

Tugas Individu Pekan5-6_Muhammad Lutfi Ramadhan_23030630021

Nama : Muhammad Lutfi Ramadhan
Kelas : Matematika B 2023
NIM : 23030630021

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

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

```
>clg; // hapus layar
>window(0,0,1024,1024); // gunakan seluruh jendela
>setplot(0,1,0,1); // atur koordinat plot
>hold on; // aktifkan mode overwrite
>n=100; X=random(n,2); Y=random(n,2); // mendapatkan poin acak
>colors=rgb(random(n),random(n),random(n)); // mendapatkan warna acak
>loop 1 to n; color(colors[#]); plot(X[#],Y[#]); end; // plot
>hold off; // akhiri mode overwrite
>insimg; // masukkan gambar ke dalam notebook
```



```
>reset;
```

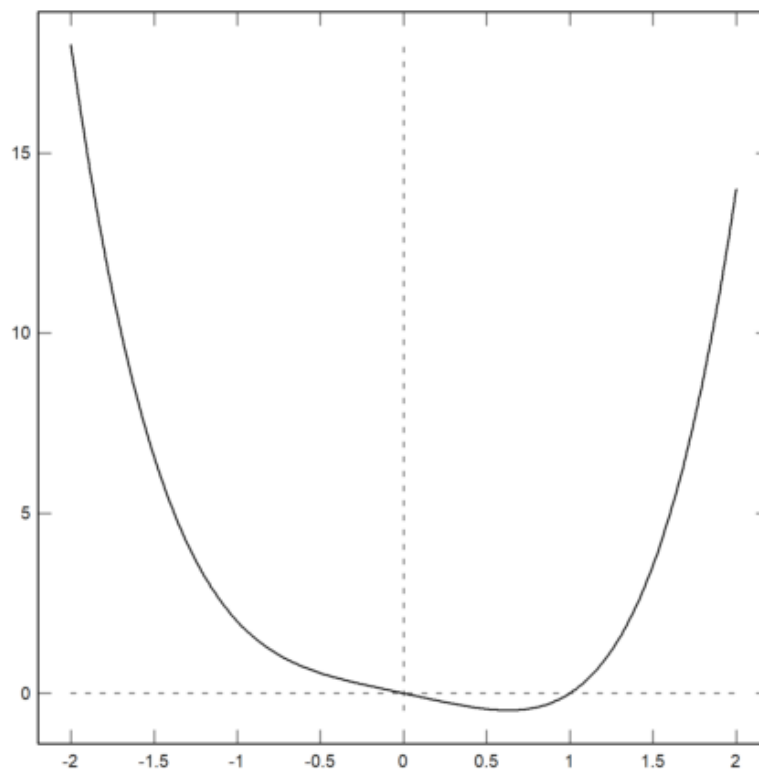
Ini penting untuk menahan grafik karena perintah `plot()` akan menghapus jendela plot.

Untuk menghapus semuanya kita menggunakan `reset()`.

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

Sebagai contoh lain, kita menggambar sebuah plot sebagai inset (sisipan) di dalam plot lain. Ini dilakukan dengan mendefinisikan jendela plot yang lebih kecil. Perlu dicatat bahwa jendela ini tidak menyediakan ruang untuk label sumbu di luar jendela plot, sehingga kita perlu menambahkan margin sesuai kebutuhan. Kita menyimpan dan mengembalikan jendela penuh, serta menahan plot saat kita menggambar inset.

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



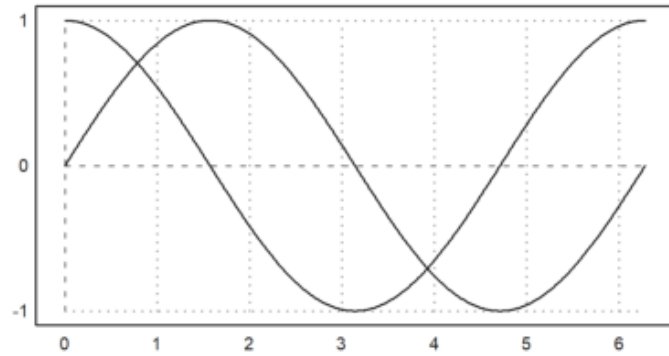
```
>hold off;  
>window(ow);
```

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

Aspek Plot

Plot default menggunakan jendela plot persegi. Kita dapat mengubah ini dengan fungsi `aspect()`. Jangan lupa untuk mengatur ulang aspek setelahnya. Kita juga dapat mengubah pengaturan ini dalam menu dengan "Set Aspect" ke rasio aspek tertentu atau ke ukuran grafik saat ini. Anda juga dapat mengubahnya untuk satu plot. Untuk ini, ukuran area plot saat ini diubah, dan jendela diatur agar label memiliki ruang yang cukup.

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



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

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

Plot 2D dalam Euler

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

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

Euler dapat memplot plot 2D dari:

- Ekspresi,
- Fungsi, variabel, atau kurva parameterisasi,
- Vektor nilai x-y,
- Sekumpulan titik di bidang,
- Kurva implisit dengan level atau area level,
- Fungsi kompleks,

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

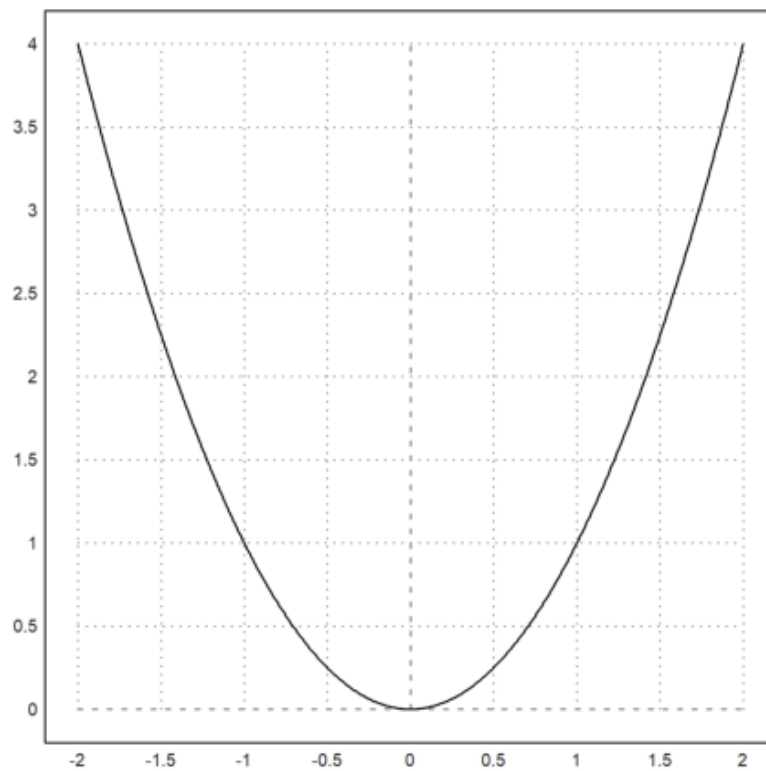
Plot Ekspresi atau Variabel

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

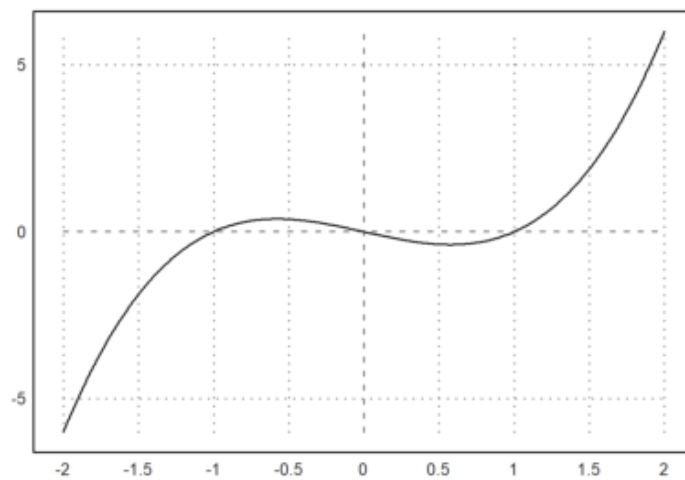
Berikut adalah contoh paling dasar, yang menggunakan rentang default dan menetapkan rentang y yang sesuai untuk memuat plot fungsi tersebut.

atatan: 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.

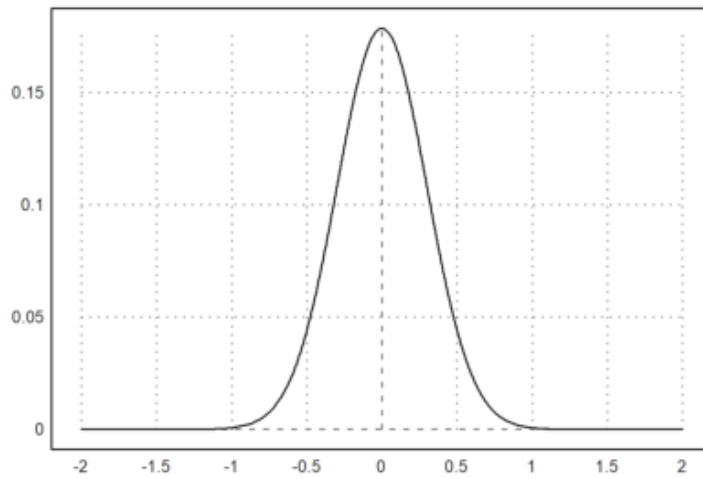
```
>plot2d("x^2") :
```



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



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

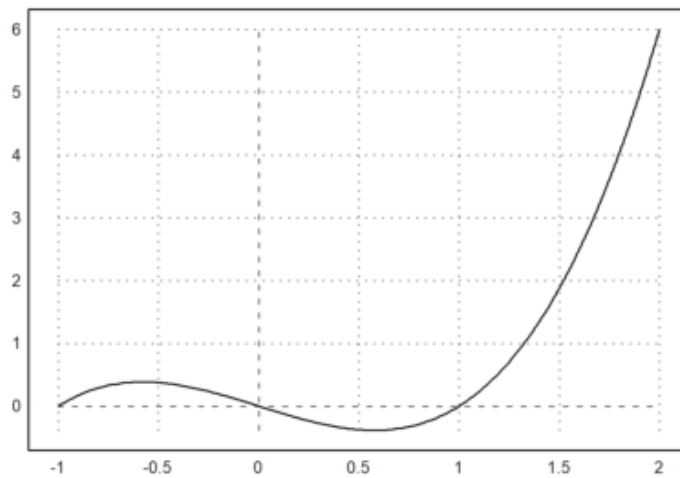


Dari beberapa contoh sebelumnya Anda dapat melihat bahwa aslinya gambar plot menggunakan sumbu X dengan rentang nilai dari -2 sampai dengan 2. Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan nilai-nilai batas X (dan Y) di belakang ekspresi yang digambar.

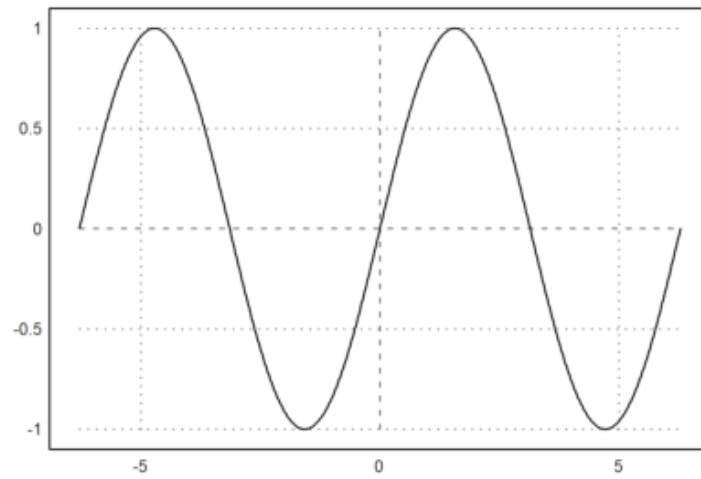
Rentang plot diatur dengan parameter yang ditetapkan berikut:

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

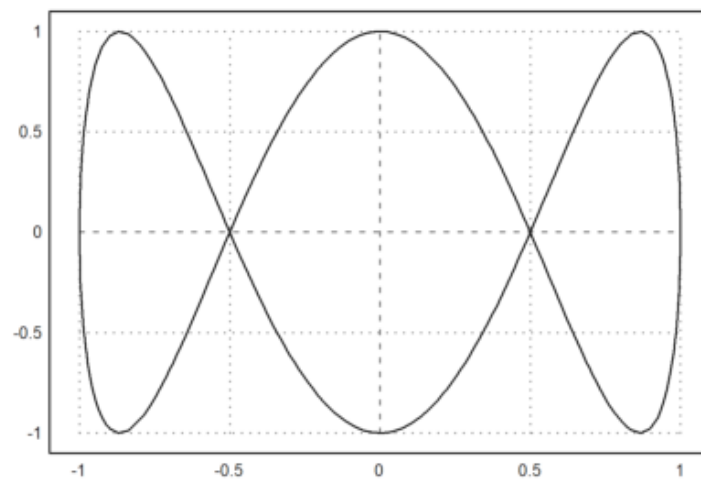
```
>plot2d("x^3-x",-1,2):
```



```
>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(lines), yang menyisipkan plot sesuai jumlah baris teks yang ditentukan.

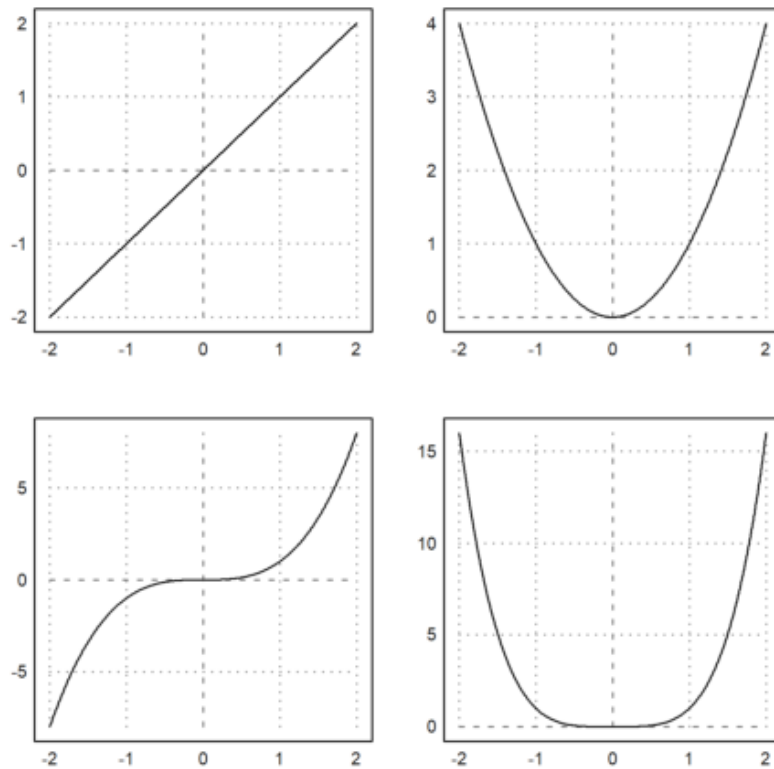
Dalam opsi lainnya, plot dapat diatur agar muncul:

- di jendela yang dapat diubah ukurannya
- di jendela notebook

Lebih banyak gaya dapat dicapai dengan perintah plot khusus.

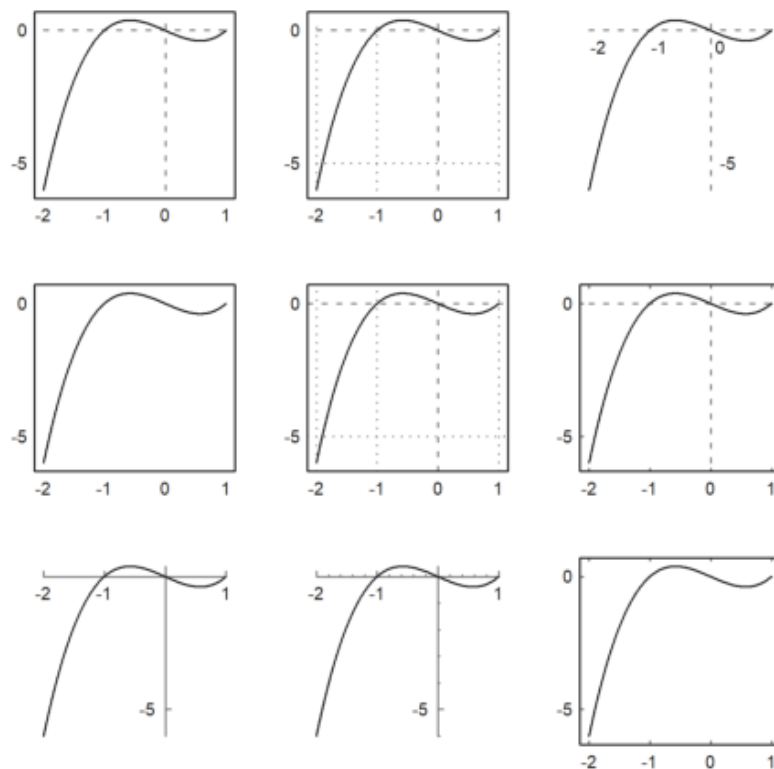
Untuk membagi jendela menjadi beberapa plot, gunakan perintah figure(). Dalam contoh berikut, kami memplotkan x^1 hingga x^4 ke dalam 4 bagian jendela. figure(0) mengatur ulang jendela ke default.

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



Dalam `plot2d()`, ada gaya alternatif yang tersedia dengan `grid=x`. Untuk gambaran, kami menunjukkan berbagai gaya grid dalam satu figur.

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

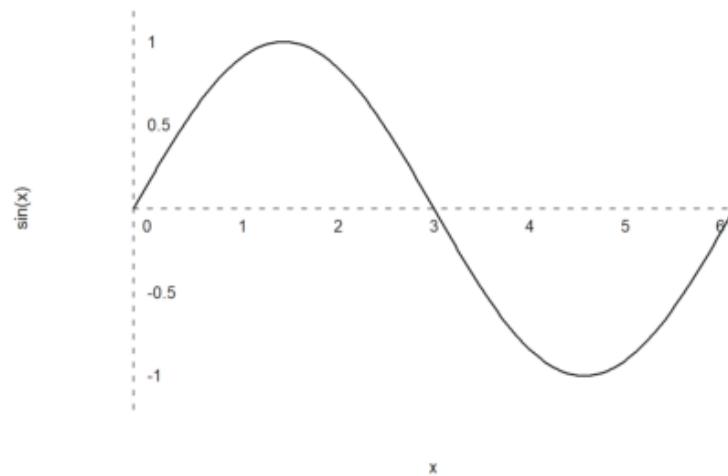


Jika argumen untuk `plot2d()` adalah ekspresi yang diikuti oleh empat angka, angka-angka tersebut adalah rentang `x` dan `y` untuk plot.

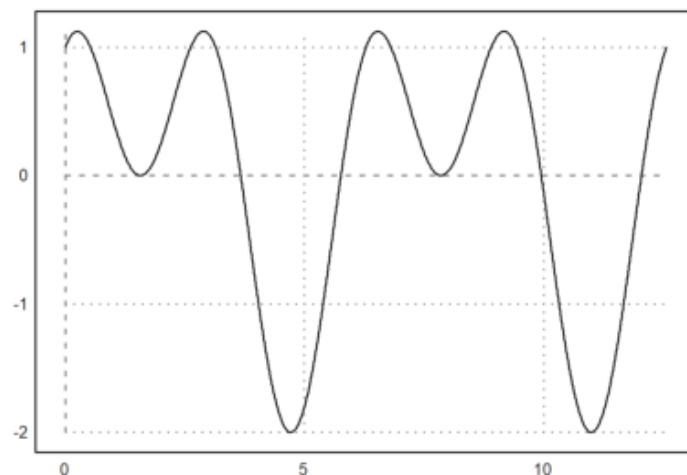
Sebagai alternatif, `a`, `b`, `c`, `d` dapat ditentukan sebagai parameter yang ditetapkan, misalnya `a=...` dll.

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

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



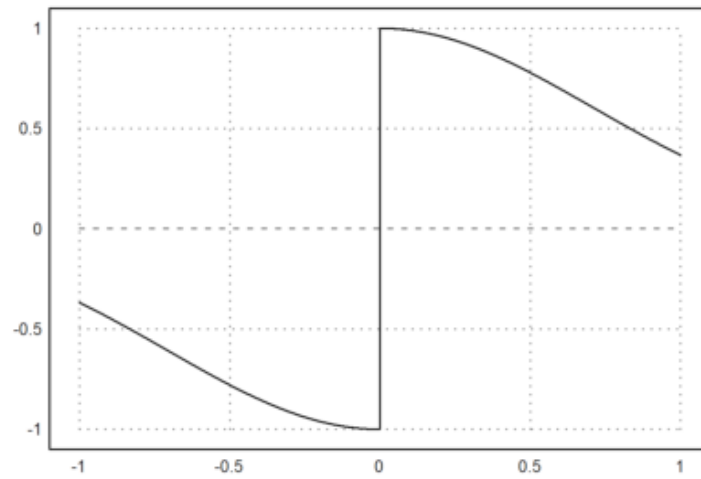
```
>plot2d("sin(x)+cos(2*x)",0,4pi):
```



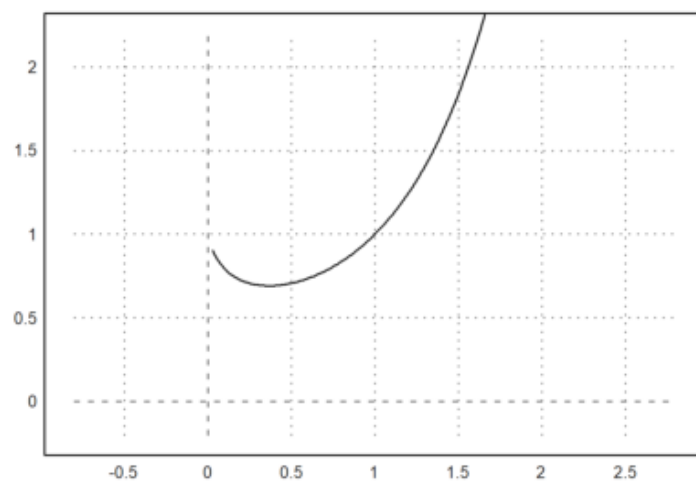
Gambar yang dihasilkan dari menyisipkan plot ke dalam jendela teks disimpan di direktori yang sama dengan notebook, secara default di subdirektori bernama "images". Gambar-gambar ini juga digunakan dalam ekspor HTML. Anda dapat menandai gambar apa pun dan menyalinnya ke clipboard dengan Ctrl-C. Tentu saja, Anda juga dapat mengekspor grafik saat ini dengan fungsi di menu File.

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

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

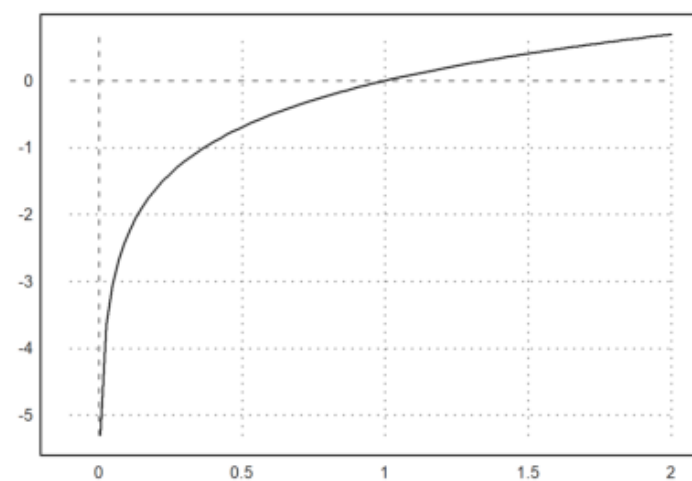



```
>plot2d("x^x",r=1.2,cx=1,cy=1):
```



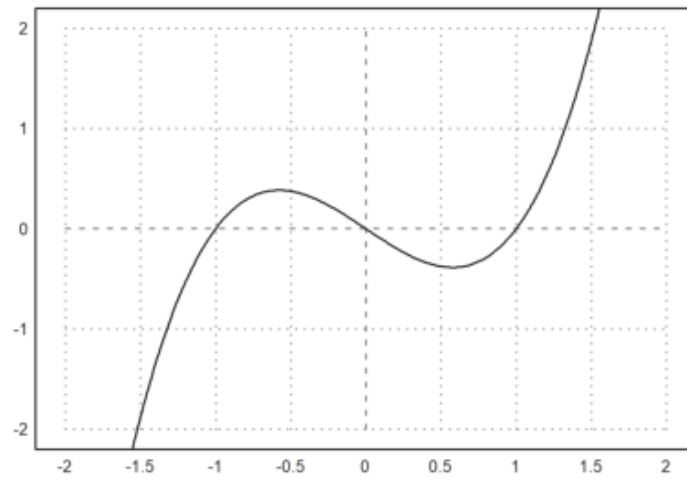
Perlu dicatat bahwa x^x tidak didefinisikan untuk $x \leq 0$. Fungsi `plot2d` menangkap kesalahan ini, dan mulai memplot begitu fungsi terdefinisi. Ini berlaku untuk semua fungsi yang mengembalikan `NAN` di luar rentang definisinya.

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

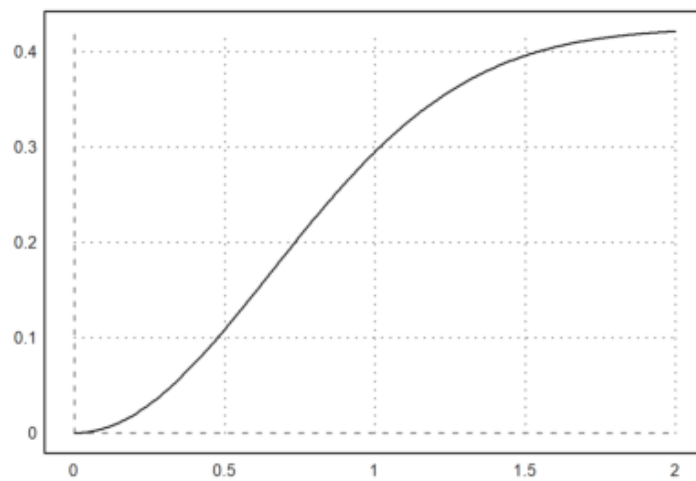


Parameter `square=true` (atau `>square`) memilih rentang y secara otomatis sehingga hasilnya adalah jendela plot persegi. Secara default, Euler menggunakan ruang persegi di dalam jendela plot.

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



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

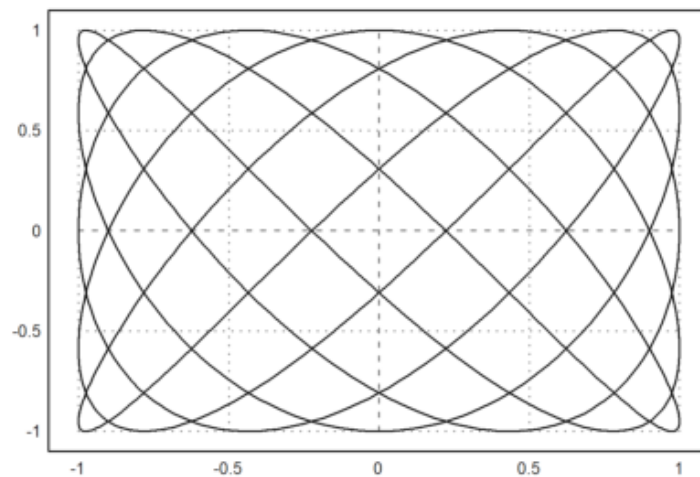


Jika Anda memerlukan lebih banyak ruang untuk label sumbu y, panggil `shrinkwindow()` dengan parameter yang lebih kecil, atau atur nilai positif untuk "smaller" di `plot2d()`.

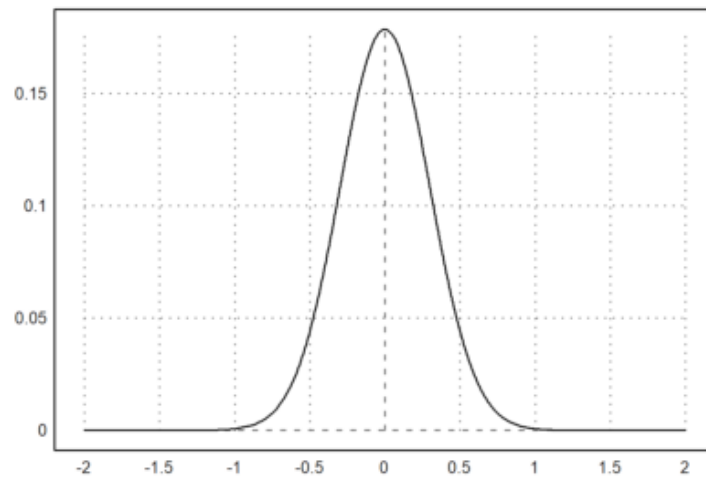
```
>plot2d("gamma(x)",1,10,yl="y-values",smaller=6,<vertical):
```

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

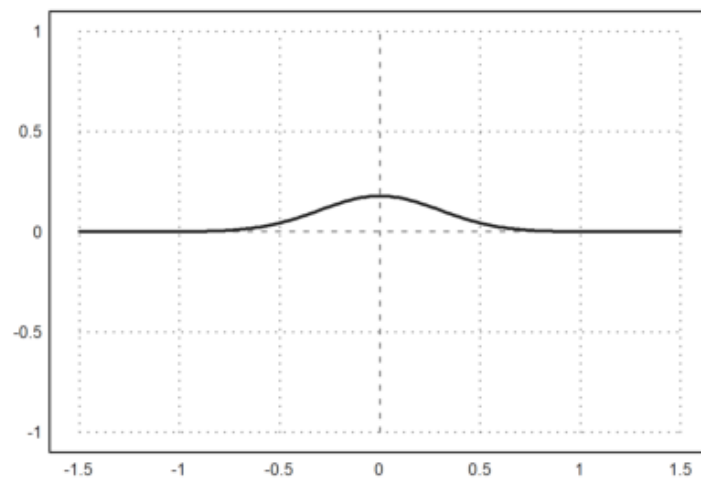
```
>x=linspace(0,2pi,1000); plot2d(sin(5x),cos(7x)):
```



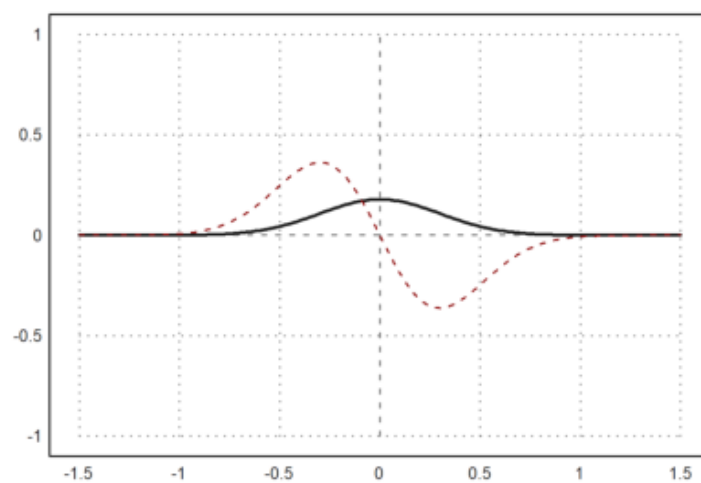
```
>a:=5.6; expr &= exp(-a*x^2)/a; // define expression
>plot2d(expr,-2,2): // plot from -2 to 2
```



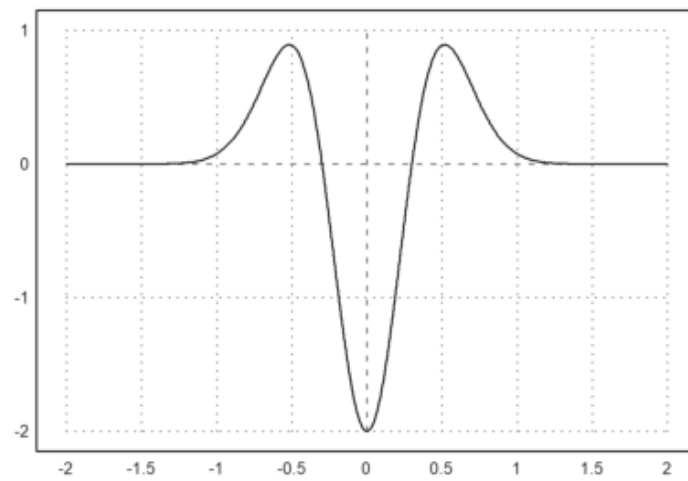
```
>plot2d(expr,r=1,thickness=2): // plot in a square around (0,0)
```



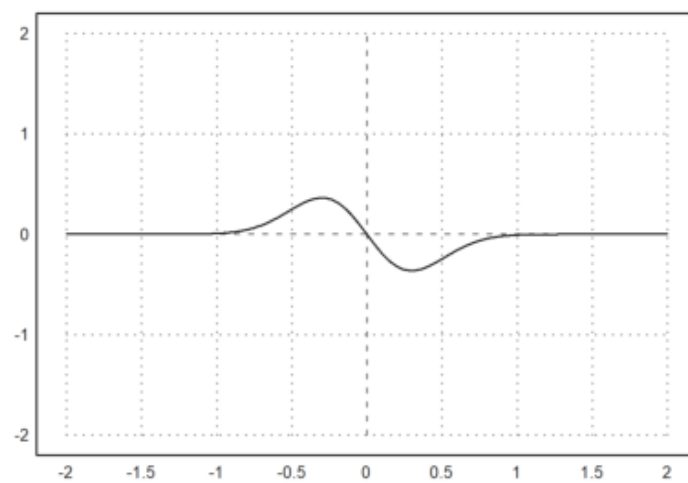
```
>plot2d(&diff(expr,x),>add,style="--",color=red): // add another plot
```



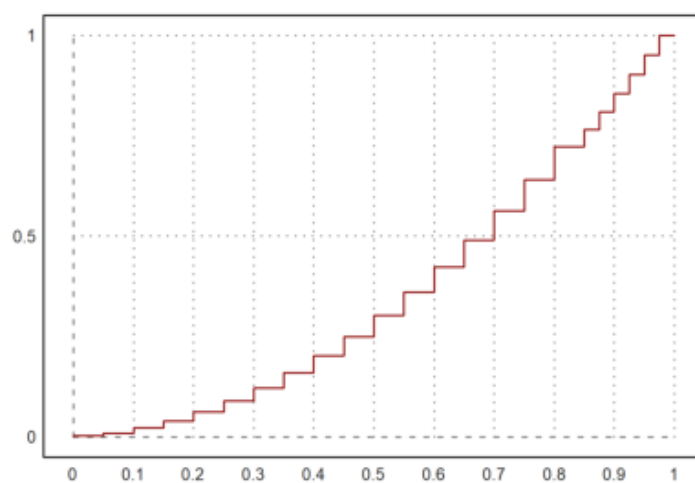
```
>plot2d(&diff(expr,x,2),a=-2,b=2,c=-2,d=1): // plot in rectangle
```



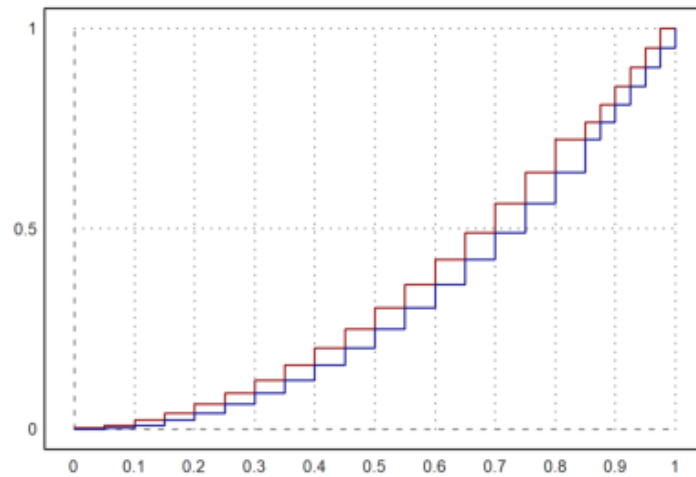
```
>plot2d(&diff(expr,x),a=-2,b=2,>square): // keep plot square
```



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



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

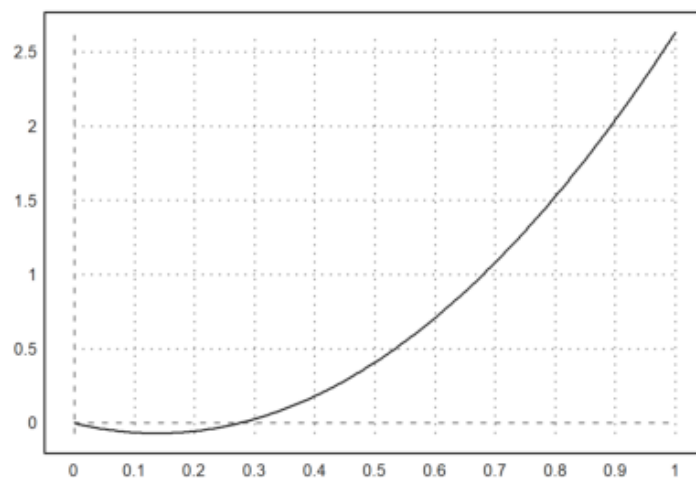


Fungsi dengan Satu Parameter

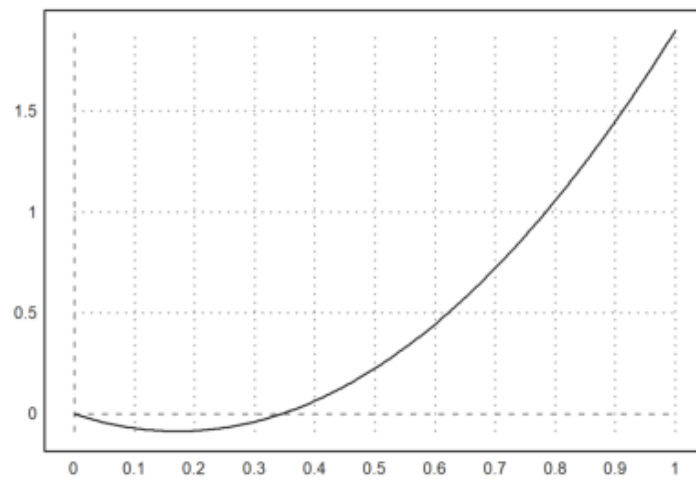
Fungsi plot paling penting untuk plot planar adalah `plot2d()`. Fungsi ini diimplementasikan dalam bahasa Euler di file "plot.e", yang dimuat saat program dimulai.

Berikut beberapa contoh menggunakan fungsi. Seperti biasa dalam EMT, fungsi yang bekerja untuk fungsi atau ekspresi lainnya, Anda dapat memberikan parameter tambahan (selain `x`) yang bukan variabel global kepada fungsi dengan parameter titik koma atau dengan koleksi panggilan.

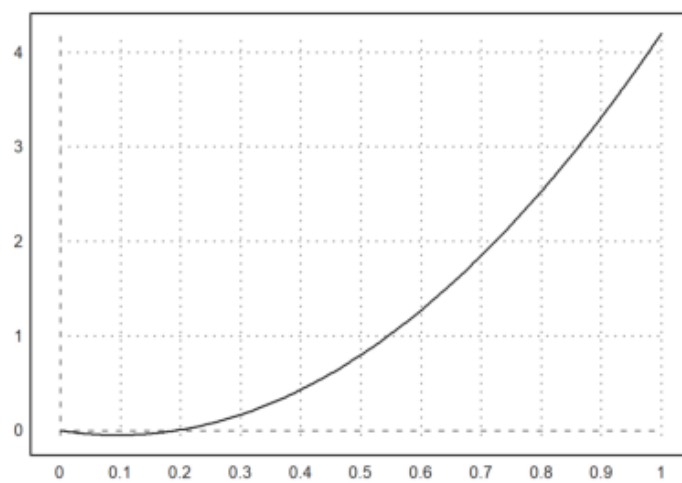
```
>function f(x,a) := x^2/a+a*x^2-x; // define a function
>a=0.3; plot2d("f",0,1;a): // plot with a=0.3
```



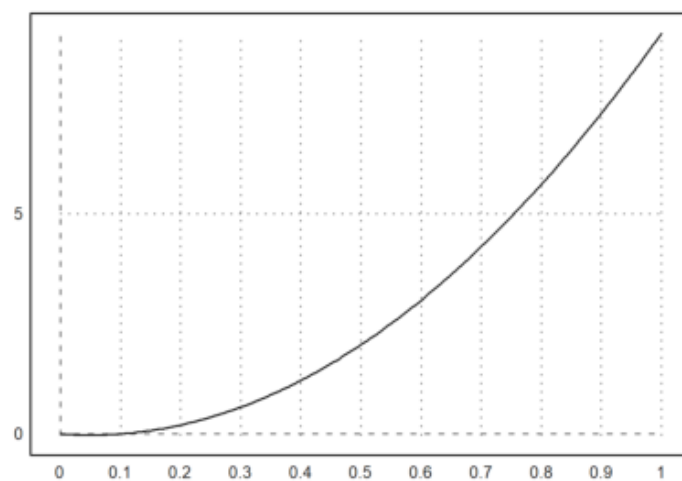
```
>plot2d("f",0,1;0.4): // plot with a=0.4
```



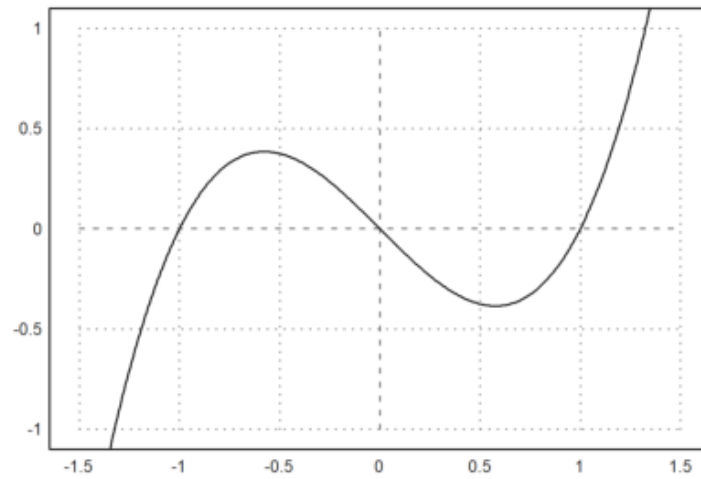
```
>plot2d({"f",0.2}),0,1): // plot with a=0.2
```



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



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



Berikut ringkasan fungsi yang diterima:

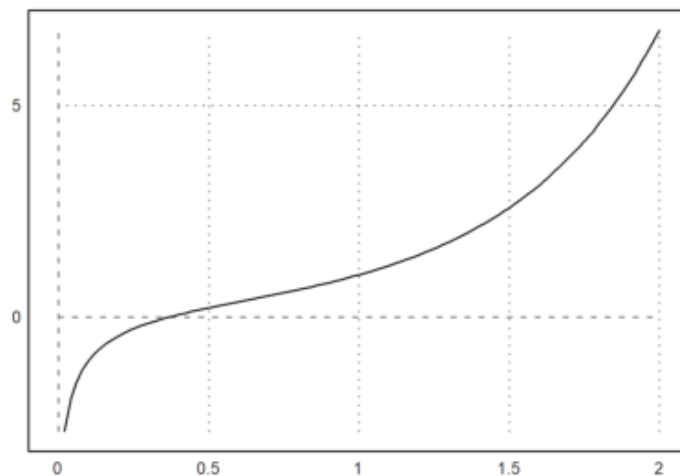
- Ekspresi atau ekspresi simbolik dalam x
- Fungsi atau fungsi simbolik berdasarkan nama seperti "f"
- Fungsi simbolik hanya dengan nama f

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

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

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

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

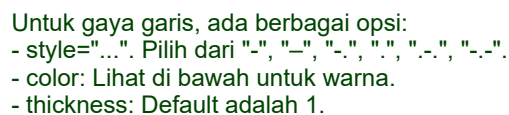
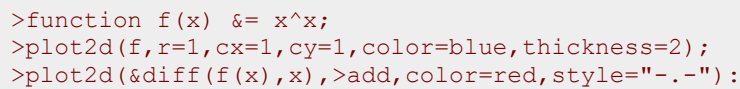


Tentu saja, untuk ekspresi atau ekspresi simbolik, nama variabel saja cukup untuk memplotkannya.

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

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

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



- 0..15: indeks warna default
- constanta warna: putih, hitam, merah, hijau, biru, cyan, olive, lightgray, gray, darkgray, orange, lightgreen, turquoise, lightblue, lightorange, kuning
- rgb(red,green,blue): parameter adalah real dalam [0,1].

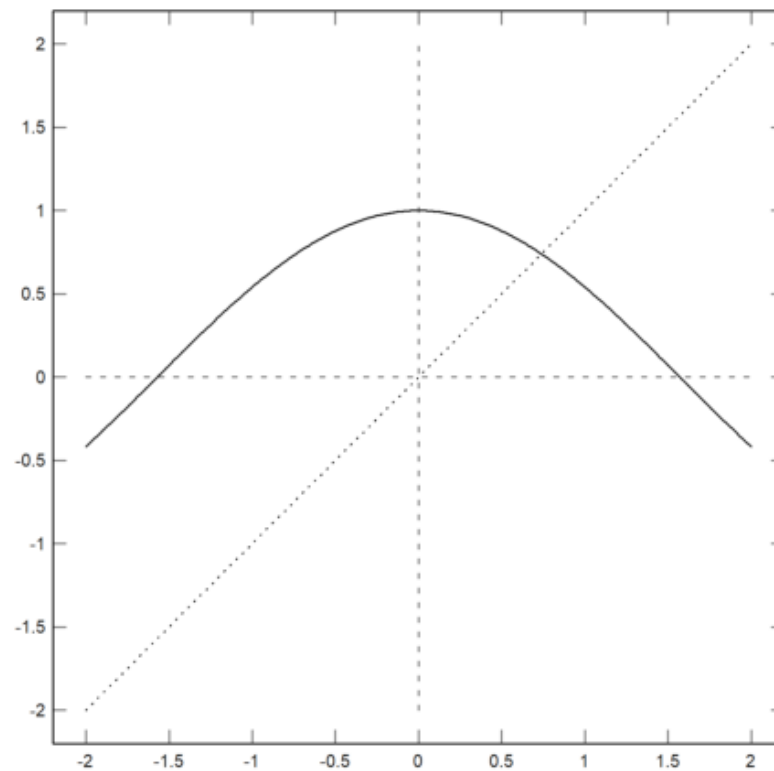
Berikut tampilan warna yang telah ditetapkan dalam EMT:

Namun, Anda dapat menggunakan warna apapun.

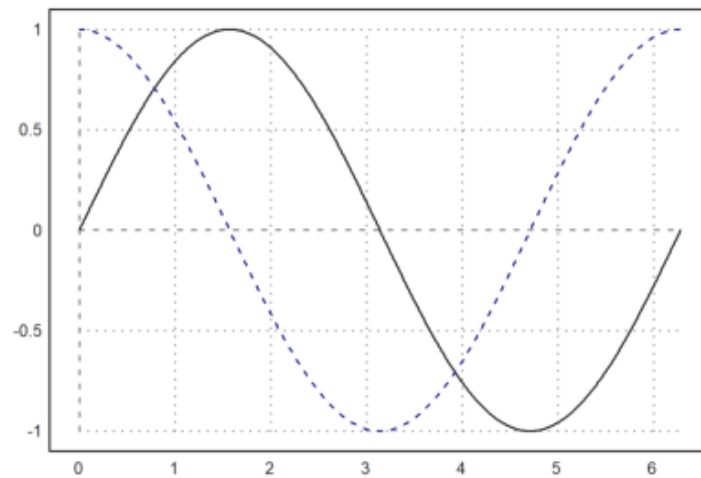
```
> columnsplot(ones(1,16),grid=0,color=rgb(0,0,linspace(0,1,15))):
```

Memplot lebih dari satu fungsi ke dalam satu jendela dapat dilakukan dengan beberapa cara. Salah satu metodenya adalah menggunakan `>add` untuk beberapa panggilan ke `plot2d` di semua, kecuali panggilan pertama. Kami telah menggunakan fitur ini dalam contoh sebelumnya.


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

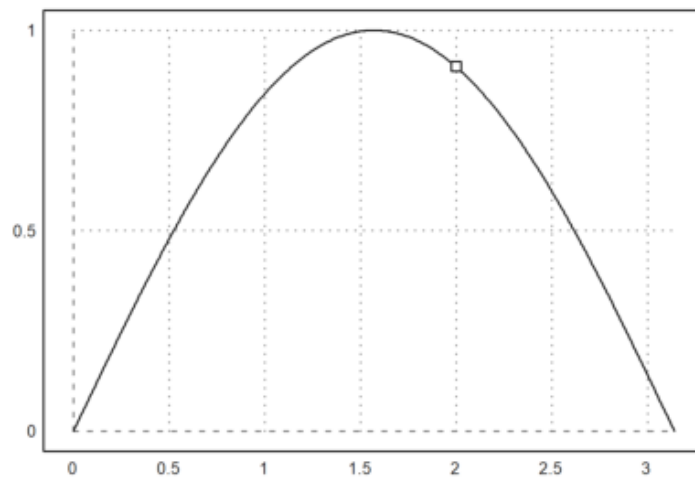


```
>aspect(1.5); plot2d("sin(x)",0,2pi); plot2d("cos(x)",color=blue,style="--",>add):
```



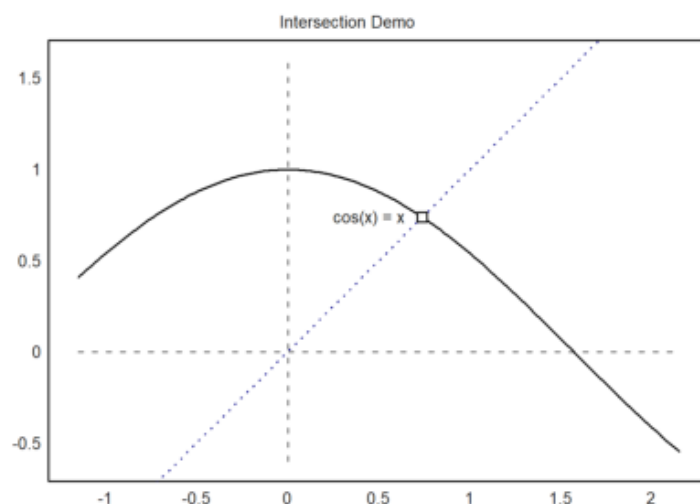
Salah satu kegunaan `>add` adalah untuk menambahkan titik pada kurva.

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



Kami menambahkan titik perpotongan dengan label (di posisi "cl" untuk tengah kiri), dan menyisipkan hasilnya ke dalam notebook. Kami juga menambahkan judul pada 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):
```



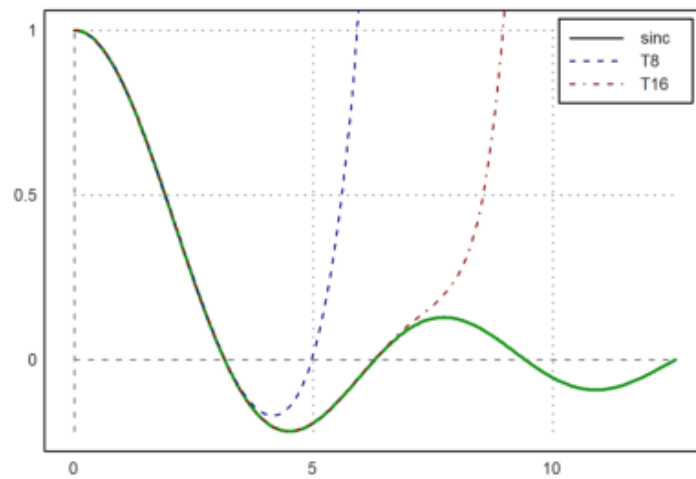
Dalam demo berikut, kami memplot fungsi sinc(x)=sin(x)/x dan ekspansi Taylor ke-8 dan ke-16. Kami menghitung ekspansi ini menggunakan Maxima melalui ekspresi simbolik. Plot ini dilakukan dalam perintah multi-baris berikut dengan tiga panggilan ke plot2d(). Panggilan kedua dan ketiga memiliki flag >add, yang membuat plot menggunakan rentang sebelumnya.

Kami menambahkan kotak label untuk menjelaskan fungsi-fungsi tersebut.

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

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

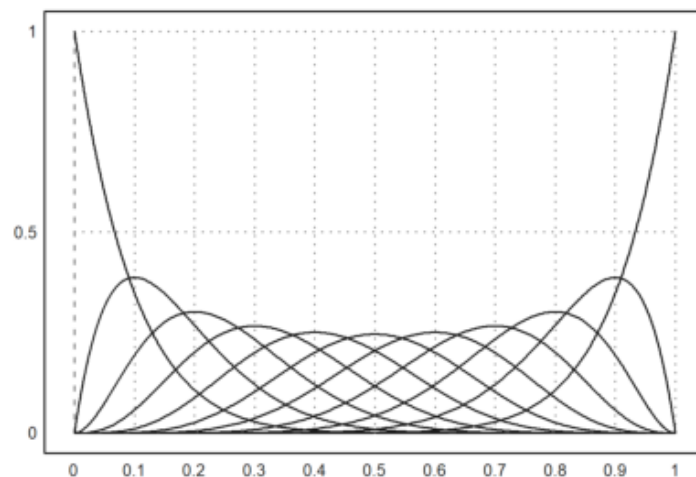
```
>plot2d("sinc(x)", 0, 4pi, color=green, thickness=2); ...
  plot2d(&taylor(sin(x)/x, x, 0, 8), >add, color=blue, style="--"); ...
  plot2d(&taylor(sin(x)/x, x, 0, 16), >add, color=red, style="-.-"); ...
  labelbox(["sinc", "T8", "T16"], styles=["-", "--", "-.-"], ...
    colors=[black, blue, red]):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



Dalam contoh berikut, kami menghasilkan polinomial Bernstein.

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

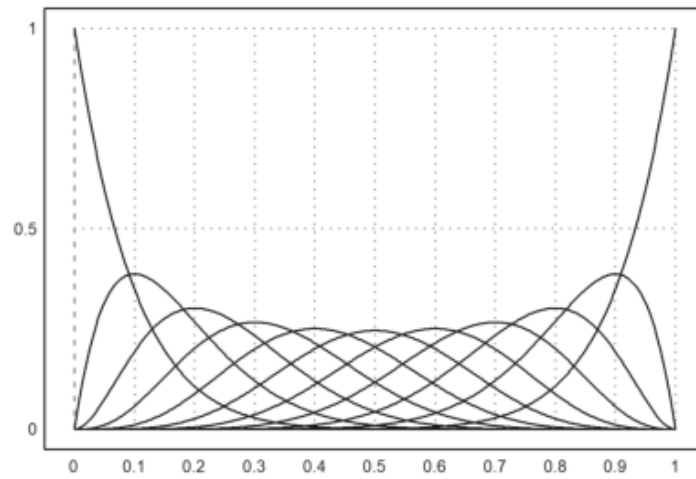
```
>plot2d("(1-x)^10",0,1); // plot first function
>for i=1 to 10; plot2d("bin(10,i)*x^i*(1-x)^(10-i)",>add); end;
>insimg;
```



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

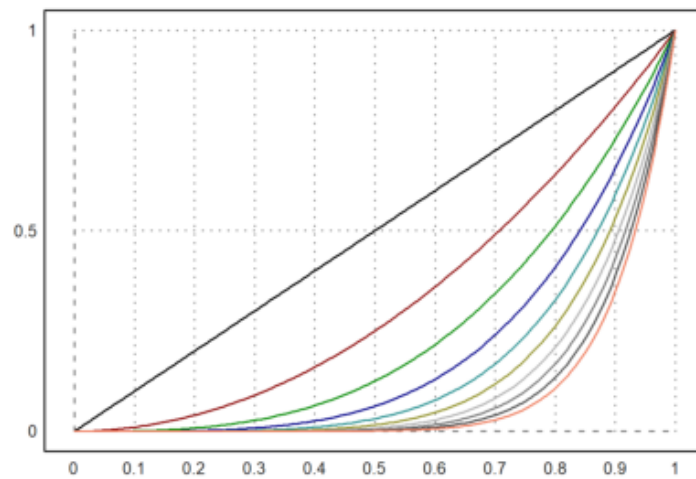
Kami menghasilkan matriks nilai dengan satu Polinomial Bernstein di setiap baris. Untuk ini, kami cukup menggunakan vektor kolom dari i. Lihat pengantar tentang bahasa matriks untuk mempelajari detail lebih lanjut.

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



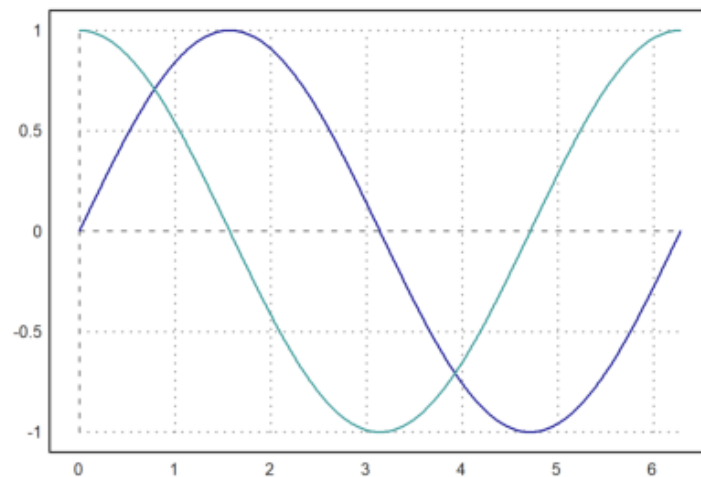
Perlu diperhatikan bahwa parameter color dapat berupa vektor. Kemudian setiap warna digunakan untuk setiap baris matriks.

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

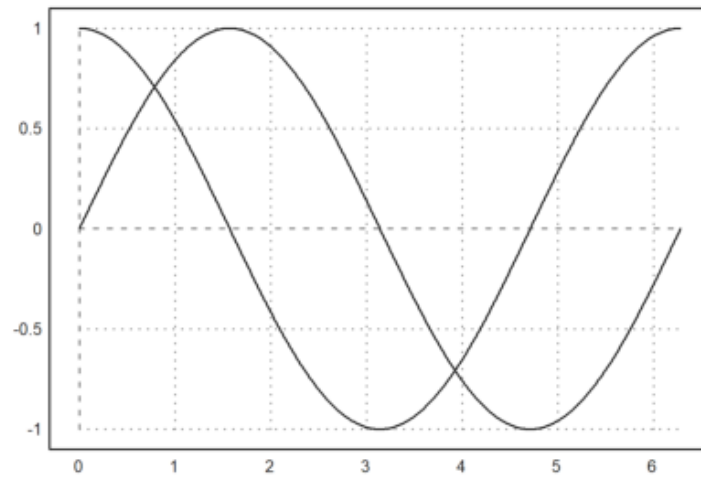


Metode lain adalah menggunakan vektor ekspresi (string). Anda kemudian dapat menggunakan array warna, array gaya, dan array ketebalan dengan panjang yang sama.

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



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



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

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

```

      10      9      8 2      7 3
[(1 - x) , 10 (1 - x) x, 45 (1 - x) x , 120 (1 - x) x ,
 6 4      5 5      4 6      3 7
210 (1 - x) x , 252 (1 - x) x , 210 (1 - x) x , 120 (1 - x) x ,
 2 8      9 10
45 (1 - x) x , 10 (1 - x) x , x ]

```

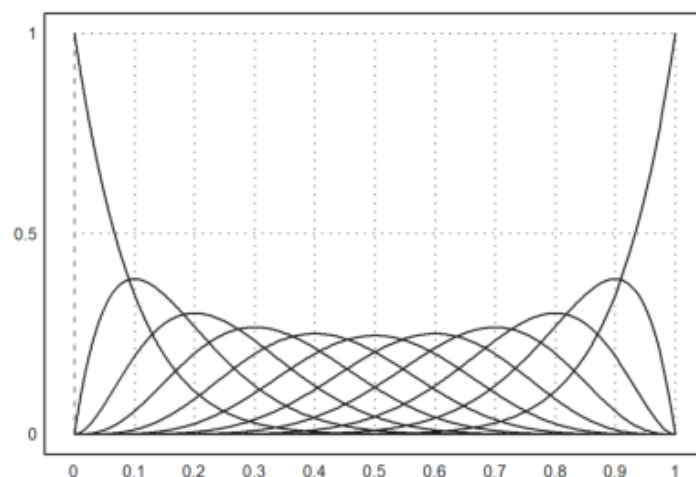
```
>mxm2str(v) // get a vector of strings from the symbolic vector
```

```

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

```

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



Alternatif lain adalah menggunakan bahasa matriks dari Euler.

Jika sebuah 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 ditambahkan array warna, ini akan digunakan untuk setiap baris plot.

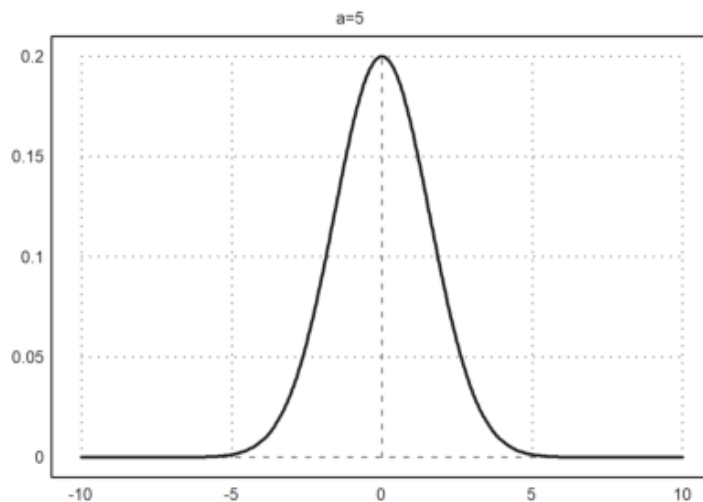
```
>n=(1:10)'; plot2d("x^n",0,1,color=1:10):
```

Ekspresi dan fungsi satu baris dapat melihat variabel global.

Jika Anda tidak bisa menggunakan variabel global, Anda perlu menggunakan fungsi dengan parameter tambahan, dan meneruskan parameter ini sebagai parameter titik koma. Pastikan semua parameter yang ditetapkan berada di akhir perintah plot2d.

Dalam contoh ini, kami meneruskan $a=5$ ke fungsi f , yang kami plot dari -10 hingga 10.

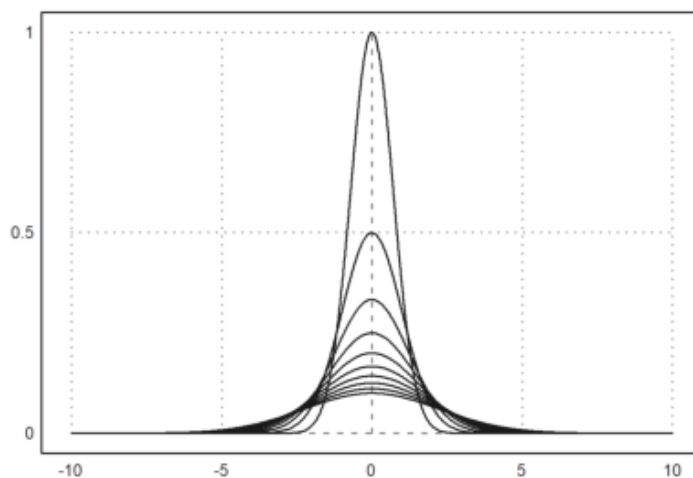
```
>function f(x,a) := 1/a*exp(-x^2/a); ...  
plot2d("f",-10,10;5,thickness=2,title="a=5"):
```



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

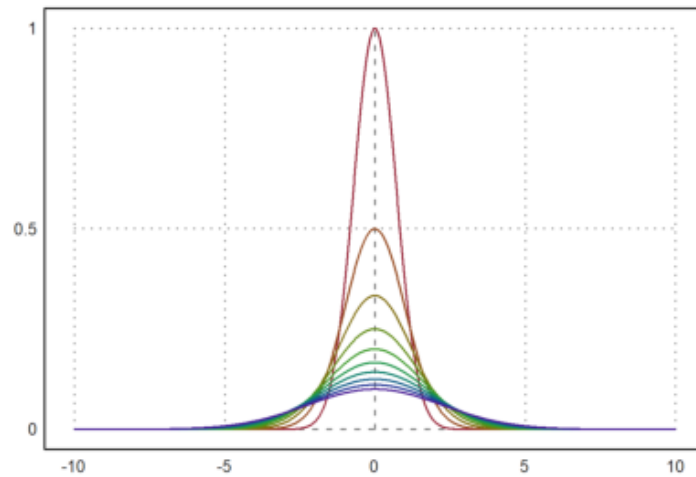
Dalam contoh berikut, kami menggunakan loop untuk memplot beberapa fungsi.

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



Hasil yang sama dapat dicapai dengan 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 penjelasan.

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



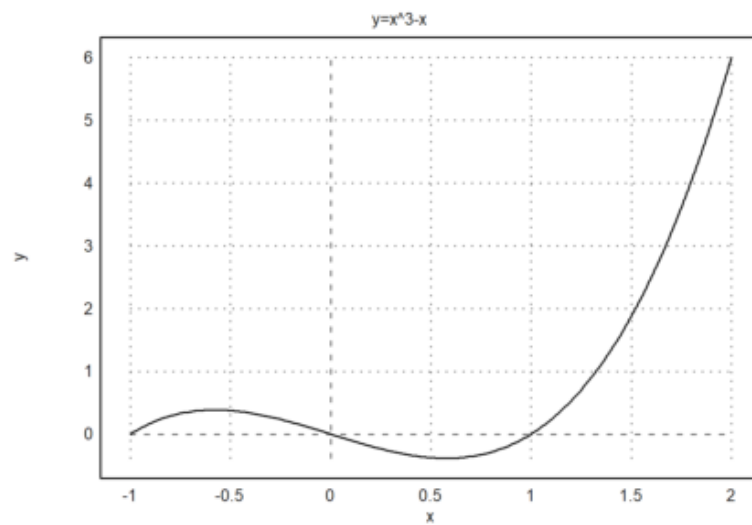
Label Teks

Dekorasi sederhana dapat berupa:

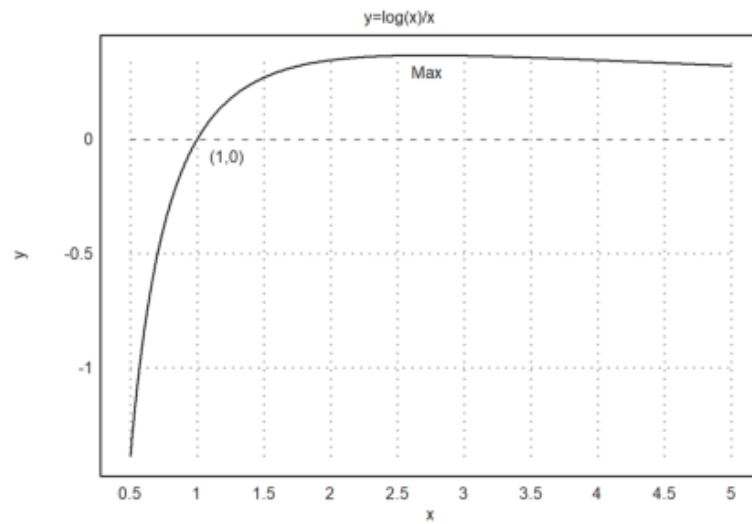
- judul dengan title="..."
- label x dan y dengan xl="...", yl="..."
- label teks lain dengan label("...",x,y)

Perintah label akan memplot ke plot saat ini di koordinat plot (x,y). Ia dapat mengambil argumen posisi.

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

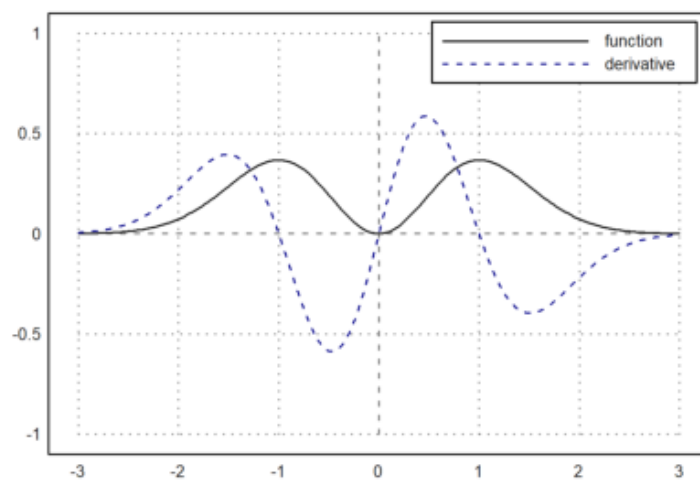


```
>expr := "log(x)/x"; ...
plot2d(expr,0.5,5,title="y="+expr,xl="x",yl="y"); ...
label("(1,0)",1,0); label("Max",E,expr(E),pos="lc"):
```



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

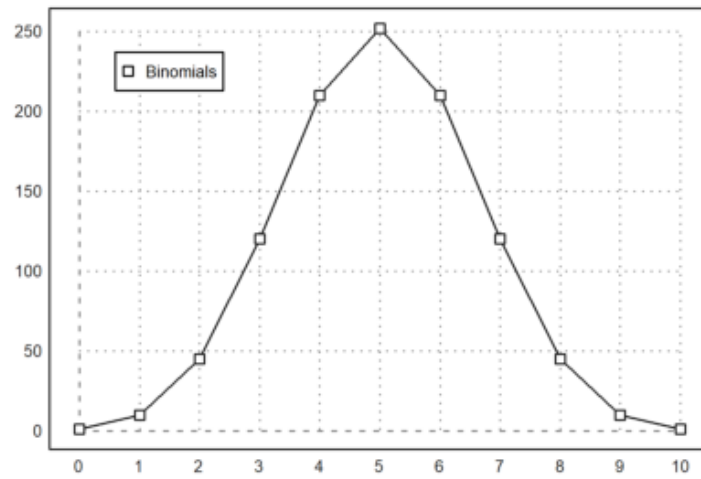
```
>function f(x) &= x^2*exp(-x^2); ...
plot2d(&f(x),a=-3,b=3,c=-1,d=1); ...
plot2d(&diff(f(x),x),>add,color=blue,style="--"); ...
labelbox(["function","derivative"],styles=["-","--"], ...
         colors=[black,blue],w=0.4):
```



Kotak ini diposisikan di sudut kanan atas secara default, tetapi `>left` akan memposisikannya di sudut kiri atas. Anda dapat memindahkannya ke tempat mana saja yang Anda inginkan. Posisi jangkar adalah sudut kanan atas kotak, dan angka-angka adalah pecahan dari ukuran jendela grafik. Lebarinya otomatis.

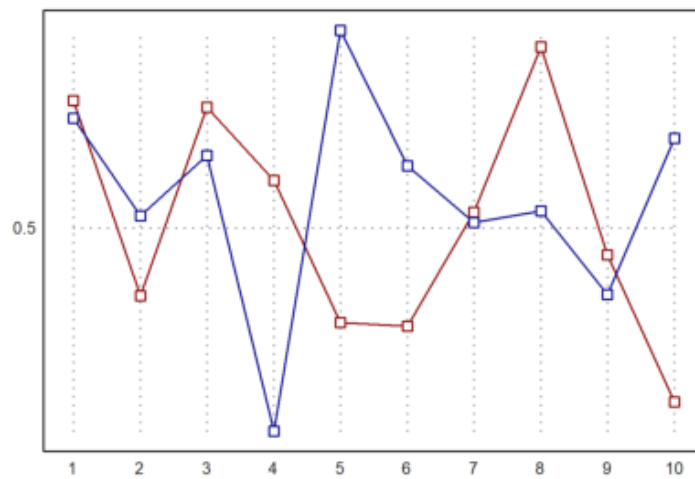
Untuk plot titik, kotak label juga berfungsi. Tambahkan parameter `>points`, atau vektor flag, satu untuk setiap label. Dalam contoh berikut, hanya ada satu fungsi. Jadi kita bisa menggunakan string alih-alih vektor string. Kami menetapkan warna teks ke hitam untuk contoh ini.

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

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

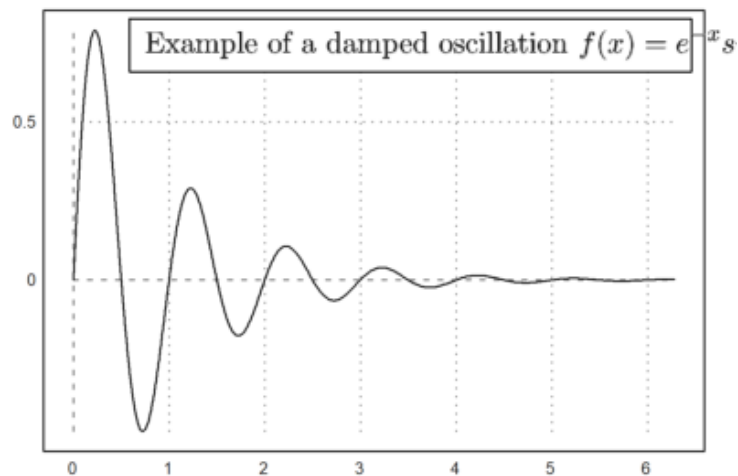
```
>statplot(1:10,random(2,10),color=[red,blue]):
```



Fitur serupa adalah fungsi `textbox()`.

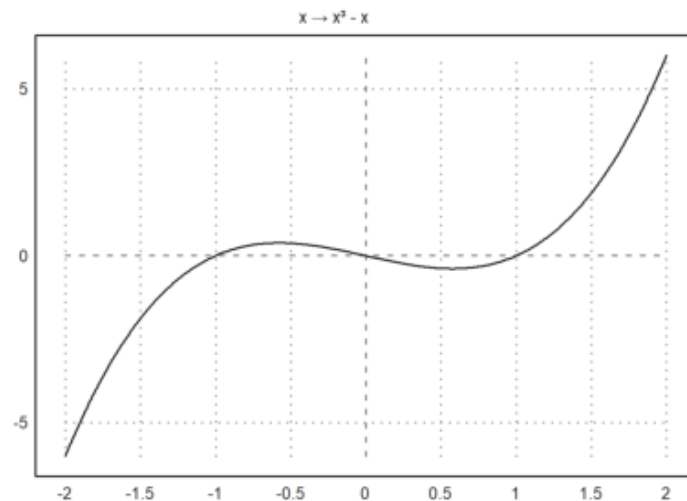
Lebarinya secara default adalah lebar maksimal dari garis teks, tetapi bisa diatur oleh pengguna juga.

```
>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):
```



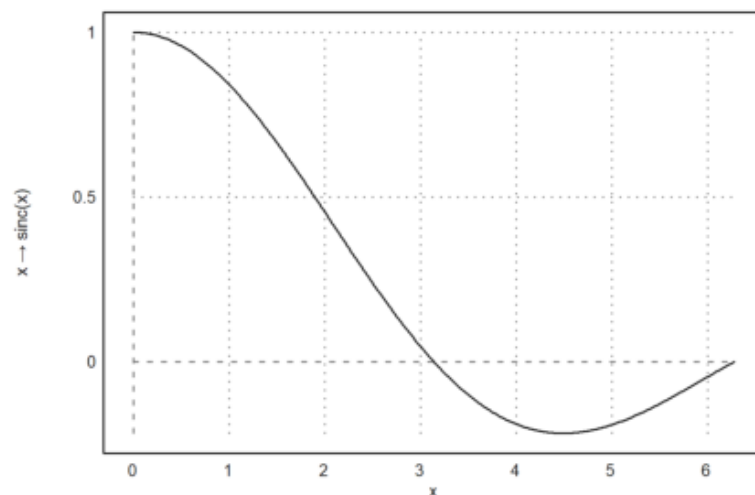
Label teks, judul, kotak label, dan teks lainnya dapat berisi string Unicode (lihat sintaks EMT untuk lebih lanjut tentang string Unicode).

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



Label pada sumbu x dan y bisa vertikal, begitu juga dengan sumbu itu sendiri.

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



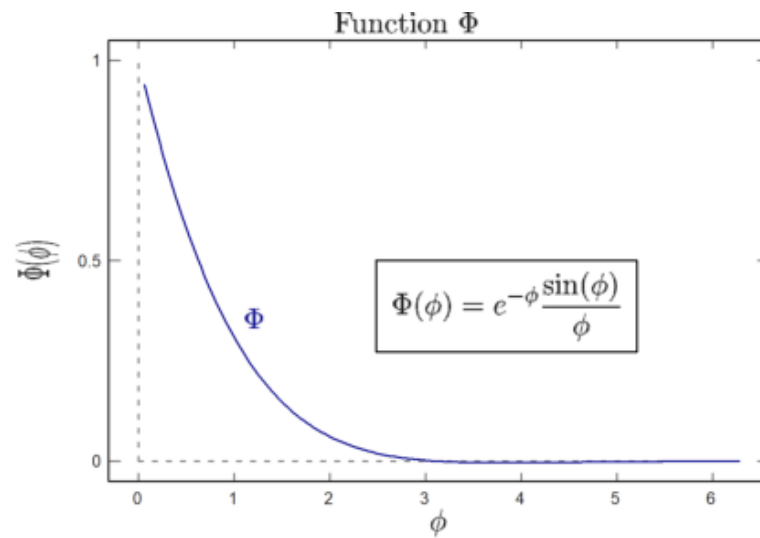
LaTeX

Anda juga dapat memplotkan formula LaTeX jika Anda sudah menginstal sistem LaTeX. Saya merekomendasikan MiKTeX. Path ke biner "latex" dan "dvi2png" harus ada dalam path sistem, atau Anda harus menyiapkan LaTeX di menu opsi.

Perlu dicatat, parsing LaTeX lambat. Jika Anda ingin menggunakan LaTeX dalam plot animasi, sebaiknya panggil `latex()` sebelum loop sekali dan gunakan hasilnya (sebuah gambar dalam matriks RGB).

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

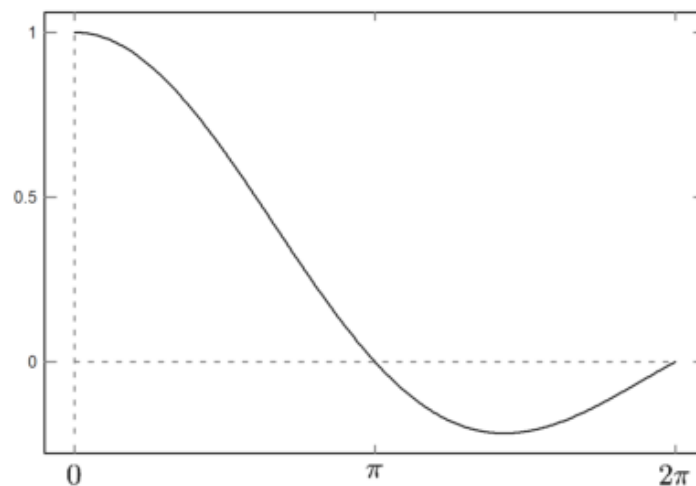
```
>plot2d("exp(-x)*sin(x)/x",a=0,b=2pi,c=0,d=1,grid=6,color=blue, ...
  title=latex("\text{Function } \Phi"), ...
  xl=latex("\phi"),yl=latex("\Phi(\phi)")); ...
textbox( ...
  latex("\Phi(\phi) = e^{-\phi} \frac{\sin(\phi)}{\phi}"),x=0.8,y=0.5); ...
label(latex("\Phi",color=blue),1,0.4):
```



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

Cara termudah adalah membuat plot kosong dengan bingkai menggunakan `grid=4`, lalu tambahkan 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 bisa digunakan.

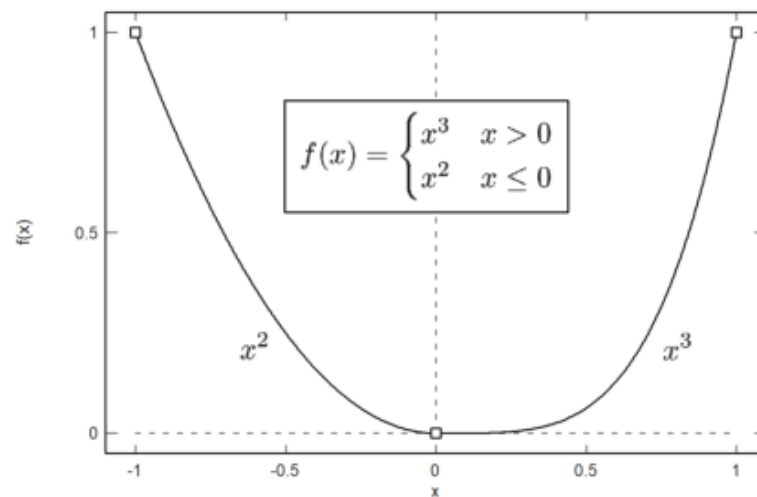
```
>function map f(x) ...
  if x>0 then return x^4
  else return x^2
endif
endfunction
```

Parameter "map" membantu menggunakan fungsi untuk vektor. Untuk plot, ini sebenarnya tidak perlu. Namun, untuk menunjukkan bahwa vektorisasi berguna, kita tambahkan beberapa titik kunci pada plot di $x=-1$, $x=0$, dan $x=1$.

Dalam plot berikut, kita juga memasukkan beberapa kode LaTeX. Kita menggunakannya untuk dua label dan sebuah kotak teks. Tentu saja, Anda hanya dapat menggunakan LaTeX jika sudah menginstalnya dengan benar.

```
>plot2d("f", -1, 1, xl="x", yl="f(x)", grid=6); ...
  plot2d([-1, 0, 1], f([-1, 0, 1]), >points, >add); ...
  label(latex("x^3"), 0.72, f(0.72)); ...
  label(latex("x^2"), -0.52, f(-0.52), pos="ll"); ...
  textbox( ...
```

```
latex("f(x)=\begin{cases} x^3 & x>0 \\ x^2 & x \leq 0 \end{cases}"), ...
x=0.7,y=0.2):
```



Interaksi Pengguna

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

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

Tombol spasi akan mengatur ulang plot ke jendela plot asli. Saat memplot data, flag `>user` 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 adalah contoh cara interaksi pengguna yang lebih maju (lihat tutorial tentang pemrograman untuk detail).

Fungsi built-in `mousedrag()` menunggu peristiwa mouse atau keyboard. Fungsi `dragpoints()` memanfaatkan ini dan memungkinkan pengguna untuk menyeret titik mana pun di dalam plot. Pertama, kita membutuhkan fungsi plot.

Sebagai contoh, kita menginterpolasi 5 titik dengan polinomial. Fungsi ini harus memplot ke area plot yang tetap.

```
>function plotf(xp,yp,select) ...
  d=interp(xp,yp);
  plot2d("interpval(xp,d,x)";d,xp,r=2);
  plot2d(xp,yp,>points,>add);
  if select>0 then
    plot2d(xp[select],yp[select],color=red,>points,>add);
  endif;
  title("Drag one point, or press space or return!");
endfunction
```

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

Sekarang kita menghasilkan beberapa nilai acak dan membiarkan pengguna menyeret titik-titik tersebut.

```
>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-parameter ini.

Pertama, kita memerlukan fungsi plot.

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

Kemudian, kita membutuhkan nama-nama untuk parameter, nilai awal, dan matriks nx2 dari rentang, opsional dengan garis heading. Ada slider interaktif, yang bisa mengatur nilai-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-nilai yang diseret menjadi bilangan bulat. Sebagai contoh, kita menulis fungsi plot yang memplot polinomial Taylor dari derajat n untuk fungsi cosinus.

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

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

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

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

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

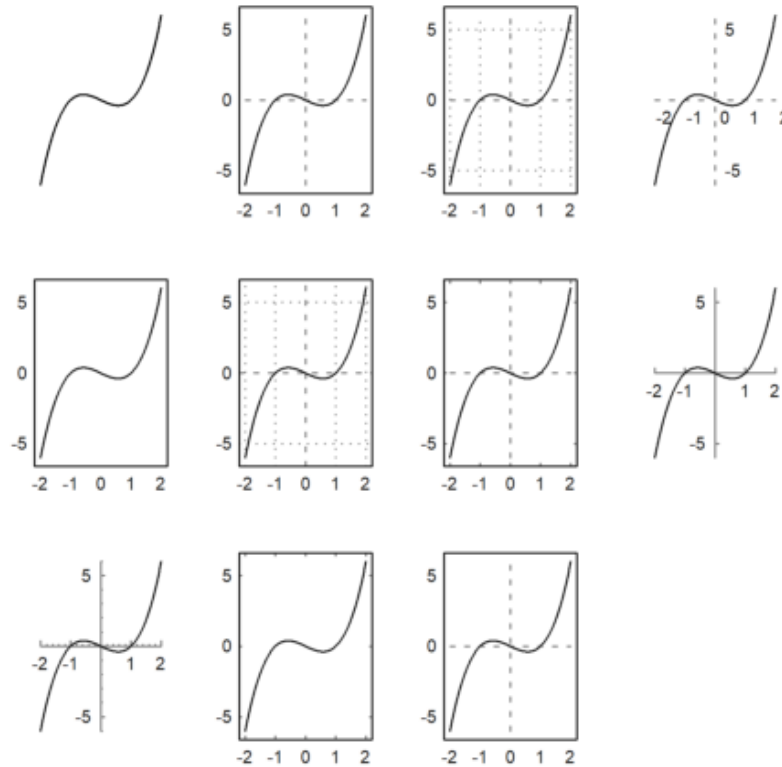
```
>dragtest // lihat hasilnya dan cobalah lakukan!
```

Gaya Plot 2D

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

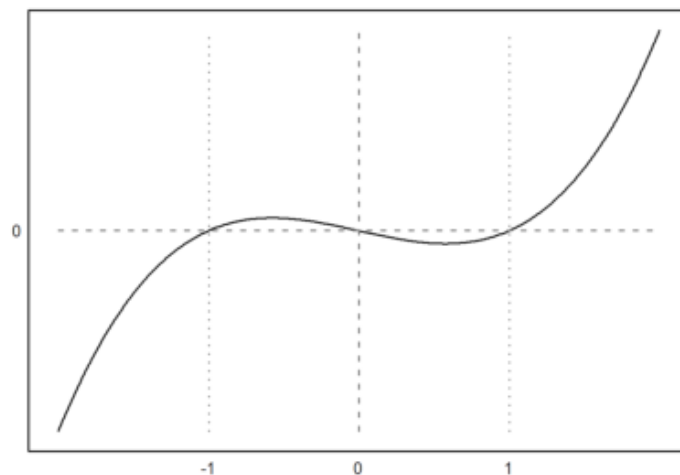
```
>aspect();  
>figure(3,4); ...  
  figure(1); plot2d("x^3-x",grid=0); ... // tanpa grid, bingkai, atau sumbu  
> figure(2); plot2d("x^3-x",grid=1); ... // sumbu x-y  
> figure(3); plot2d("x^3-x",grid=2); ... // centang default  
> figure(4); plot2d("x^3-x",grid=3); ... // sumbu x-y dengan label di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

```
> figure(11); plot2d("x^3-x",grid=10); ...// tanpa centang, hanya sumbu
> figure(0):
```



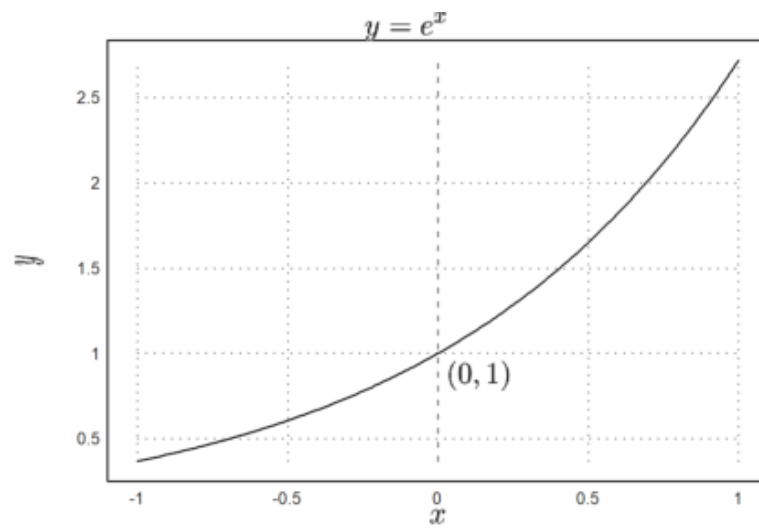
Parameter `<frame` mematikan bingkai, dan `framecolor=blue` mengatur bingkai ke warna biru. Jika Anda ingin menambahkan centang sendiri, Anda bisa menggunakan `style=0`, dan tambahkan semuanya kemudian.

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



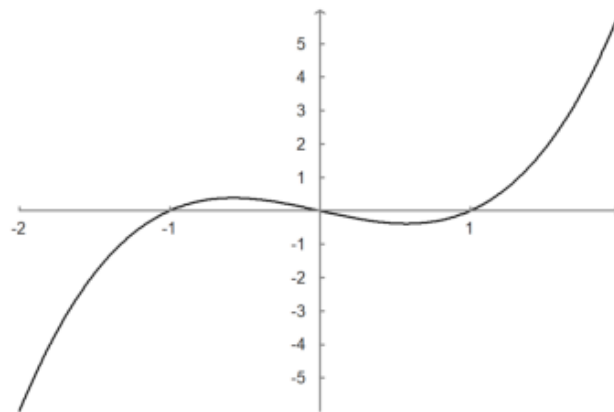
Untuk judul plot dan label sumbu, lihat contoh berikut.

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



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

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

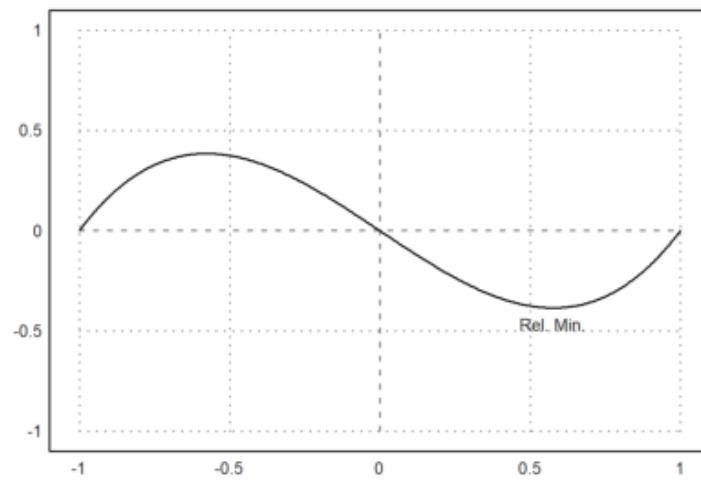


Teks pada plot dapat diatur dengan label(). Dalam contoh berikut, "lc" berarti lower center. Ini mengatur posisi label relatif terhadap koordinat plot.

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

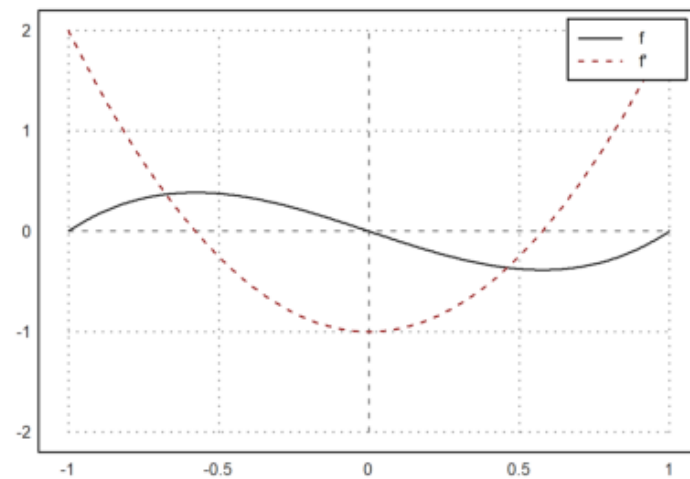
$$x^3 - x$$

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

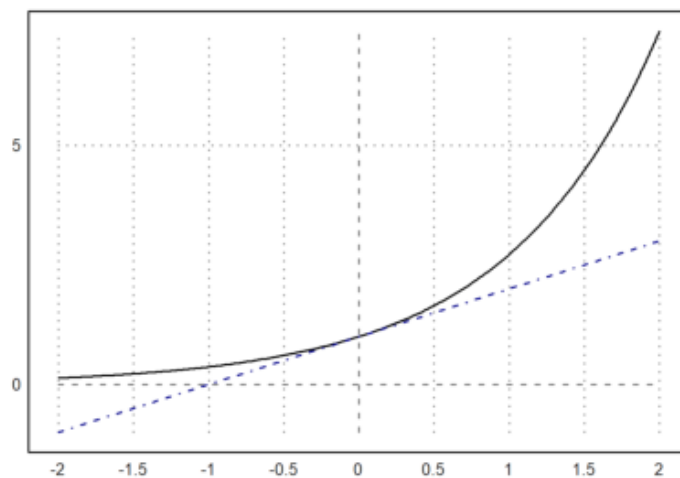


Ada juga kotak teks (text boxes).

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



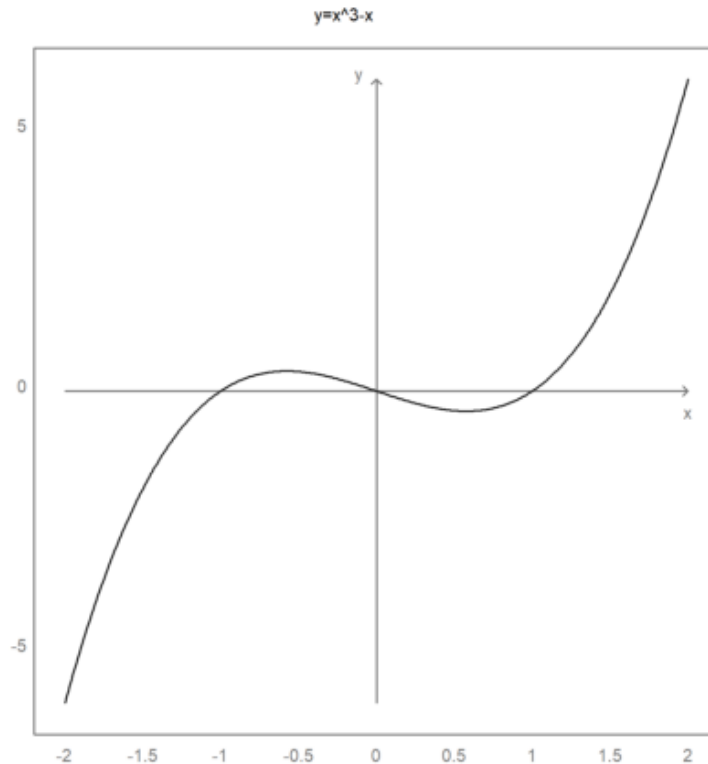
```
>plot2d(["exp(x)", "1+x"],color=[black,blue],style=["-","-.-"]):
```



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



```
label("y",0,6,pos="cl",color=gray); ...
reset():
```



Untuk kontrol yang lebih mendetail, sumbu x dan sumbu y dapat dilakukan secara manual. Perintah `fullwindow()` memperluas jendela plot karena kita tidak lagi memerlukan tempat untuk label di luar jendela plot. Gunakan `shrinkwindow()` atau `reset()` untuk mengatur ulang ke default.

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

Berikut contoh lainnya, di mana digunakan string Unicode 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;)"):
```

Plotting Data 2D

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

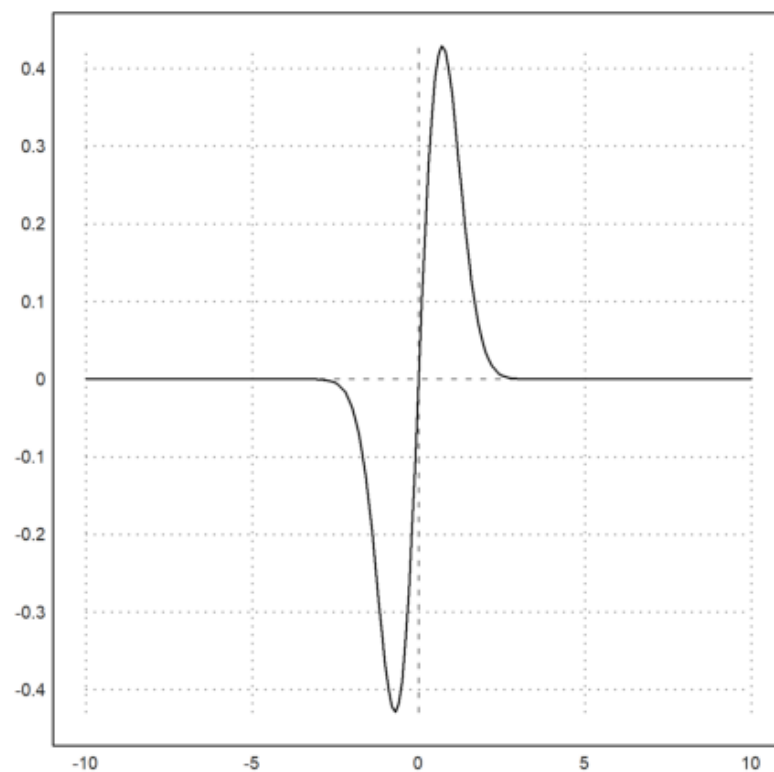
Memplot ekspresi hanyalah singkatan untuk plot data. Untuk plot data, Anda memerlukan satu atau lebih baris nilai x , dan satu atau lebih baris nilai y . Dari rentang dan nilai x , fungsi `plot2d` akan menghitung data untuk dipetakan, secara default dengan evaluasi adaptif terhadap fungsi tersebut. Untuk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

Namun, Anda dapat memasukkan data secara langsung.

- Gunakan vektor baris untuk x dan y untuk satu fungsi.
- Matriks untuk x dan y akan dipetakan 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):
```



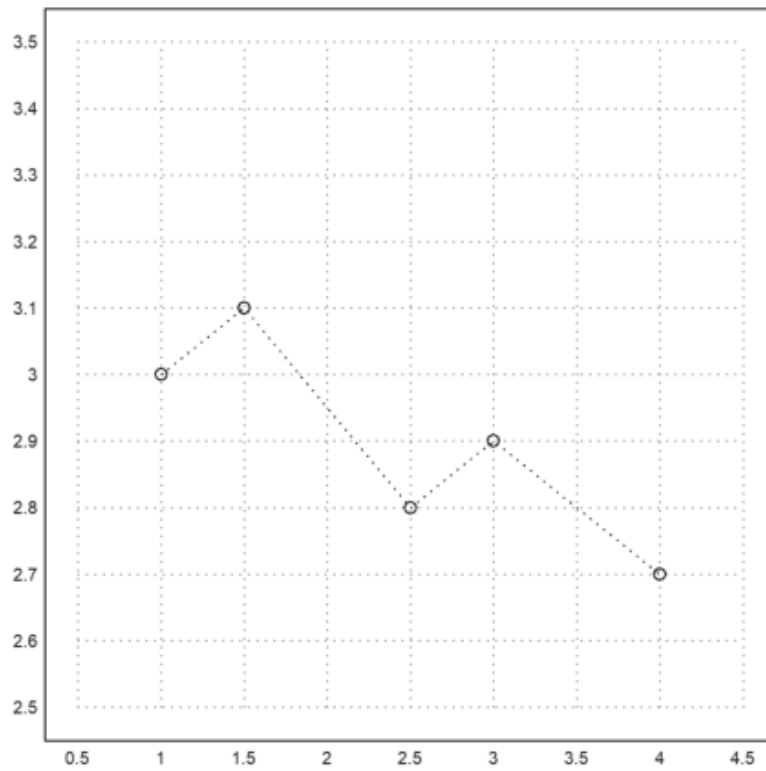
Data juga dapat dipetakan sebagai titik-titik. Gunakan points=true untuk ini. Plot bekerja seperti poligon, tetapi hanya menggambar sudut-sudutnya.

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

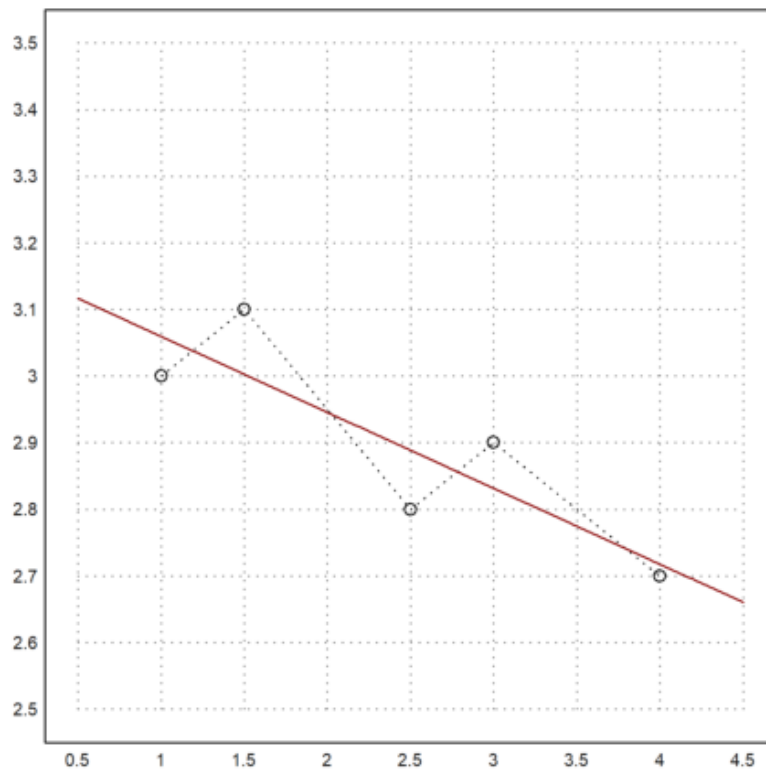
Untuk memetakan kumpulan titik gunakan points. Jika warna adalah vektor warna, setiap titik akan memiliki warna yang berbeda. Untuk matriks koordinat dan vektor kolom, warna berlaku untuk baris-baris matriks.

Parameter 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="."); // lines
>plot2d(xdata,ydata,>points,>add,style="o"): // add points
```



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



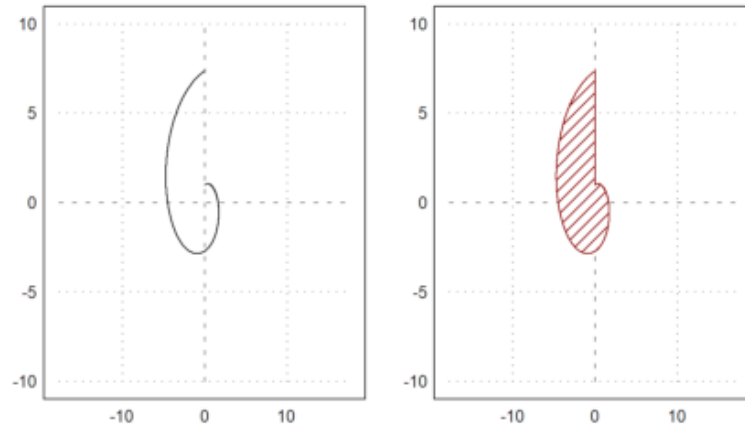
Menggambar Daerah Yang Dibatasi Kurva

Plot data sebenarnya adalah poligon. Kita juga bisa memplot kurva atau kurva terisi (filled curves).

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

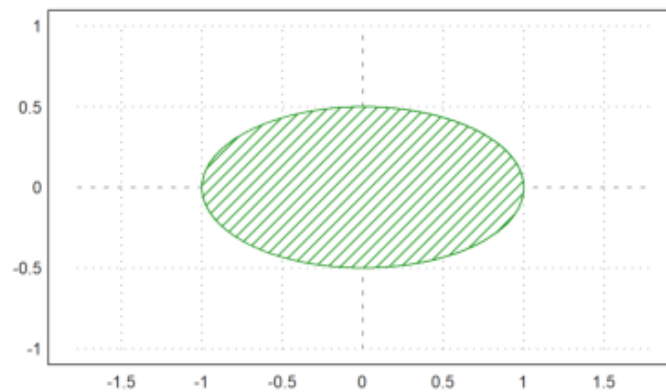
Warna pengisian ditentukan oleh argumen "fillcolor", dan opsional <outline mencegah menggambar batas untuk semua gaya kecuali gaya default.

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

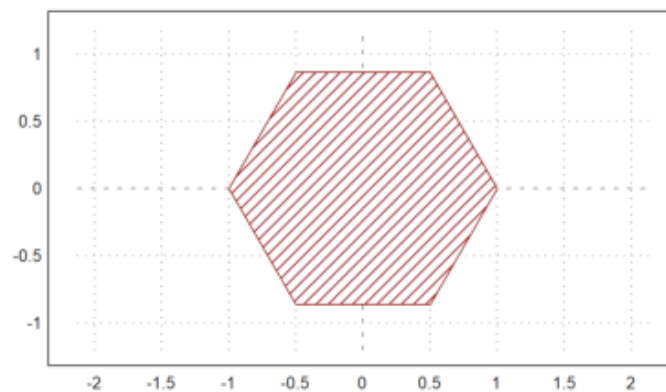


Dalam contoh berikut, kita memplot sebuah elips yang terisi dan dua heksagon terisi menggunakan kurva tertutup dengan 6 titik dan gaya pengisian yang berbeda.

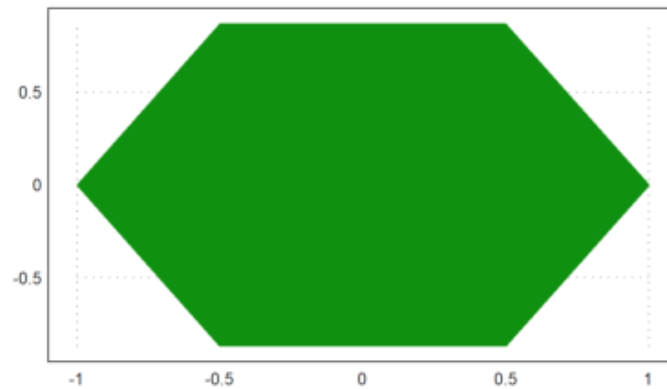
```
>x=linspace(0,2pi,1000); plot2d(sin(x),cos(x)*0.5,r=1,>filled,style="/"):
```



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



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

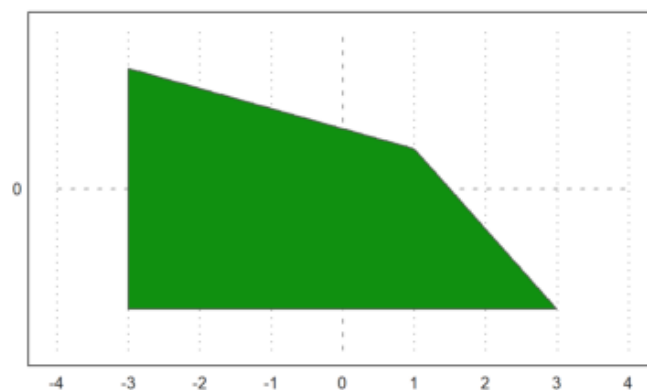


Contoh berikut adalah sebuah septagon, yang kita buat dengan 7 titik pada lingkaran satuan.

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

Berikut adalah set nilai maksimal dari empat kondisi linear yang kurang dari atau sama dengan 3. Ini adalah $A[k].v \leq 3$ untuk semua baris dari 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 kita untuk menghasilkan tabel fungsi dengan mudah.

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

Sekarang kita memiliki vektor x dan y dari nilai. Fungsi plot2d() dapat menggambar nilai-nilai ini sebagai kurva yang menghubungkan titik-titik tersebut. Plot dapat diisi. Dalam hal ini, hasilnya menjadi indah karena aturan pengisian, yang digunakan untuk mengisi.

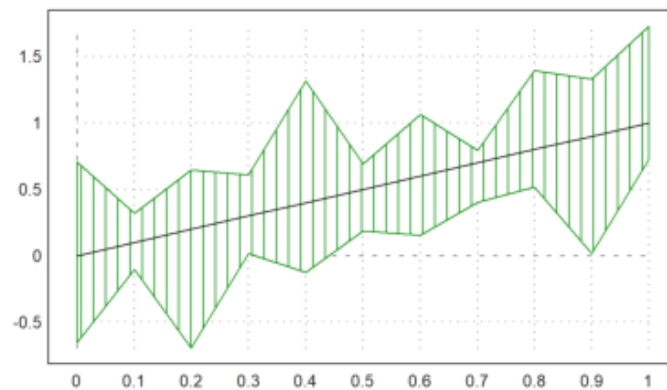
```
>plot2d(x,y,<grid,<frame,>filled):
```



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

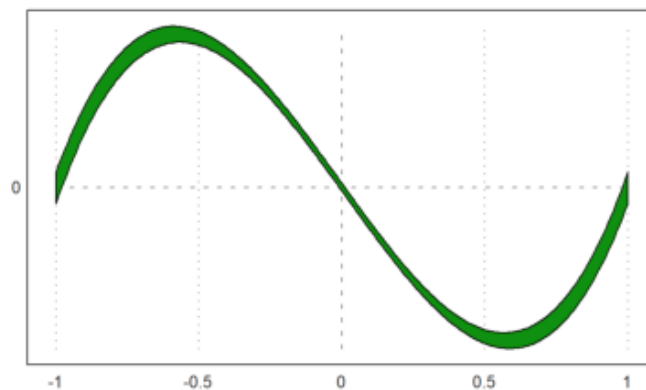
Ini berguna untuk menggambarkan kesalahan perhitungan. Namun, ini juga dapat digunakan untuk memplot kesalahan statistik.

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



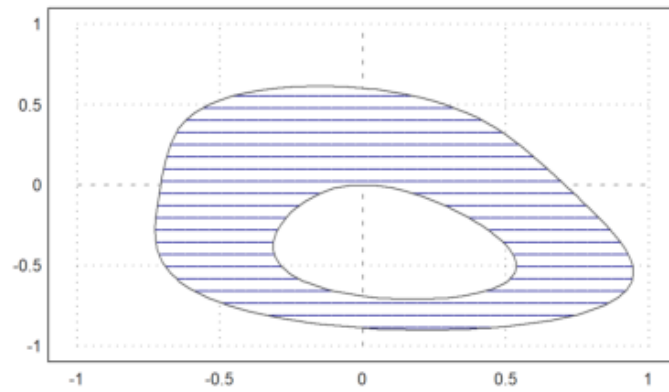
Jika x adalah vektor yang diurutkan, dan y adalah vektor interval, maka `plot2d()` akan memplot rentang interval yang terisi di dalam bidang. Gaya pengisian sama dengan gaya poligon.

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



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

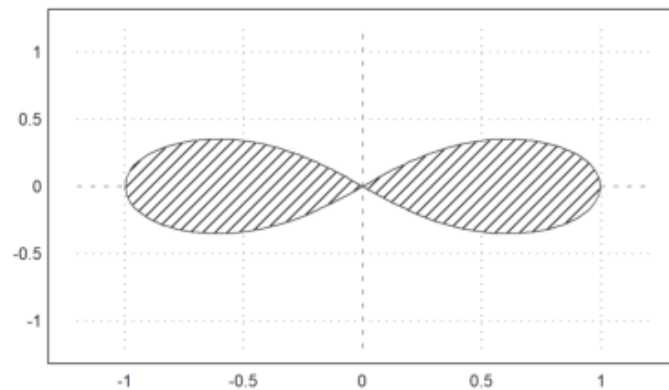
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=[0;1],style="-",color=blue): // 0 <= f(x,y) <= 1
```



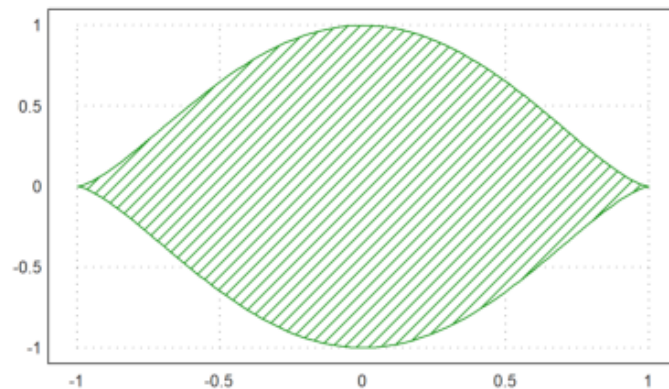
Kita juga dapat mengisi rentang nilai seperti:

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

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



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



Grafik Fungsi Parametrik

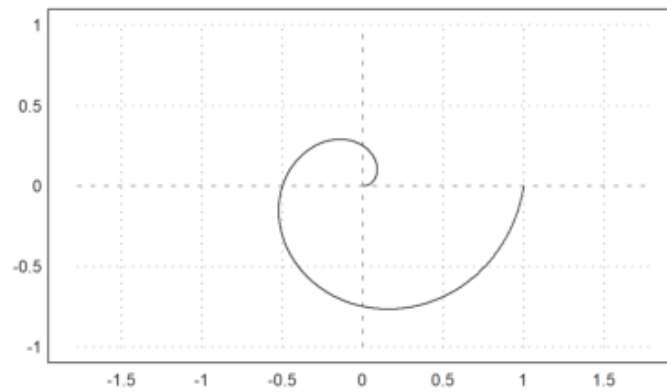
Nilai x tidak perlu diurutkan. (x, y) hanya menggambarkan sebuah kurva. Jika x diurutkan, maka kurva tersebut adalah grafik suatu fungsi.

Pada contoh berikut, kita menggambar spiral.

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

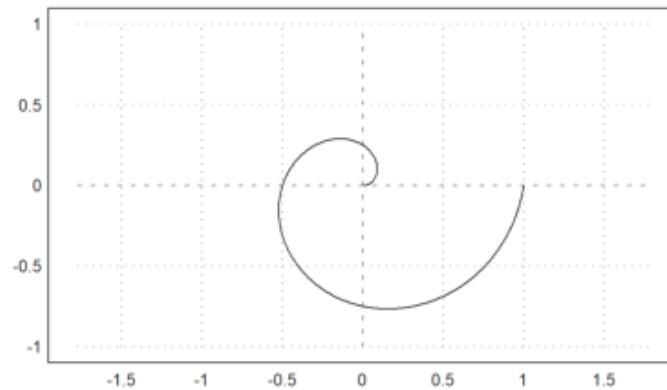
Kita perlu menggunakan banyak titik untuk tampilan yang halus atau menggunakan fungsi `adaptive()` untuk mengevaluasi ekspresi (lihat fungsi `adaptive()` untuk lebih detail).

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

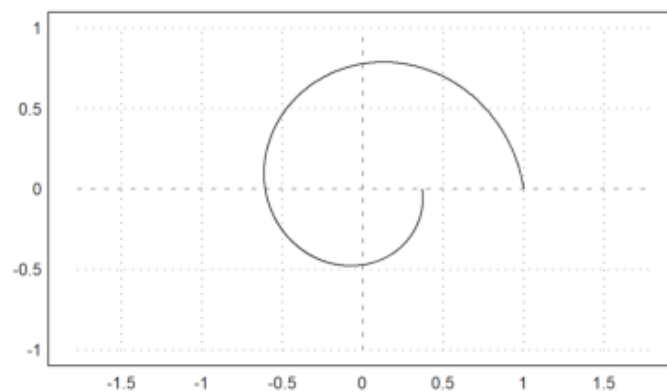


Alternatifnya, dimungkinkan menggunakan dua ekspresi untuk kurva. Berikut ini adalah plot yang sama seperti di atas.

```
>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(2*pi*t); y=r*sin(2*pi*t);
>plot2d(x,y,r=1):
```



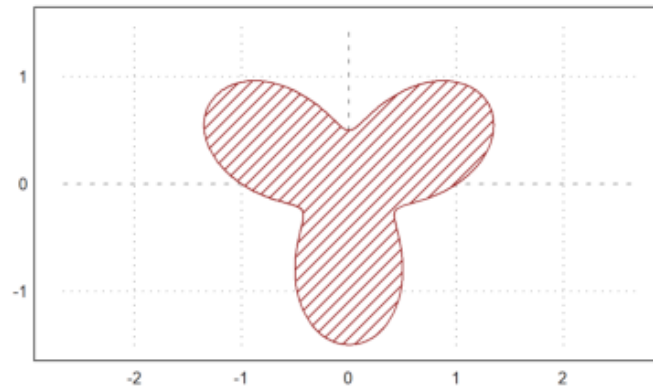
Pada contoh berikut, kita menggambar kurva:

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

dengan

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


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

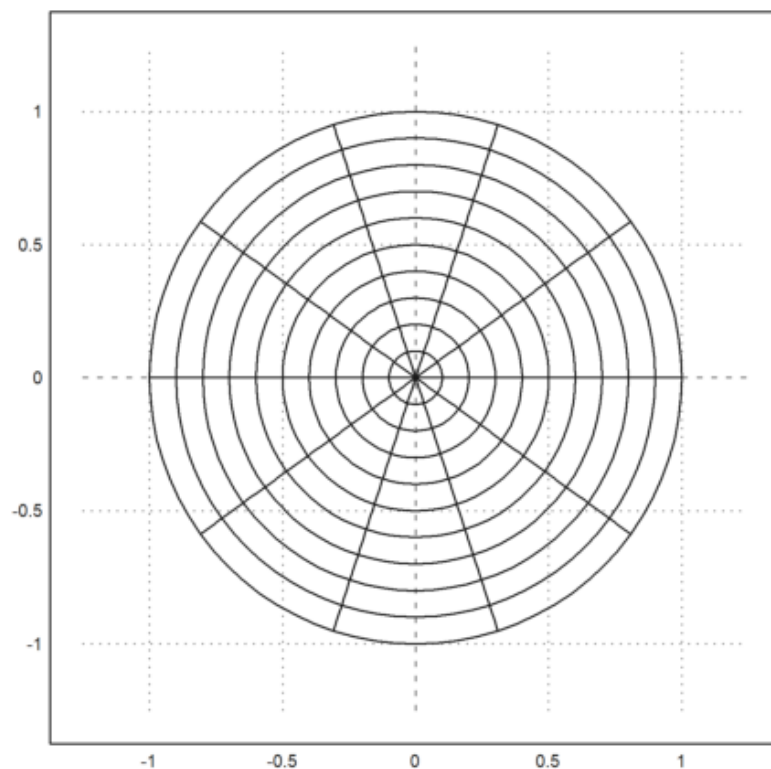


Menggambar Grafik Bilangan Kompleks

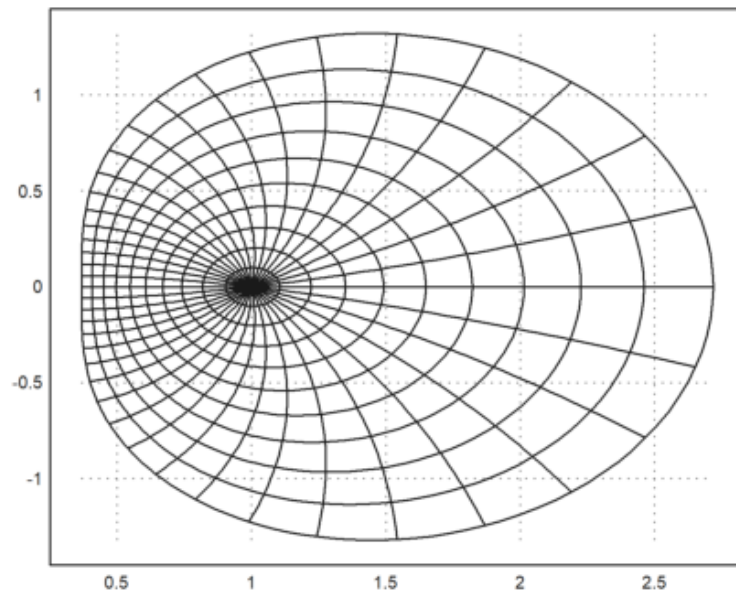
Sebuah array dari bilangan kompleks juga dapat diplotkan. Titik-titik grid akan terhubung. Jika sejumlah garis grid ditentukan (atau vektor 1x2 dari garis grid) dalam argumen `cgrid`, hanya garis-garis grid tersebut yang akan terlihat. Matriks bilangan kompleks secara otomatis akan diplot sebagai grid di bidang kompleks.

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

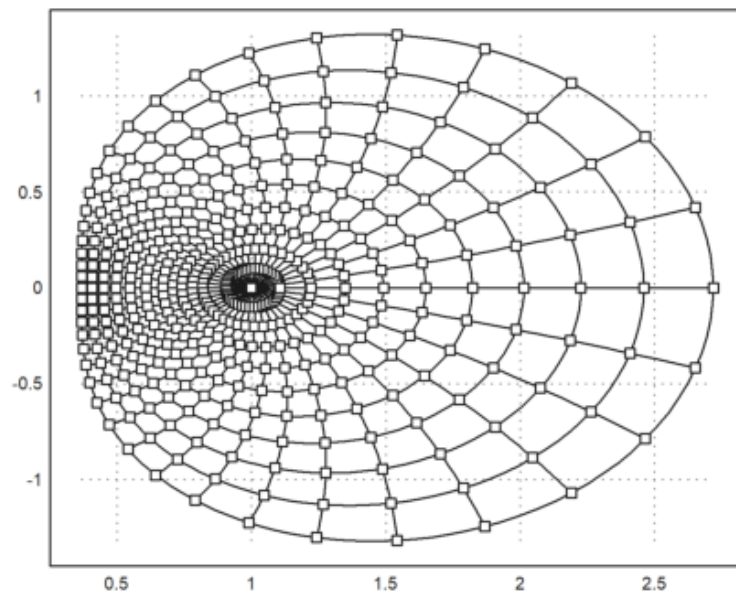
```
>aspect(); r=linspace(0,1,50); a=linspace(0,2pi,80)'; z=r*exp(I*a);...
plot2d(z,a=-1.25,b=1.25,c=-1.25,d=1.25,cgrid=10):
```



```
>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 di bidang kompleks dengan bagian nyata dan bagian imajiner.

Pada contoh ini, kita menggambar lingkaran satuan dengan

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

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

Plot Statistik

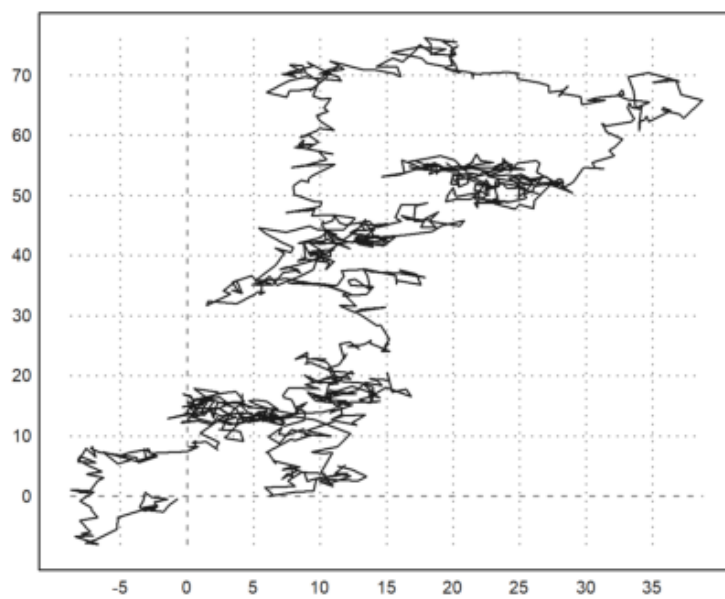
Ada banyak fungsi yang dikhususkan untuk plot statistik. Salah satu plot yang sering digunakan adalah plot kolom. Penjumlahan kumulatif dari nilai yang terdistribusi normal 0-1 menghasilkan gerak acak.

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

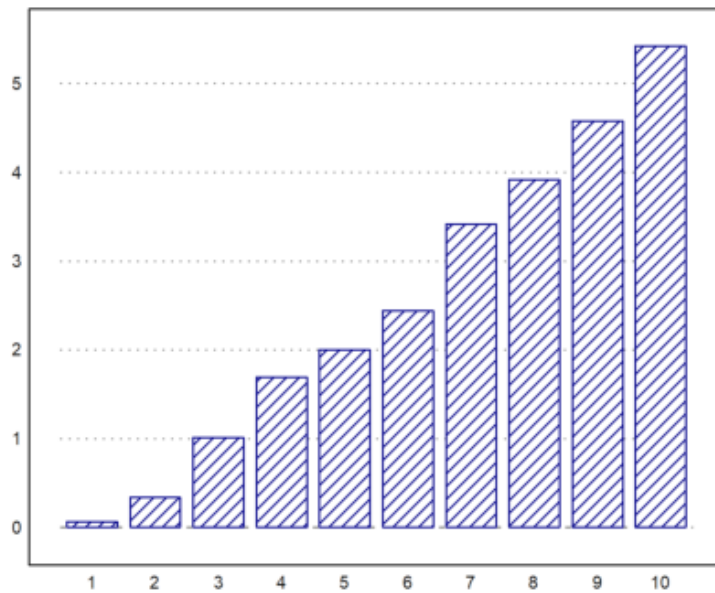


Dengan menggunakan dua baris, kita menunjukkan gerak acak dalam dua dimensi.

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

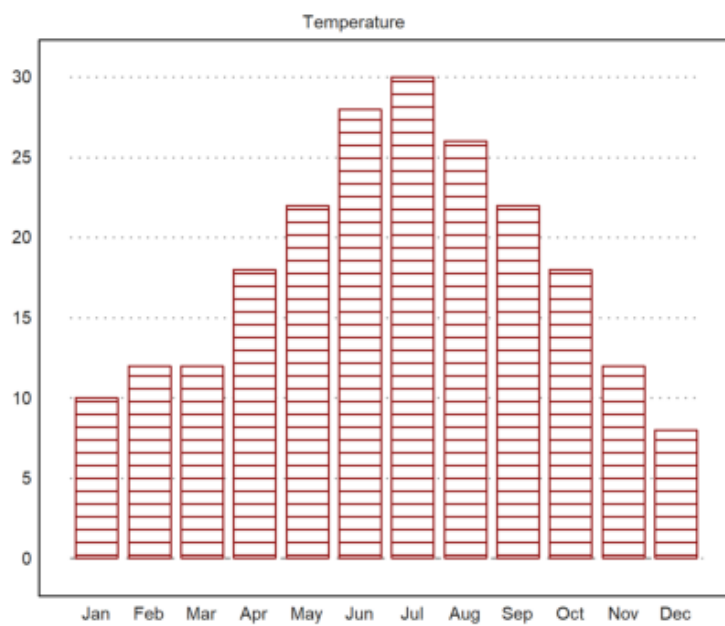


```
>columnplot(cumsum(random(10)),style="/",color=blue):
```

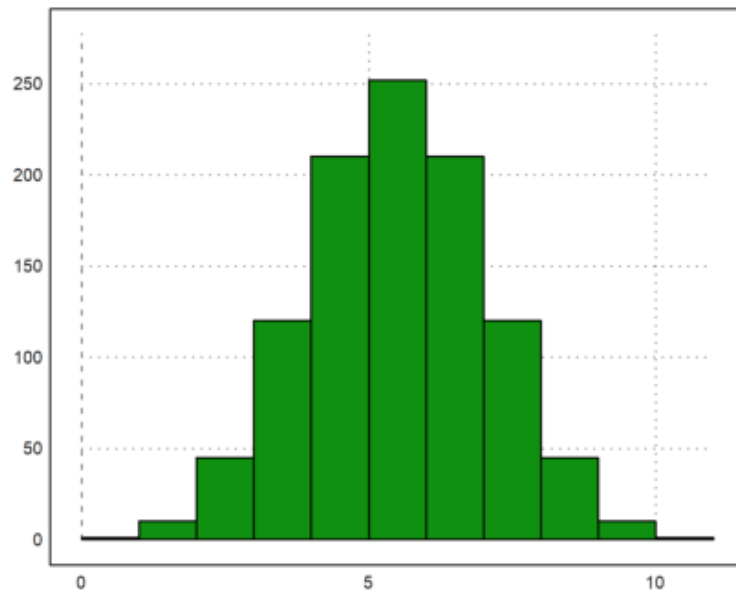


Plot ini juga dapat menampilkan string sebagai label.

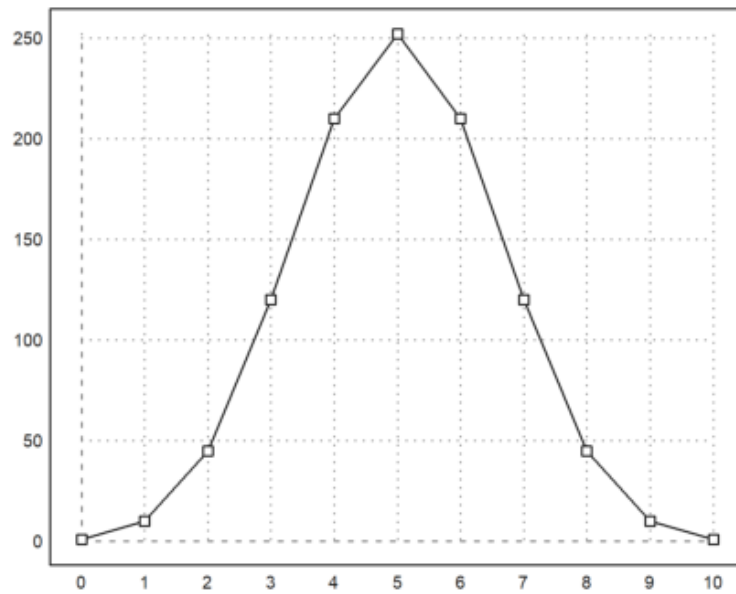
```
>months=["Jan", "Feb", "Mar", "Apr", "May", "Jun", ...
        "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"];
>values=[10,12,12,18,22,28,30,26,22,18,12,8];
>columnsplot(values,lab=months,color=red,style="-");
>title("Temperature"):
```



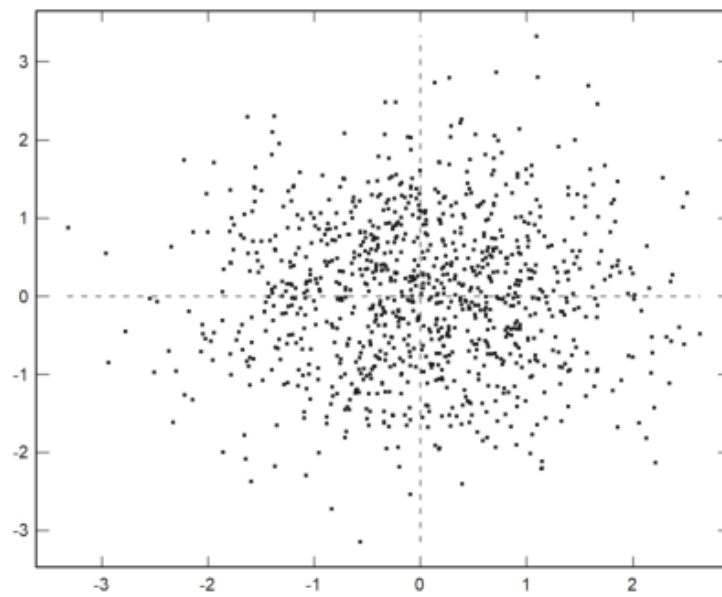
```
>k=0:10;
>plot2d(k,bin(10,k),>bar):
```



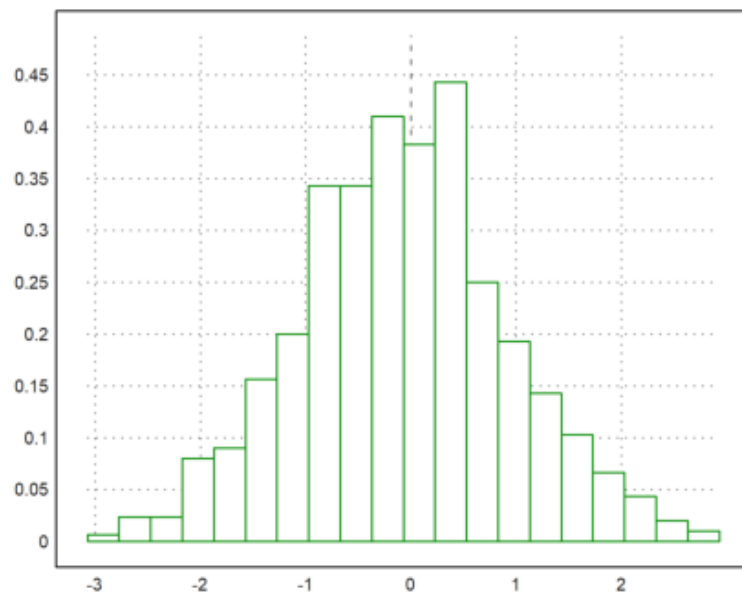
```
>plot2d(k,bin(10,k)); plot2d(k,bin(10,k),>points,>add):
```



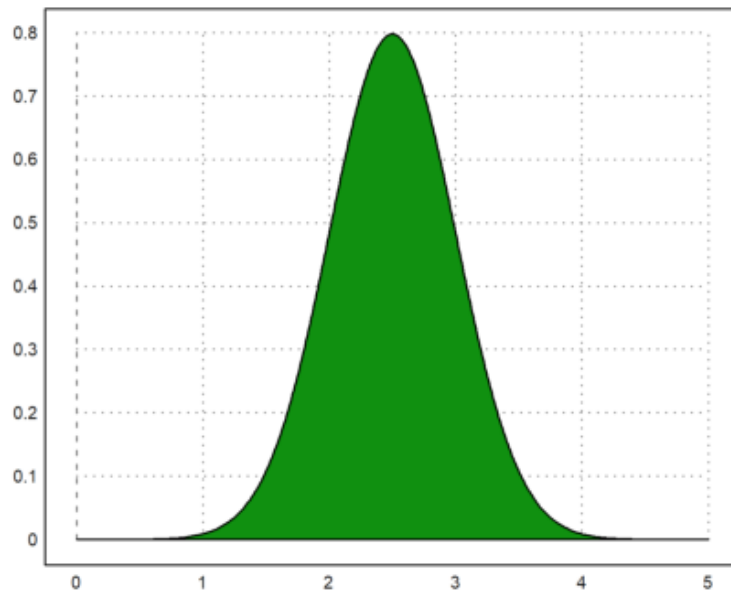
```
>plot2d(normal(1000),normal(1000),>points,grid=6,style=".."):
```



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

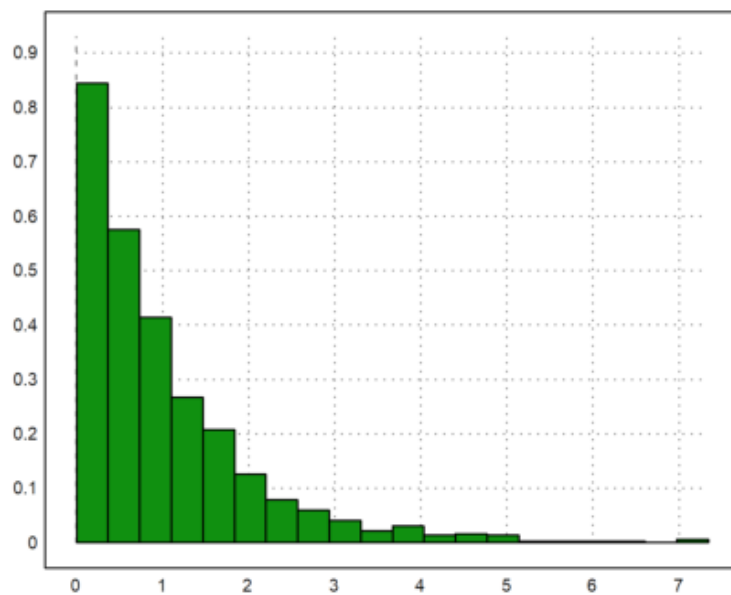


```
>plot2d("qnormal",0,5;2.5,0.5,>filled):
```



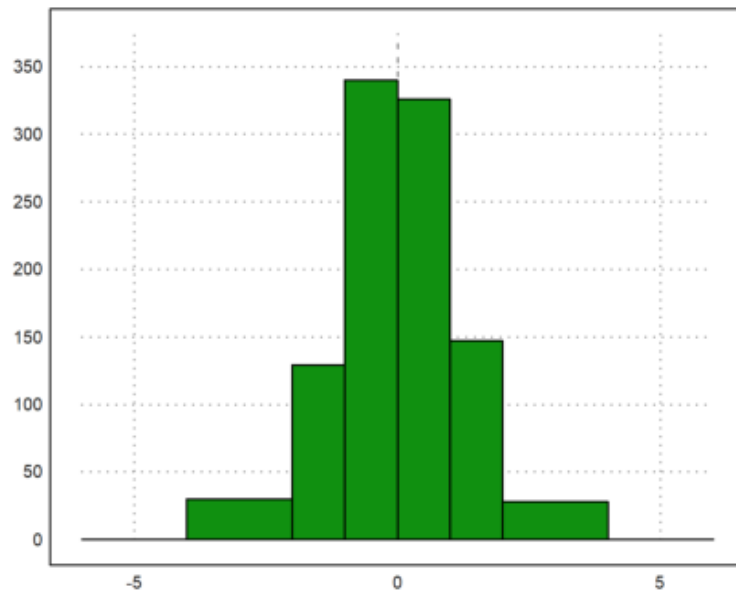
Distribusi eksperimen statistik dapat menggunakan `distribution=n` dengan `plot2d`.

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



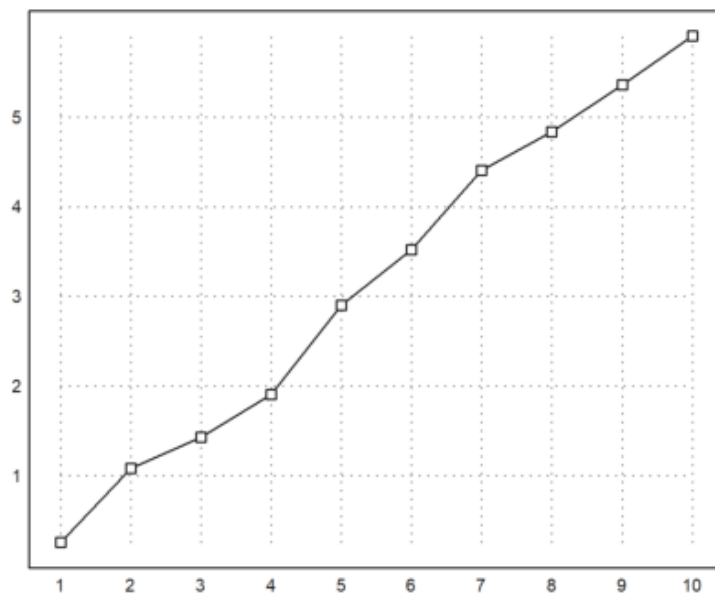
Atau kita bisa menghitung distribusi dari data dan memplot hasilnya dengan menggunakan bar dalam `plot2d` atau column plot.

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

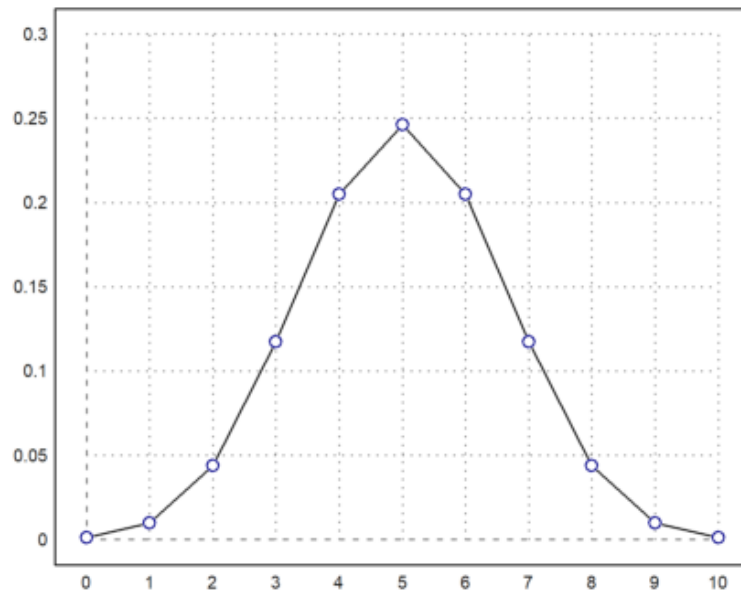


Fungsi `statplot()` menetapkan gaya dengan string sederhana.

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

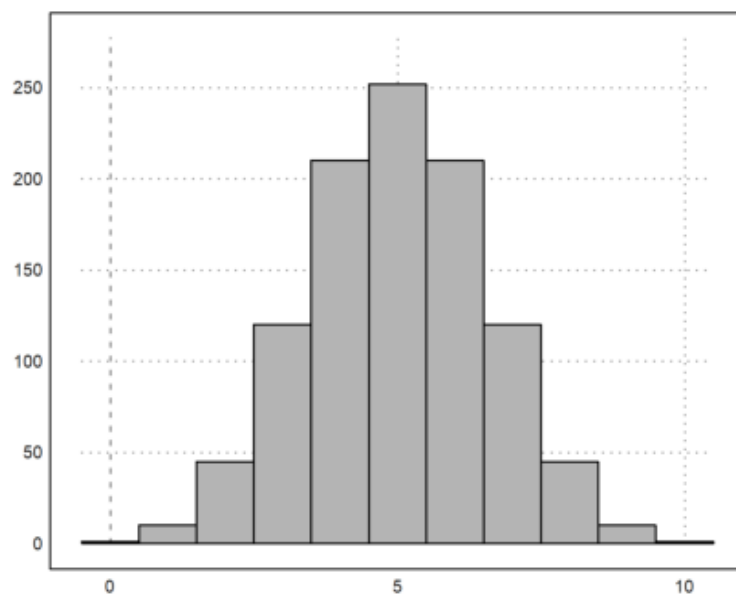


```
>n=10; i=0:n; ...
plot2d(i,bin(n,i)/2^n,a=0,b=10,c=0,d=0.3); ...
plot2d(i,bin(n,i)/2^n,points=true,style="ow",add=true,color=blue):
```

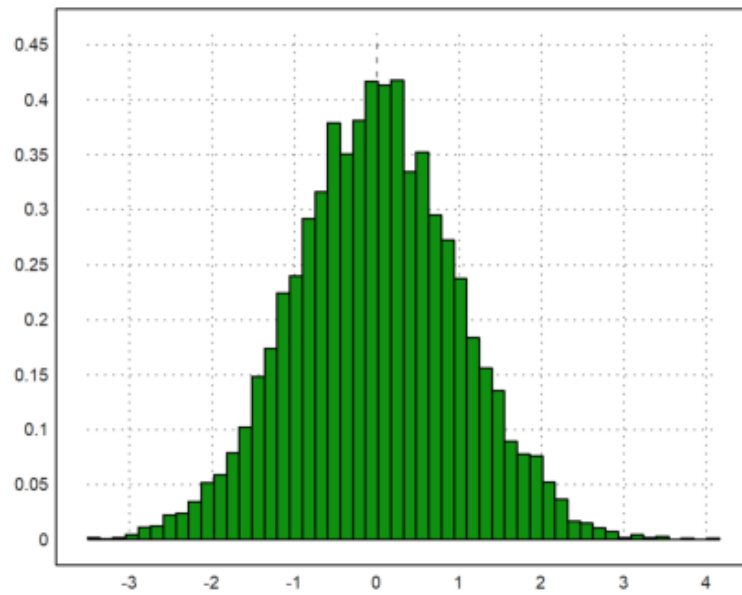
Selain itu, data dapat diplot sebagai batang (bars). Dalam hal ini, x harus diurutkan dan satu elemen lebih panjang dari y. Batang akan diperpanjang dari $x[i]$ hingga $x[i+1]$ dengan nilai $y[i]$. Jika x memiliki ukuran yang sama dengan y, maka akan diperpanjang dengan satu elemen dengan jarak terakhir.

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

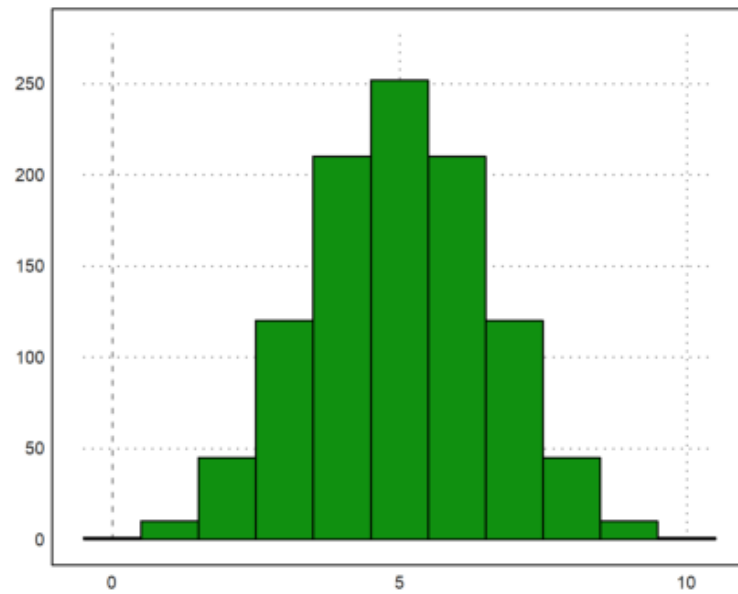


Data untuk plot batang (bar=1) dan histogram (histogram=1) dapat diberikan secara eksplisit dalam xv dan yv, atau dihitung dari distribusi empiris dalam xv dengan distribution=n. Histogram dari nilai-nilai xv akan dihitung secara otomatis dengan histogram. Jika even=true ditentukan, nilai-nilai xv akan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

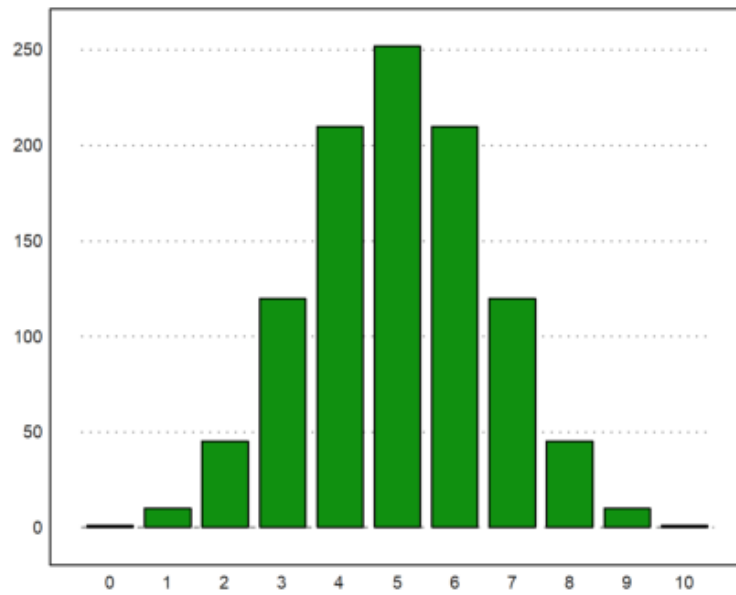
```
>plot2d(normal(10000),distribution=50):
```



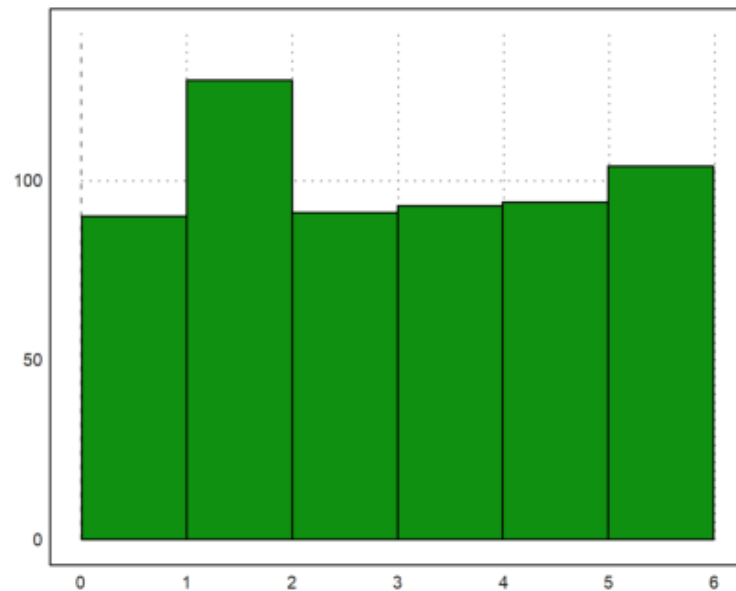
```
>k=0:10; m=bin(10,k); x=(0:11)-0.5; plot2d(x,m,>bar):
```



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

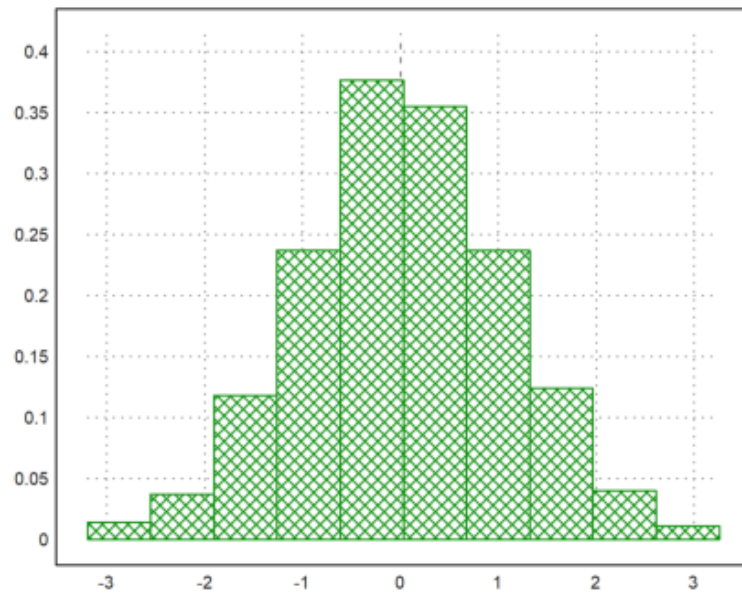


```
>plot2d(random(600)*6,histogram=6):
```



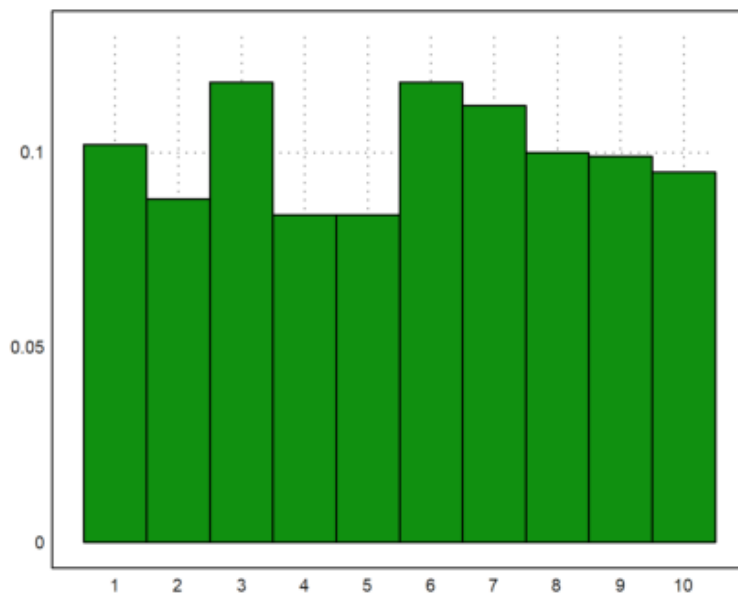
Untuk distribusi, terdapat parameter `distribution=n`, yang secara otomatis menghitung nilai dan mencetak distribusi relatif dengan `n` sub-interval.

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



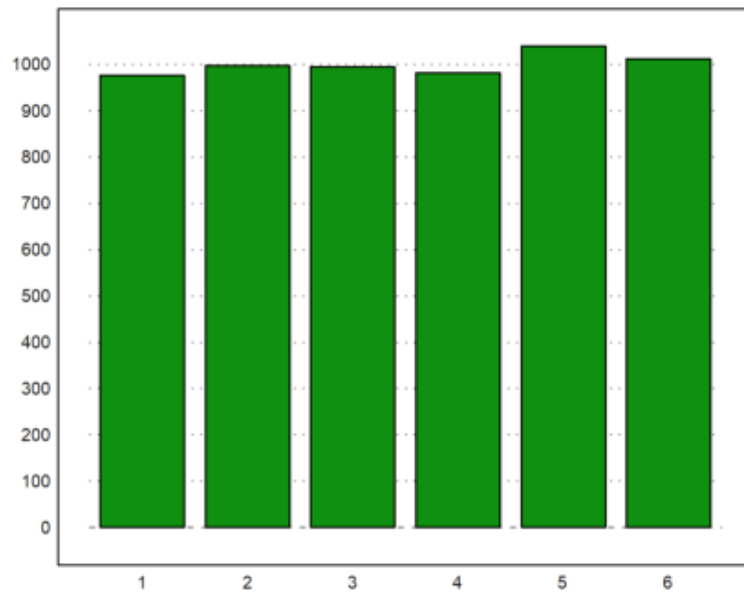
Dengan parameter `even=true`, ini akan menggunakan interval bilangan bulat.

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

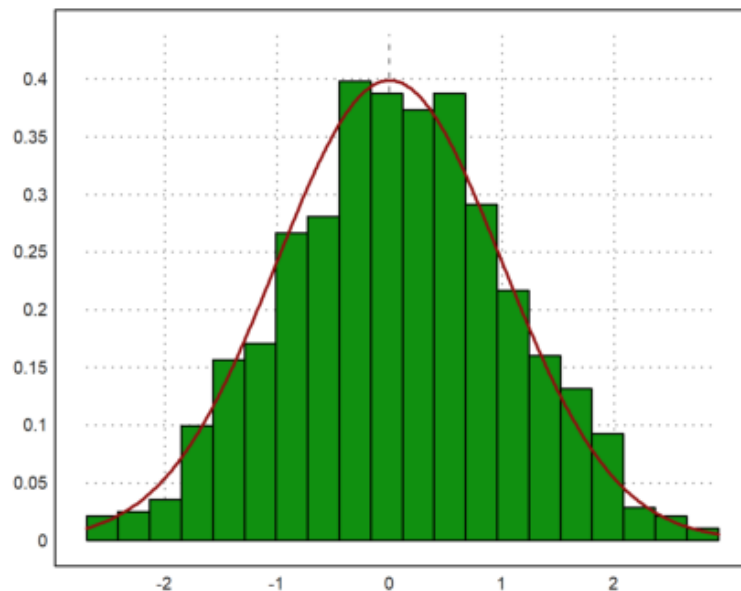


Perlu diingat bahwa ada banyak plot statistik yang mungkin berguna. Lihatlah tutorial tentang statistik.

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

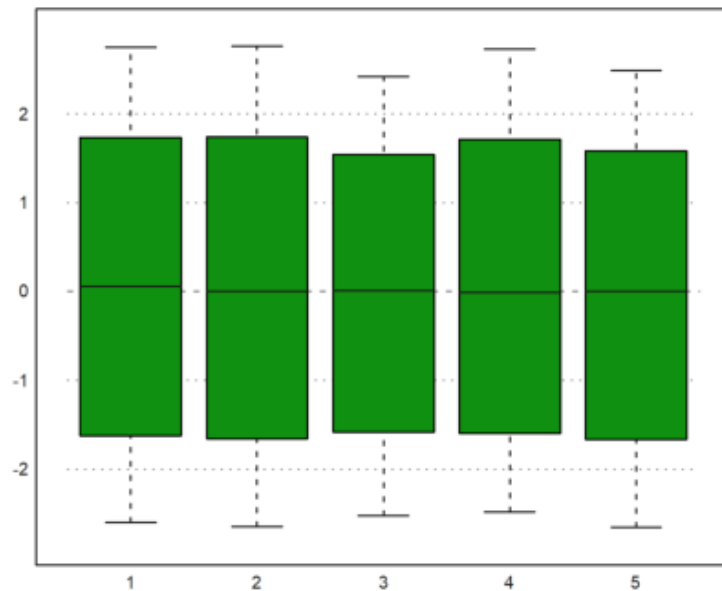


```
>plot2d(normal(1,1000),>distribution); ...
  plot2d("qnormal(x)",color=red,thickness=2,>add):
```



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

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



Fungsi Implisit

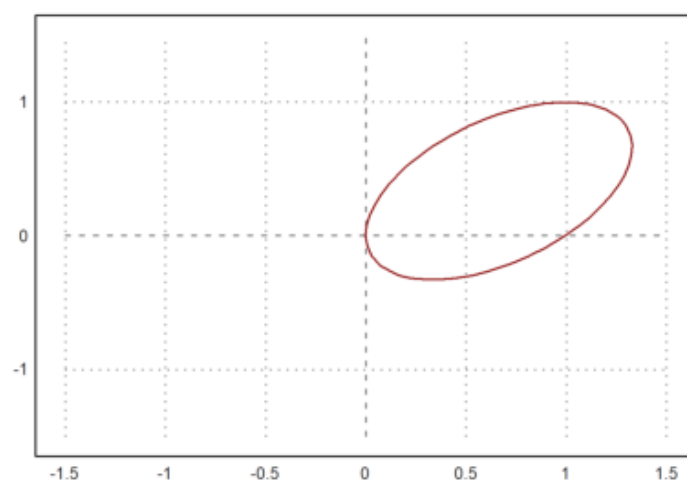
Plot implisit menampilkan garis level yang menyelesaikan persamaan $f(x,y)=\text{level}$, dimana "level" dapat berupa nilai tunggal atau vektor nilai. Jika $\text{level}=\text{"auto"}$, akan ada nc garis level, yang akan tersebar merata antara nilai minimum dan maksimum fungsi. Warna yang lebih gelap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

Euler dapat menandai garis level dari fungsi apapun seperti

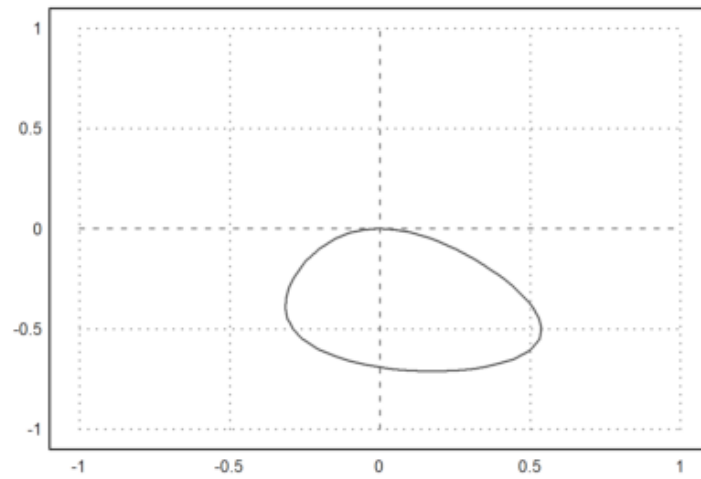
$$f(x,y) = c$$

Untuk satu atau lebih konstanta c , Anda bisa menggunakan `plot2d()` dengan plot implisitnya di bidang. Parameter untuk c adalah $\text{level}=c$, di mana c bisa menjadi vektor dari garis level. Selain itu, skema warna dapat digambar di latar belakang untuk menunjukkan nilai fungsi untuk setiap titik dalam plot. Parameter n menentukan tingkat ketelitian plot.

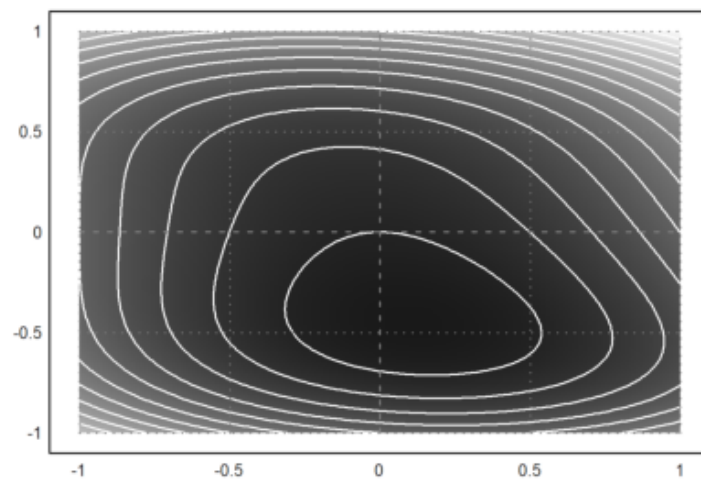
```
>aspect(1.5);
>plot2d("x^2+y^2-x*y-x",r=1.5,level=0,contourcolor=red):
```



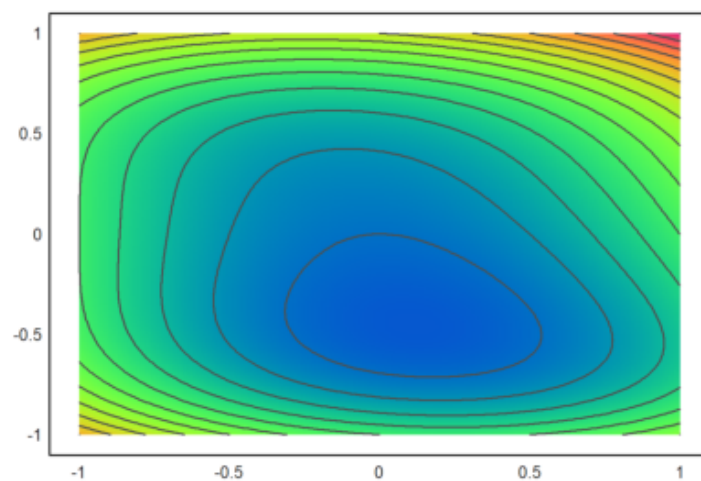
```
>expr := "2*x^2+x*y+3*y^4+y"; // definisikan ekspresi f(x,y)
>plot2d(expr,level=0): // solusi dari f(x,y)=0
```



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

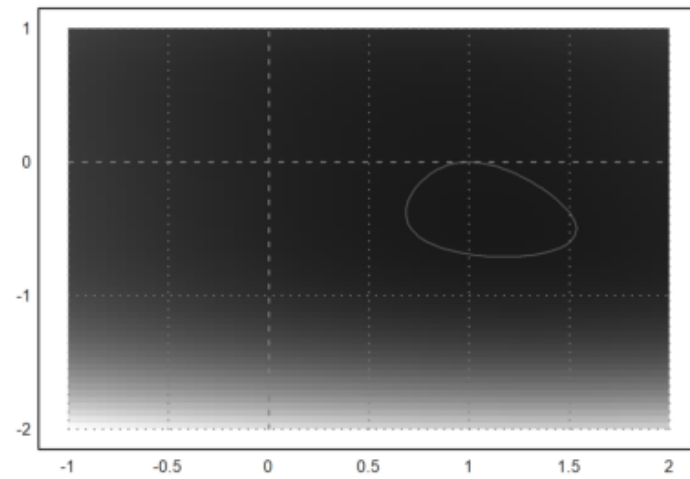


```
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=200,grid=4): // tampilan yang lebih baik
```

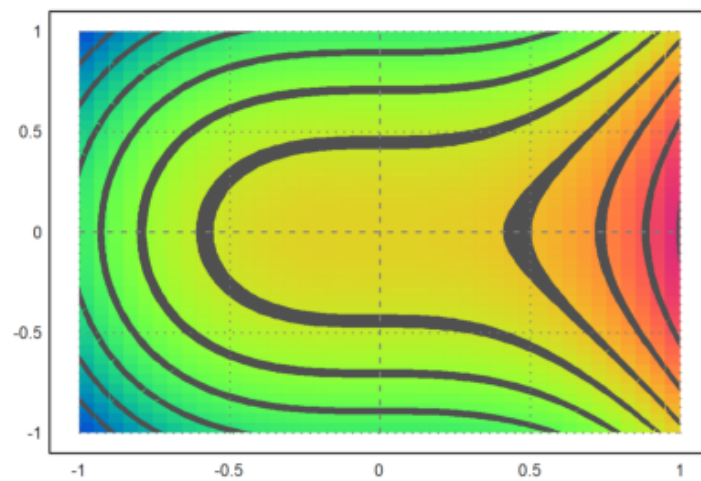


Ini juga bekerja untuk plot data. Namun, Anda harus menentukan rentang untuk label sumbu.

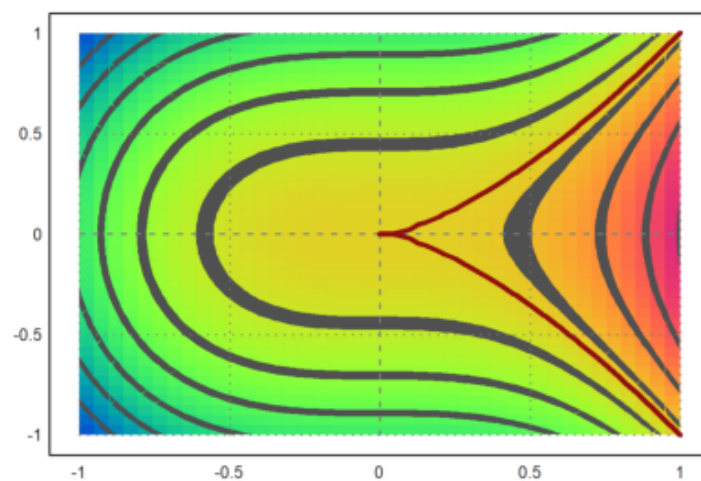
```
>x=-2:0.05:1; y=x'; z=expr(x,y);  
>plot2d(z,level=0,a=-1,b=2,c=-2,d=1,>hue):
```



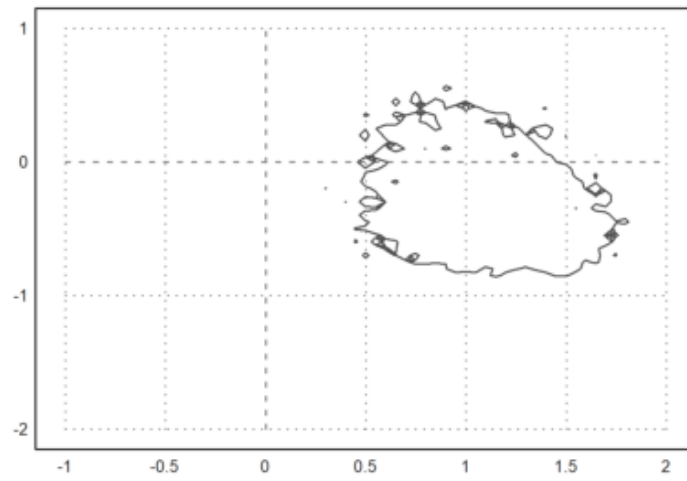
```
>plot2d("x^3-y^2",>contour,>hue,>spectral):
```



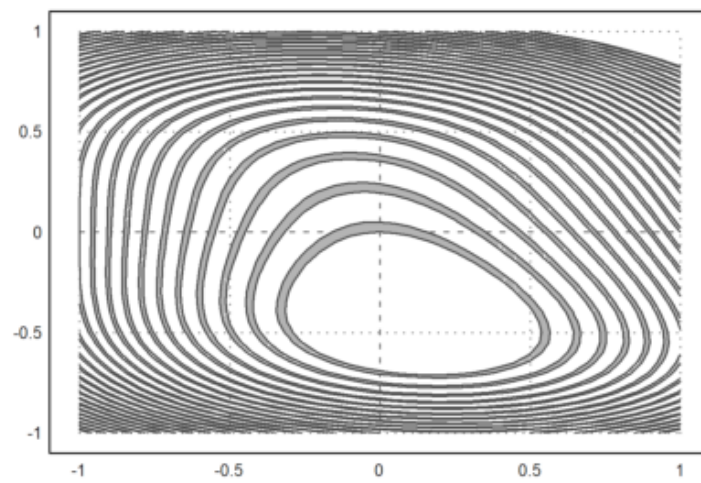
```
>plot2d("x^3-y^2",level=0,contourwidth=3,>add,contourcolor=red):
```



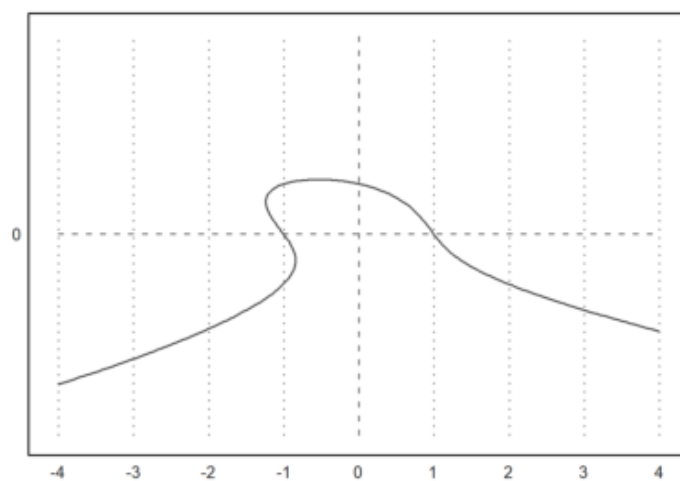
```
>z=z+normal(size(z))*0.2;
>plot2d(z,level=0.5,a=-1,b=2,c=-2,d=1):
```

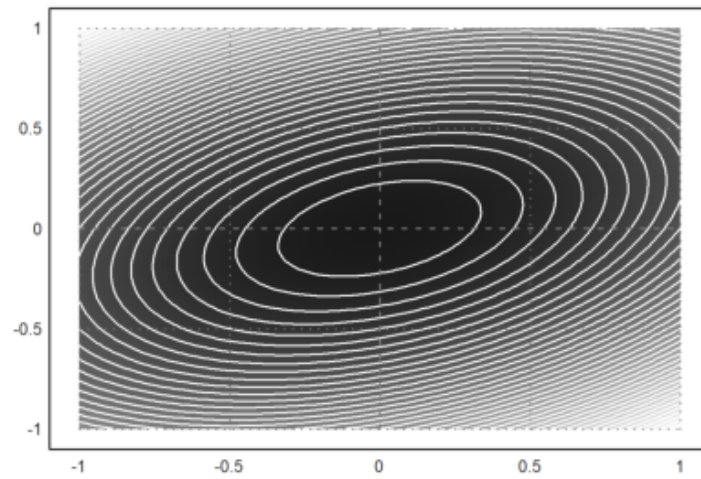
```
>plot2d(expr,level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
```



```
>plot2d("x^2+y^3+x*y",level=1,r=4,n=100):
```



```
>plot2d("x^2+2*y^2-x*y",level=0:0.1:10,n=100,contourcolor=white,>hue):
```



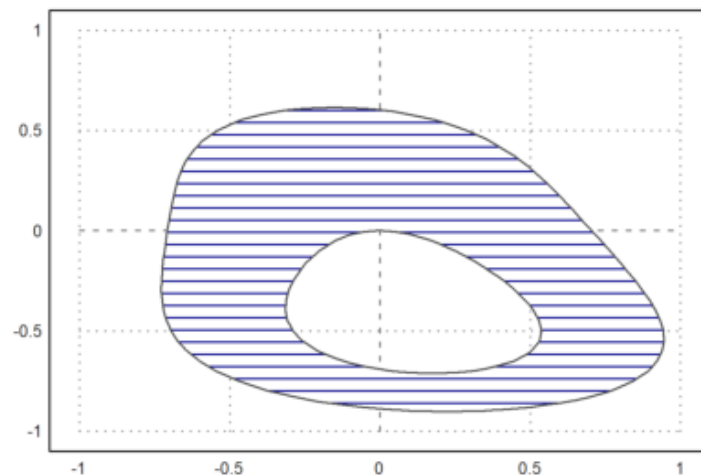
Dimungkinkan juga untuk mengisi himpunan

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

dengan rentang level

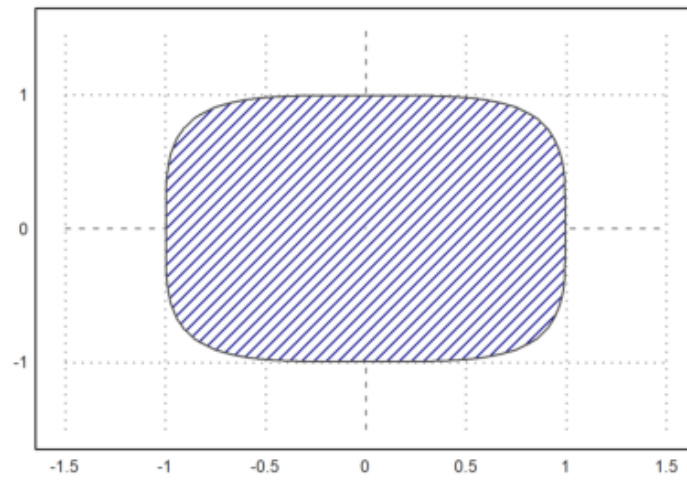
Dimungkinkan untuk mengisi area nilai untuk fungsi tertentu. Untuk ini, level harus berupa matriks 2xn. Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

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

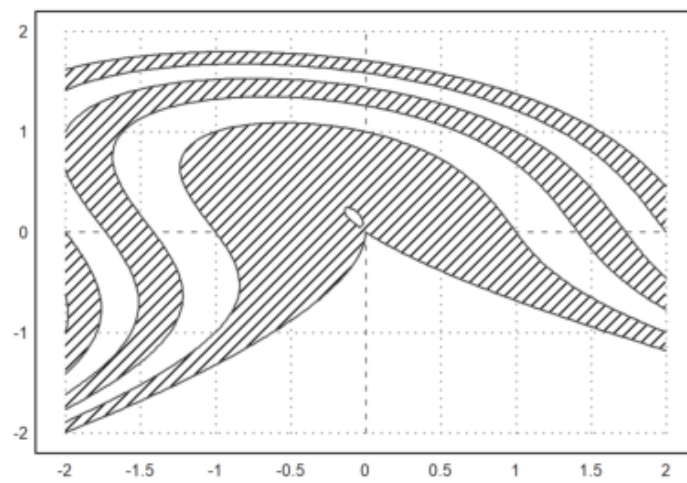


Plot implisit juga dapat menunjukkan rentang level. Maka level harus berupa matriks 2xn dari interval level, di mana baris pertama berisi awal dan baris kedua berisi akhir dari setiap interval. Sebagai alternatif, vektor baris sederhana dapat digunakan untuk level, dan parameter dl memperluas nilai level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

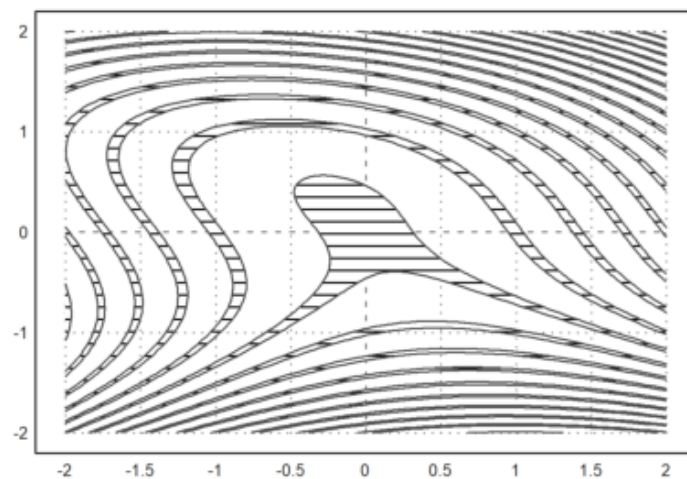
```
>plot2d("x^4+y^4", r=1.5, level=[0;1], color=blue, style="/"):
```



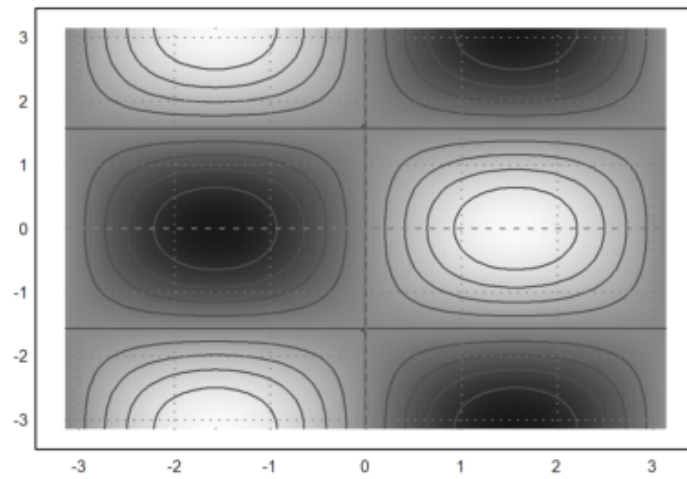
```
>plot2d("x^2+y^3+x*y",level=[0,2,4;1,3,5],style="/",r=2,n=100):
```



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



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

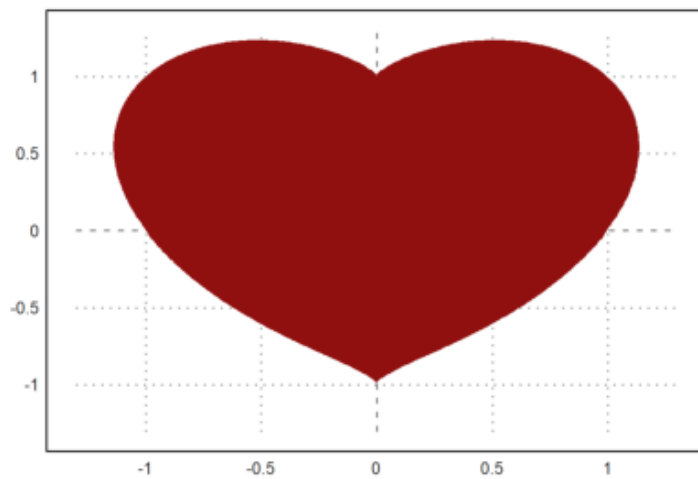


Dimungkinkan juga untuk menandai area

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

Ini dilakukan dengan menambahkan level dengan dua baris.

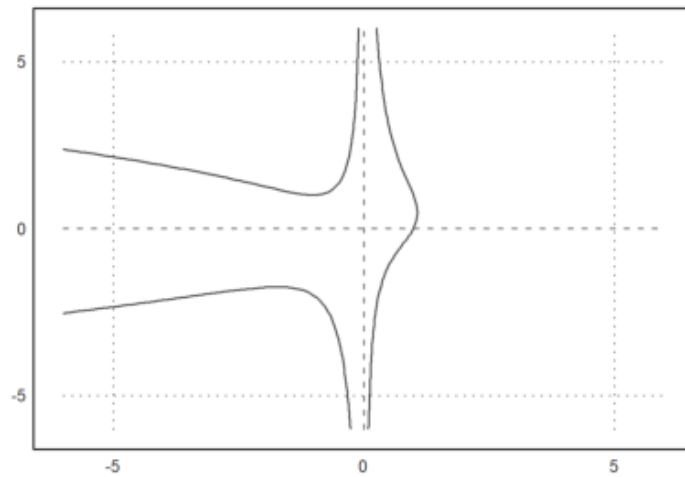
```
>plot2d("(x^2+y^2-1)^3-x^2*y^3",r=1.3, ...
  style="#",color=red,<outline, ...
  level=[-2;0],n=100):
```



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

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

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

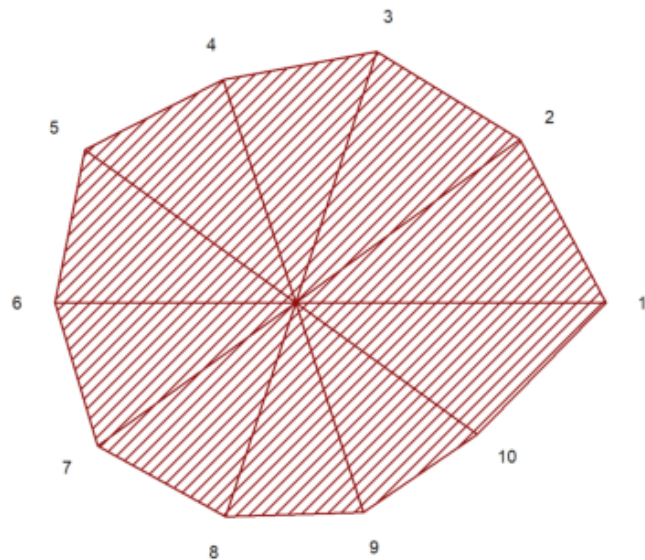


```
>function starplot1 (v, style="/", color=green, lab=none) ...
  if !holding() then clg; endif;
  w=window(); window(0,0,1024,1024);
  h=holding(1);
  r=max(abs(v))*1.2;
  setplot(-r,r,-r,r);
  n=cols(v); t=linspace(0,2pi,n);
  v=v[v[1]; c=v*cos(t); s=v*sin(t);
  cl=barcolor(color); st=barstyle(style);
  loop 1 to n
    polygon([0,c[#],c[#+1]], [0,s[#],s[#+1]],1);
    if lab!=none then
      rlab=v[#]+r*0.1;
      {col,row}=toscreen(cos(t[#])*rlab,sin(t[#])*rlab);
      ctext(""+lab[#],col,row-textheight()/2);
    endif;
  end;
  barcolor(cl); barstyle(st);
  holding(h);
  window(w);
endfunction
```

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

Kita memanggil reset sebelum menguji plot ini untuk mengembalikan default grafis. Ini tidak diperlukan jika Anda yakin plot Anda berfungsi.

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



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

Pada fungsi berikut, kita membuat plot impuls logaritmik. plot2d dapat membuat plot logaritmik, tetapi tidak untuk batang impuls.

```
>function logimpulseplot1 (x,y) ...
    {x0,y0}=makeimpulse(x,log(y)/log(10));
    plot2d(x0,y0,>bar,grid=0);
    h=holding(1);
    frame();
    xgrid(ticks(x));
    p=plot();
    for i=-10 to 10;
        if i<=p[4] and i>=p[3] then
            ygrid(i,yt="10^"+i);
        endif;
    end;
    holding(h);
endfunction
```

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

Fungsi wait(0) memaksa plot muncul di jendela grafis. Jika tidak, penggambaran 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 tombol apa saja untuk menghentikan animasi ini.

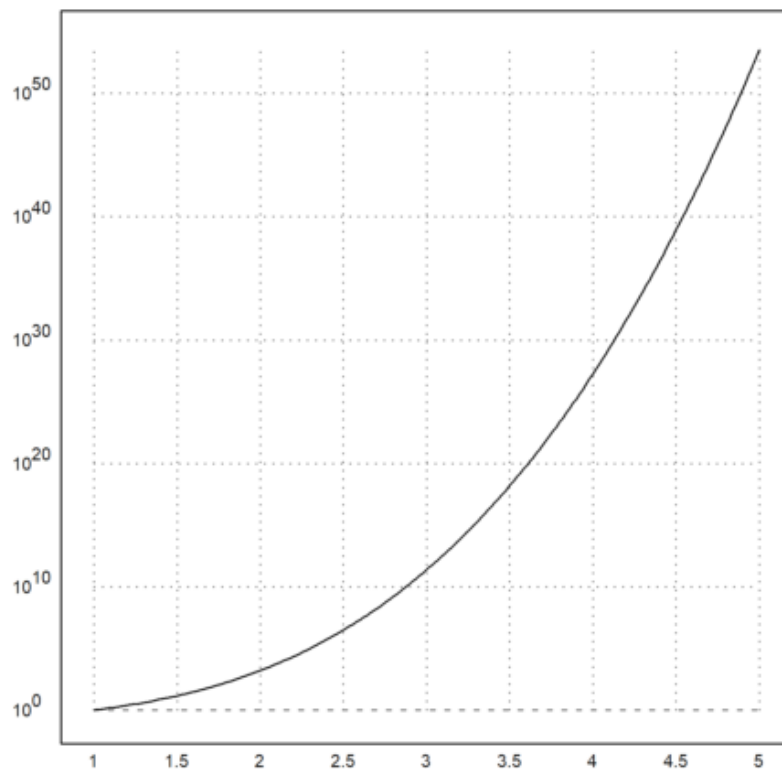
```
>animliss(2,3); // lihat hasilnya, jika sudah puas, tekan ENTER
```

Grafik Logaritmik

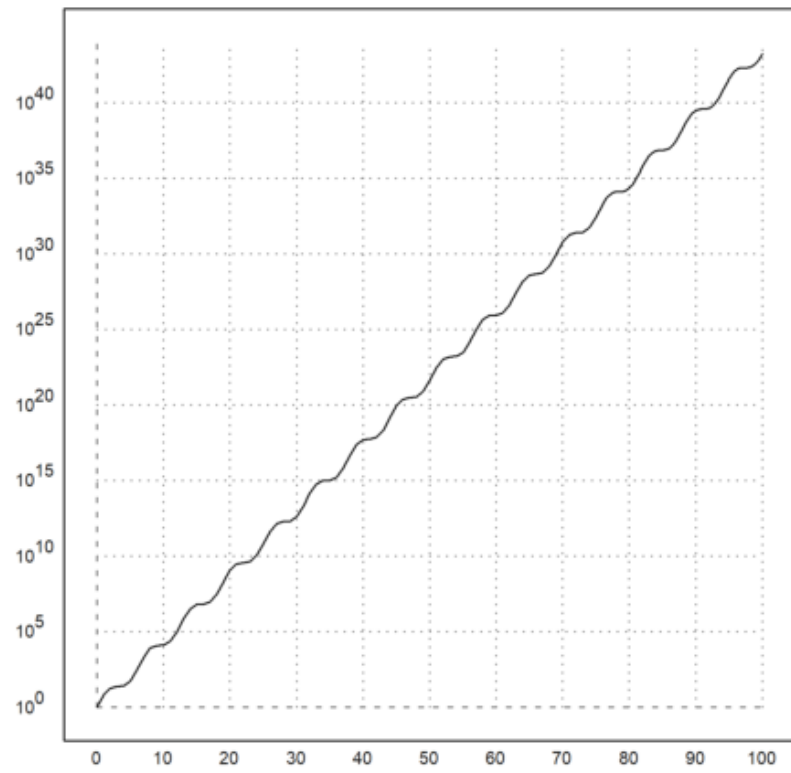
EMT menggunakan parameter `logplot` untuk skala logaritmik. Plot logaritmik dapat dibuat dengan menggunakan skala logaritmik pada sumbu y dengan `logplot=1`, atau menggunakan skala logaritmik pada sumbu x dan y dengan `logplot=2`, atau hanya pada sumbu x dengan `logplot=3`.

- `logplot=1`: skala logaritmik pada sumbu y
- `logplot=2`: skala logaritmik pada sumbu x dan y
- `logplot=3`: skala logaritmik pada sumbu x

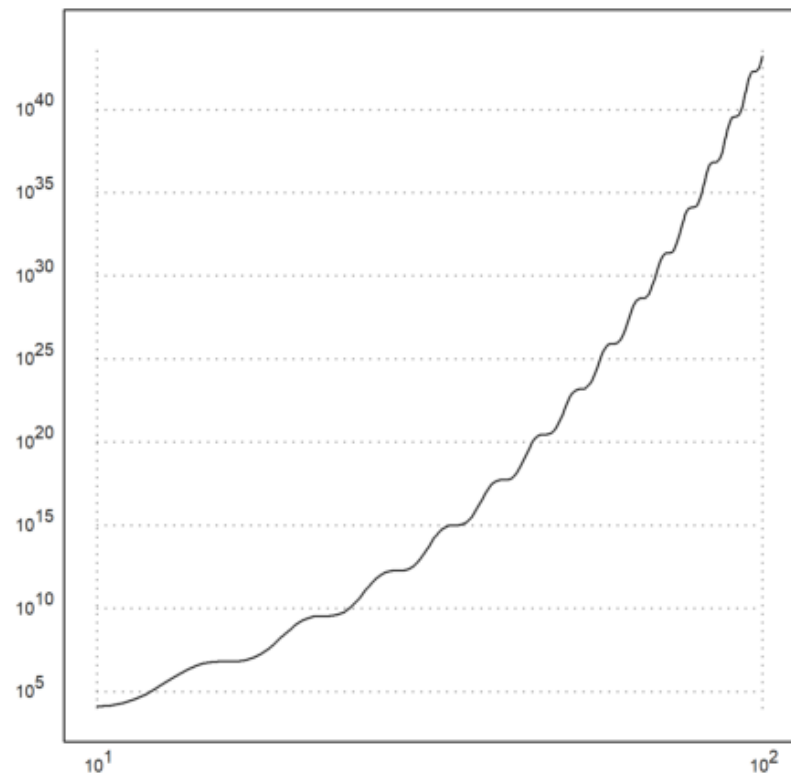
```
>plot2d("exp(x^3-x)*x^2",1,5,logplot=1):
```



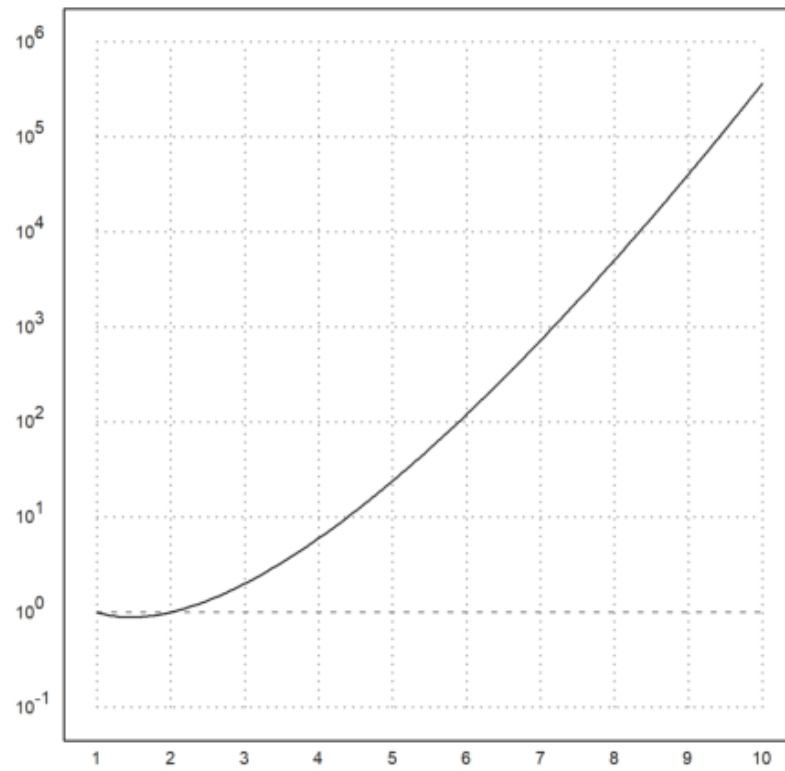
```
>plot2d("exp(x+sin(x))",0,100,logplot=1):
```



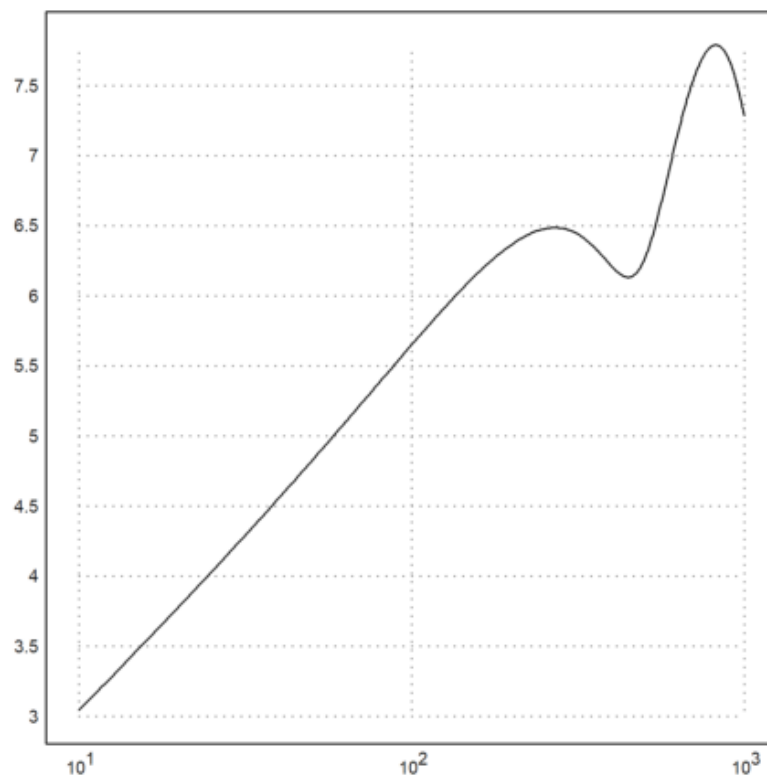
```
>plot2d("exp(x+sin(x))",10,100,logplot=2):
```



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

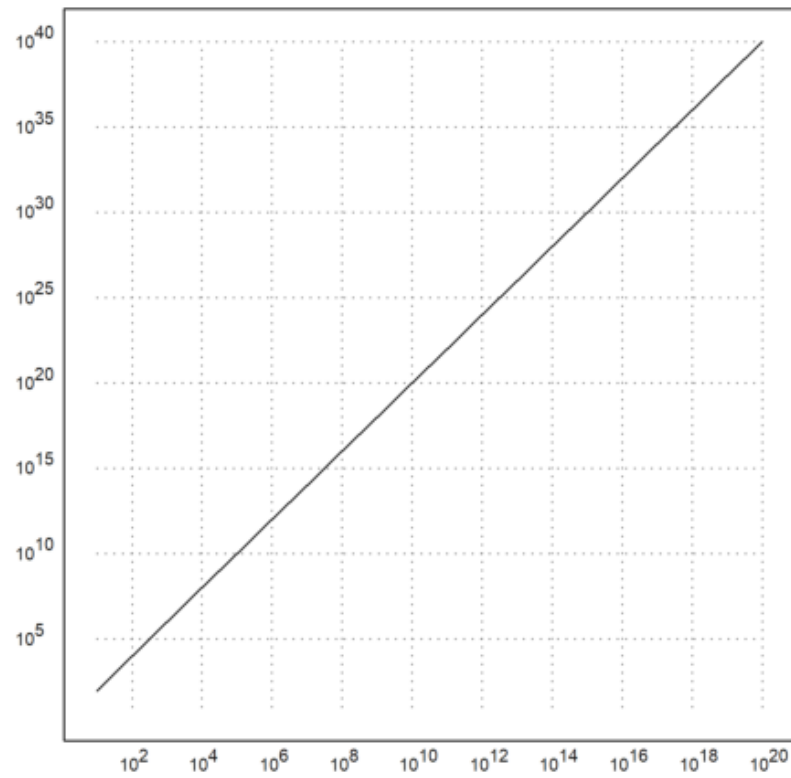



```
>plot2d("log(x*(2+sin(x/100)))",10,1000,logplot=3):
```



Ini juga bekerja pada grafik data.

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



Latihan Soal

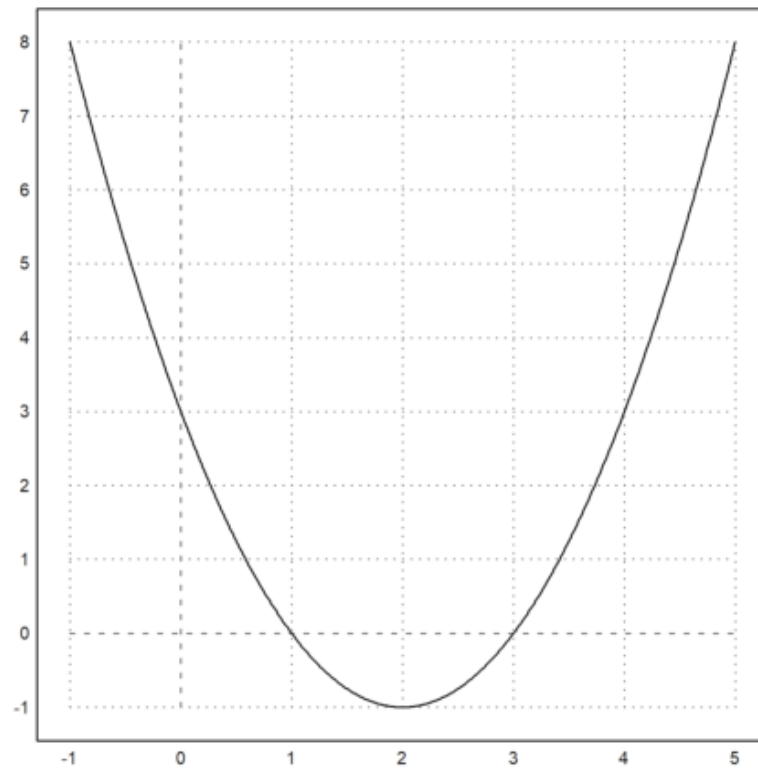
Nama : Muhammad Lutfi Ramadhan
Kelas : Matematika B 2023
NIM : 23030630021

1. Gambarkan grafik fungsi berikut!

$$f(x) = x^2 - 4x + 3$$

Penyelesaian:

```
>plot2d("x^2-4*x+3",-1,5):
```

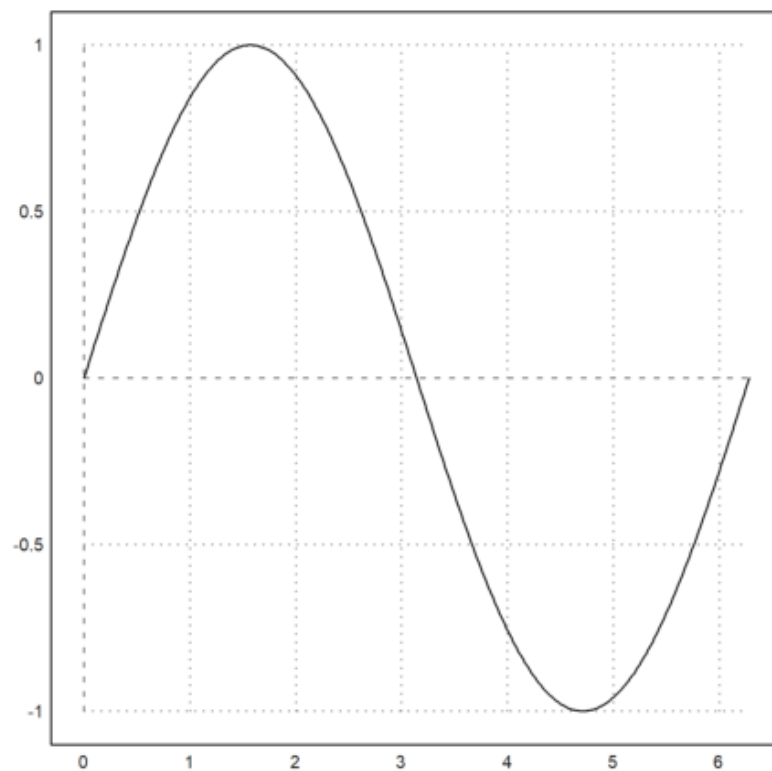


2. Buatkan grafik fungsi berikut pada interval 0 sampai 2pi!

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

Penyelesaian:

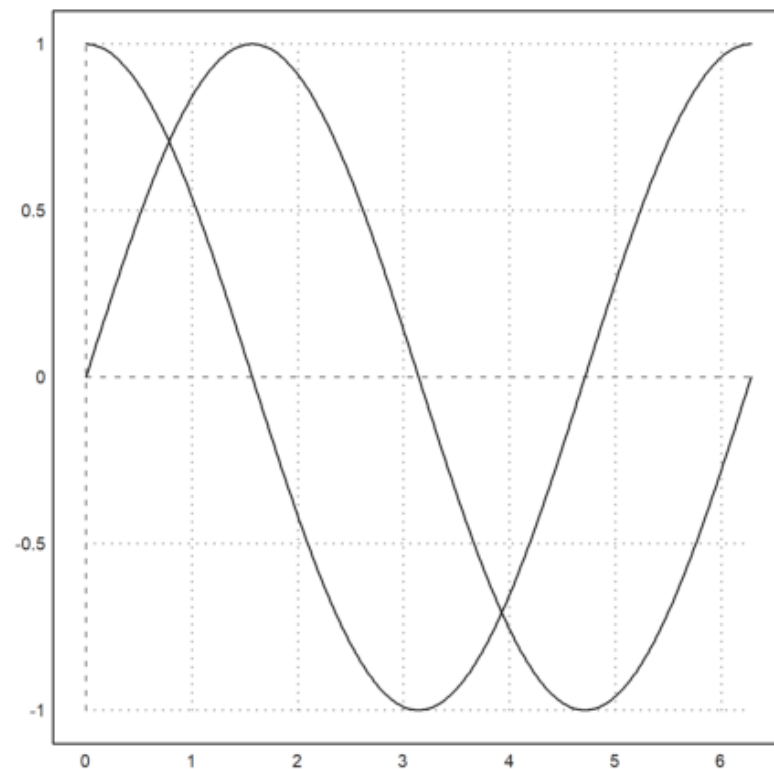
```
>plot2d("sin(x)", 0, 2*pi):
```



3. Gambarkan grafik fungsi $f(x)=\sin(x)$ dan $g(x)=\cos(x)$ secara bersamaan pada interval 0 sampai 2pi!

Penyelesaian

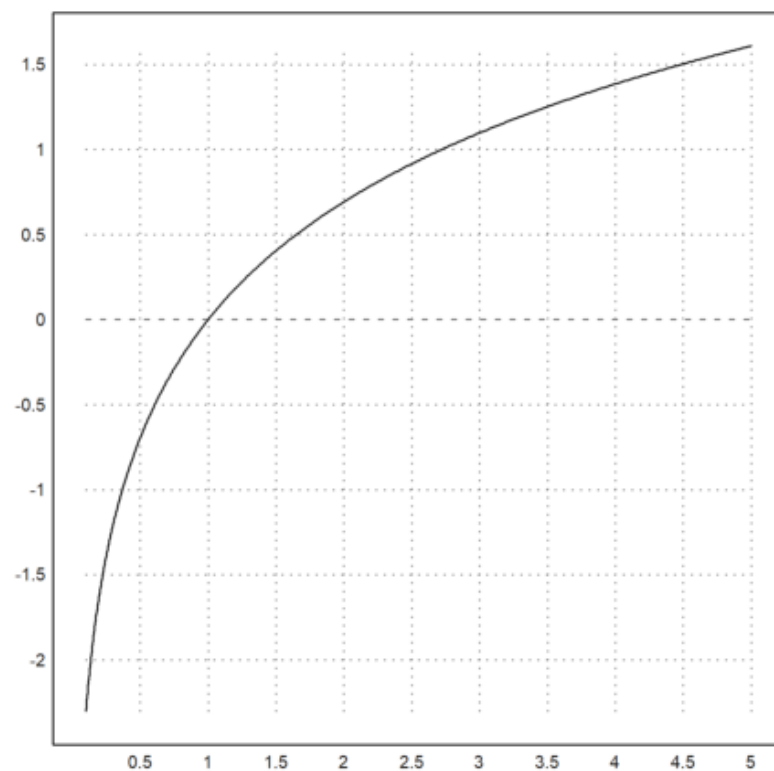
```
>plot2d(["sin(x)", "cos(x)"], 0, 2*pi):
```



4. Gambarkan grafik fungsi $f(x)=\log(x)$ pada interval $[0.1,5]$!

Penyelesaian:

```
>plot2d("log(x)", 0.1, 5):
```

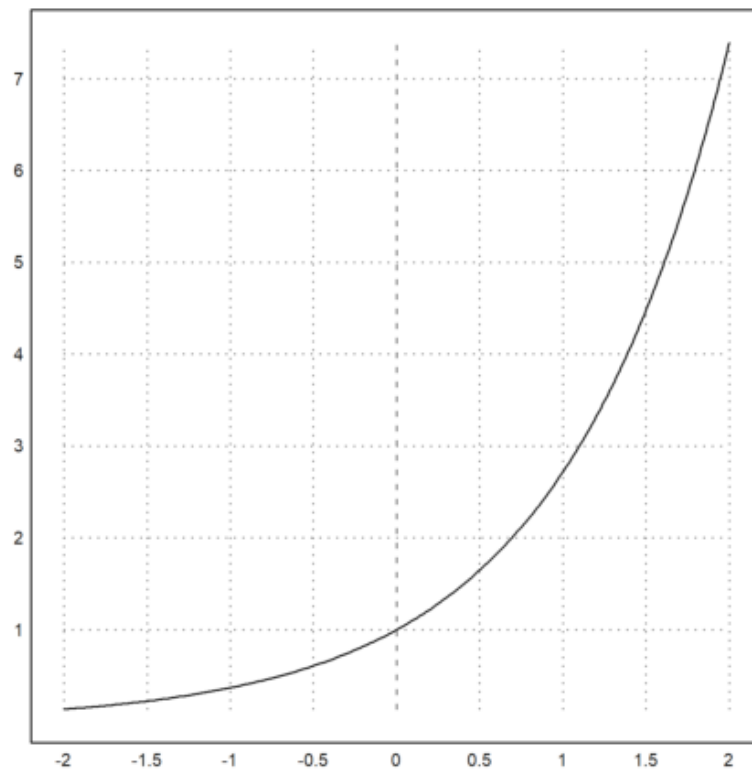


5. Gambarkan fungsi eksponen berikut pada interval $[-2,2]$!

$$f(x) = e^x$$

Penyelesaian:

```
>plot2d("exp(x)", -2, 2):
```



>

Rujukan Lengkap Fungsi plot2d()

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

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

Parameters

x,y : equations, functions or data vectors
a,b,c,d : Plot area (default a=-2,b=2)
r : if r is set, then a=cx-r, b=cx+r, c=cy-r, d=cy+r
r can be a vector [rx,ry] or a vector [rx1,rx2,ry1,ry2].
xmin,xmax : range of the parameter for curves
auto : Determine y-range automatically (default)
square : if true, try to keep square x-y-ranges
n : number of intervals (default is adaptive)
grid : 0 = no grid and labels,
1 = axis only,
2 = normal grid (see below for the number of grid lines)
3 = inside axis
4 = no grid
5 = full grid including margin
6 = ticks at the frame
7 = axis only
8 = axis only, sub-ticks
frame : 0 = no frame

framecolor: color of the frame and the grid
margin : number between 0 and 0.4 for the margin around the plot
color : Color of curves. If this is a vector of colors, it will be used for each row of a matrix of plots. In the case of point plots, it should be a column vector. If a row vector or a full matrix of colors is used for point plots, it will be used for each data point.

thickness : line thickness for curves

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

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

For points use

"[]", "<>", ".", "...", "...",

"*", "+", "|", "-", "o"

"[]#", "<>#", "o#" (filled shapes)

"[]w", "<>w", "ow" (non-transparent)

For lines use

"-", "--", "-.", ".", "-.", "-.", "->"

For filled polygons or bar plots use

"#", "#O", "O", "I", "V", "V",

"+", "|", "-", "t"

points : plot single points instead of line segments

addpoints : if true, plots line segments and points

add : add the plot to the existing plot

user : enable user interaction for functions

delta : step size for user interaction

bar : bar plot (x are the interval bounds, y the interval values)

histogram : plots the frequencies of x in n subintervals

distribution=n : plots the distribution of x with n subintervals

even : use inter values for automatic histograms.

steps : plots the function as a step function (steps=1,2)

adaptive : use adaptive plots (n is the minimal number of steps)

level : plot level lines of an implicit function of two variables

outline : draws boundary of level ranges.

If the level value is a 2xn matrix, ranges of levels will be drawn in the color using the given fill style. If outline is true, it will be drawn in the contour color. Using this feature, regions of $f(x,y)$ between limits can be marked.

hue : add hue color to the level plot to indicate the function value

contour : Use level plot with automatic levels

nc : number of automatic level lines

title : plot title (default "")

xl, yl : labels for the x- and y-axis

smaller : if >0, there will be more space to the left for labels.

vertical :

Turns vertical labels on or off. This changes the global variable

verticallabels locally for one plot. The value 1 sets only vertical

text, the value 2 uses vertical numerical labels on the y axis.

filled : fill the plot of a curve

fillcolor : fill color for bar and filled curves

outline : boundary for filled polygons

logplot : set logarithmic plots

1 = logplot in y,

2 = logplot in xy,

3 = logplot in x

own :

A string, which points to an own plot routine. With >user, you get

the same user interaction as in plot2d. The range will be set

before each call to your function.

maps : map expressions (0 is faster), functions are always mapped.

contourcolor : color of contour lines

contourwidth : width of contour lines

clipping : toggles the clipping (default is true)

title :

This can be used to describe the plot. The title will appear above

the plot. Moreover, a label for the x and y axis can be added with

xl="string" or yl="string". Other labels can be added with the

functions label() or labelbox(). The title can be a unicode

string or an image of a Latex formula.

cgrid :

Determines the number of grid lines for plots of complex grids.

Should be a divisor of the the matrix size minus 1 (number of subintervals). cgrid can be a vector [cx,cy].

Overview

The function can plot

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

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

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