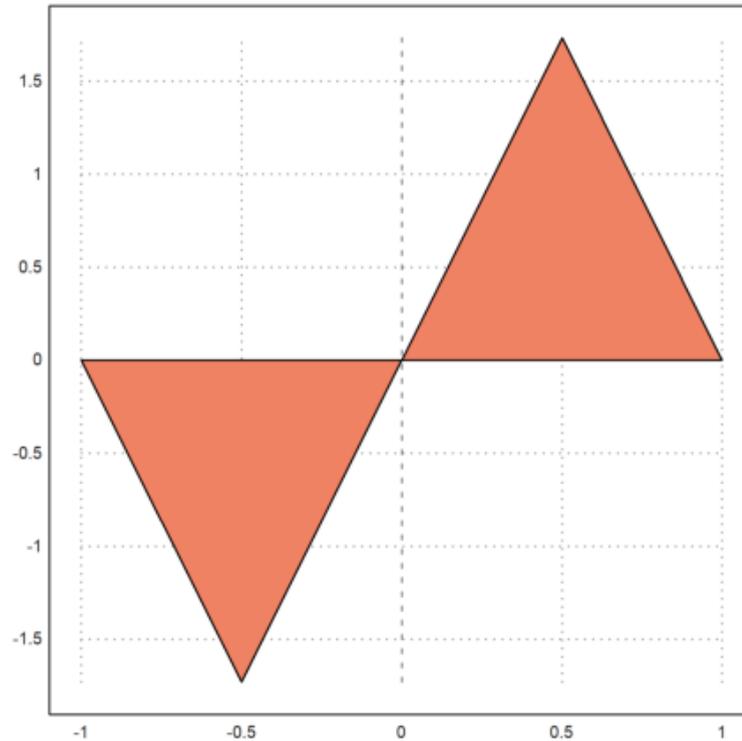
Penyelesaian:

```
>t=linspace(0,4pi,5);...  
>plot2d(sin(t),cos(t),>filled,fillcolor=red,<outline):
```



2. Gambarkan 2 segitiga yang salah satu sudut dari kedua segitiga saling bertolak belakang.
Penyelesaian:

```
>t=linspace(0,pi,3);.....  
>plot2d(cos(t),tan(t),>filled,fillcolor=orange):
```



>

menggunakan perintah >user dapat dilakukan untuk memperbesar dan memperkecil intervalnya (grafiknya), dengan menekan '-' untuk memperkecil dan 'shift +' untuk memperbesar. Jika grafiknya ingin ditampilkan pada notebook maka perlu menambahkan perintah insimg. Insimg secara default berukuran 35.