# TI_EMT_Plot2D_Prima Rosalina_24030130035

Nama : Prima Rosalina
NIM : 24030130035
Kelas : Pendidikan Matemeatika C 2024

## Menggambar Grafik 2D dengan EMT

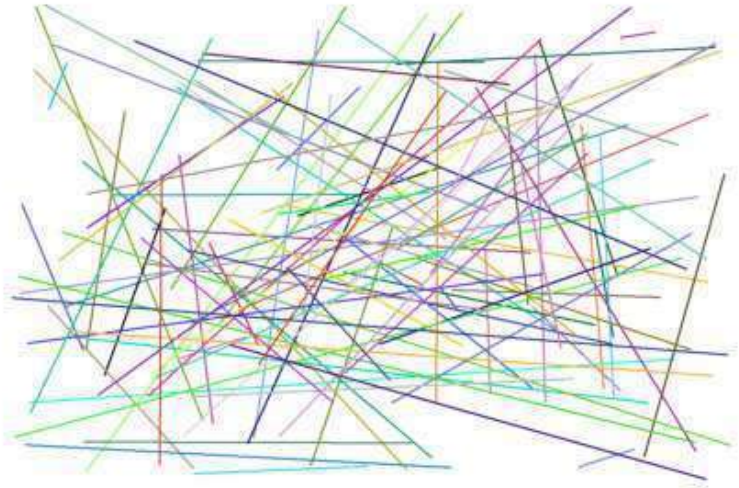
Notebook ini menjelaskan tentang cara menggambar berbagaikurva dan grafik 2D dengan software EMT. EMT menyediakan fungsi plot2d() untuk menggambar berbagai kurva dan grafik dua dimensi (2D).

## Plot Dasar

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

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

```
>clg; // menghapus layar
>window(0,0,1024,1024); // menggunakan seluruh jendela
>setplot(0,1,0,1); // mengatur koordinat plot
>hold on; // memulai mode timpa
>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 timpa
>insimg; // menyisipkan ke notebook
```



```
>reset;
```

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

Untuk menghapus semua yang telah kita lakukan, kita gunakan 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.

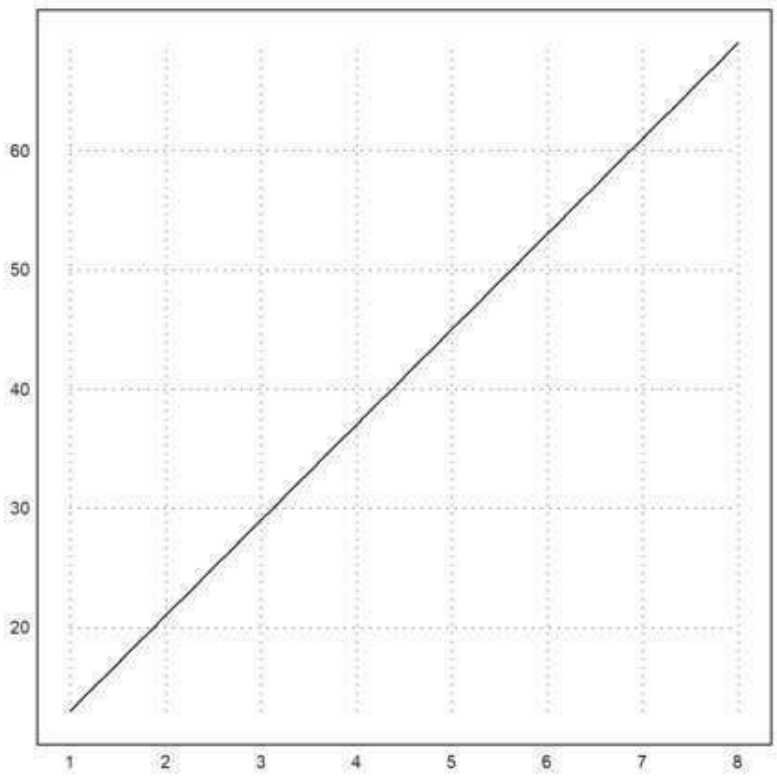
Untuk contoh lain, kita menggambar sebuah plot sebagai inset di dalam plot lain. Hal ini dilakukan dengan mendefinisikan jendela plot yang lebih kecil. Perlu diperhatikan bahwa jendela ini tidak menyediakan ruang untuk label sumbu di luar jendela plot. Kita harus menambahkan sedikit margin sesuai kebutuhan. Perlu dicatat juga bahwa kita menyimpan dan mengembalikan jendela penuh, serta menahan plot saat ini ketika kita membuat plot inset.

## Langkah-langkah membuat plot 2D

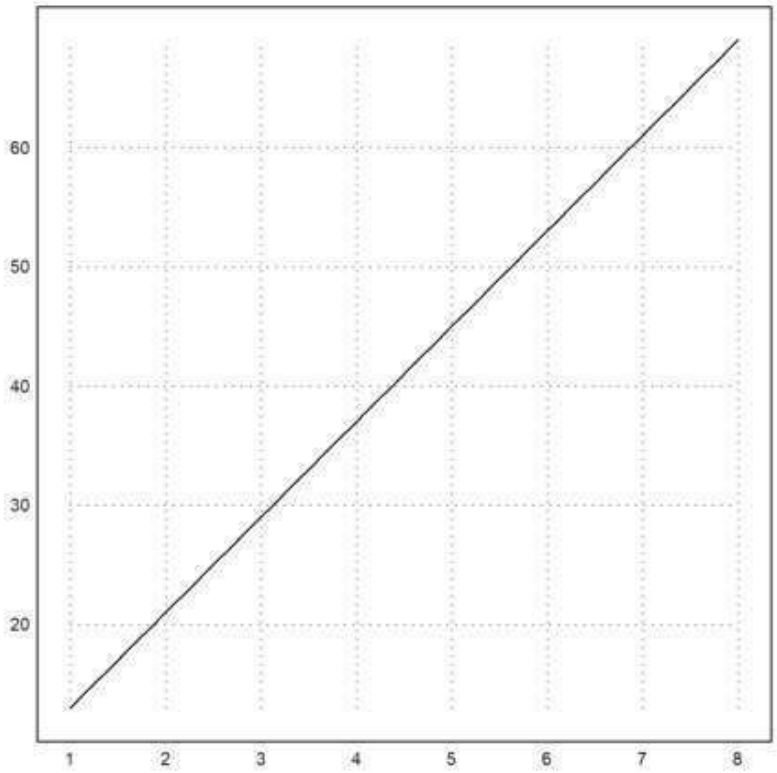
1. Menentukan domain (rentang nilai untuk sumbu x)
2. Mendefinisikan fungsi y
3. Membuat plot dengan memanggil fungsi plot2d()
4. Menambahkan kustomisasi (opsional). Contoh: title(), xlabel(), ylabel(), dan legend().
5. Menampilkan plot

Berikut sintaks untuk grafik fungsi satu variabel:
plot2d("ekspresi_fungsi",rentang_x_min, rentang_x_max)

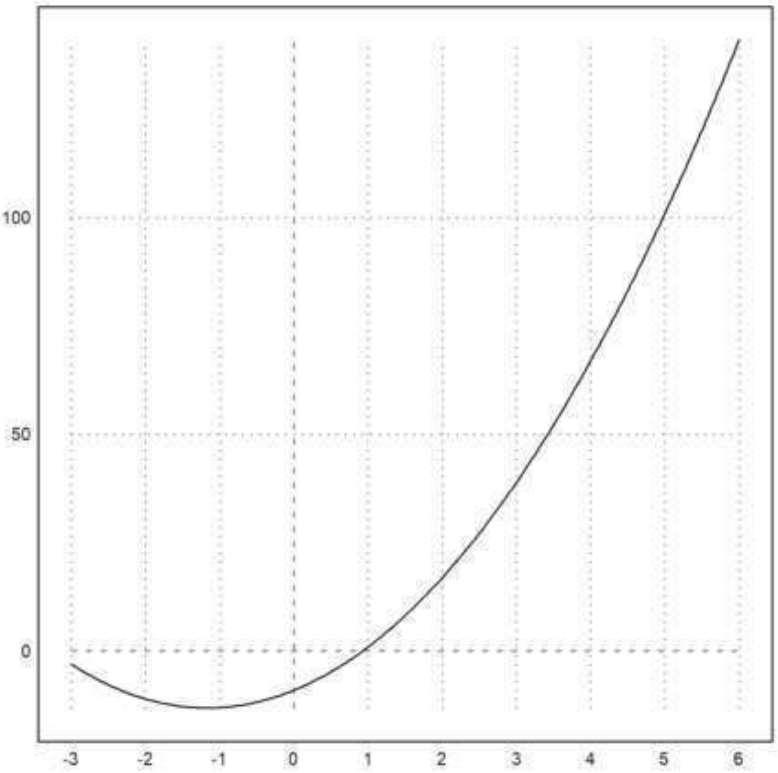
```
>//Contoh pada fungsi linear
>plot2d("8x+5",1,8):
```



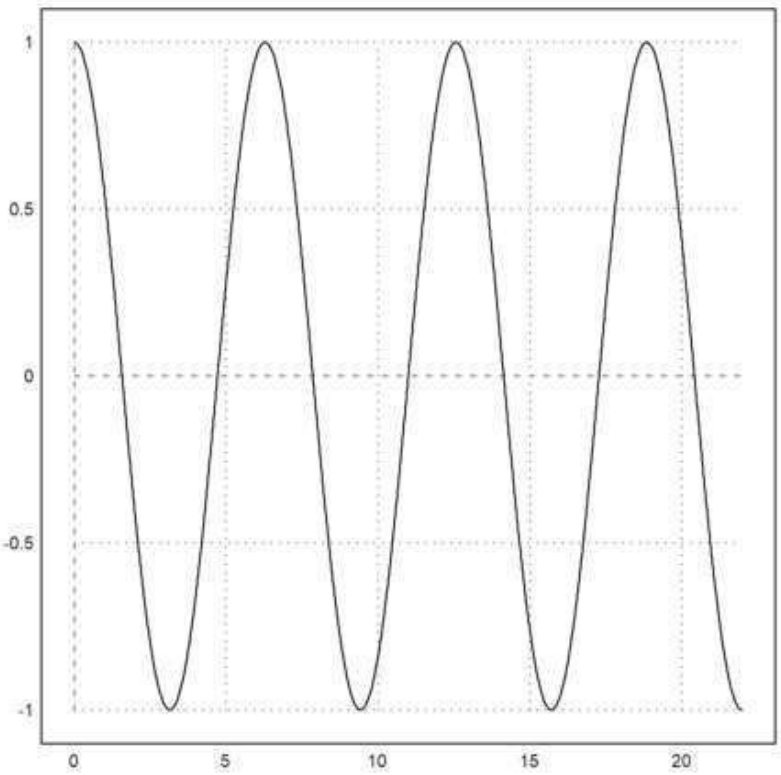

```
>insimg // fungsinya untuk menampilkan grafik seperti tanda titik dua (:)
```



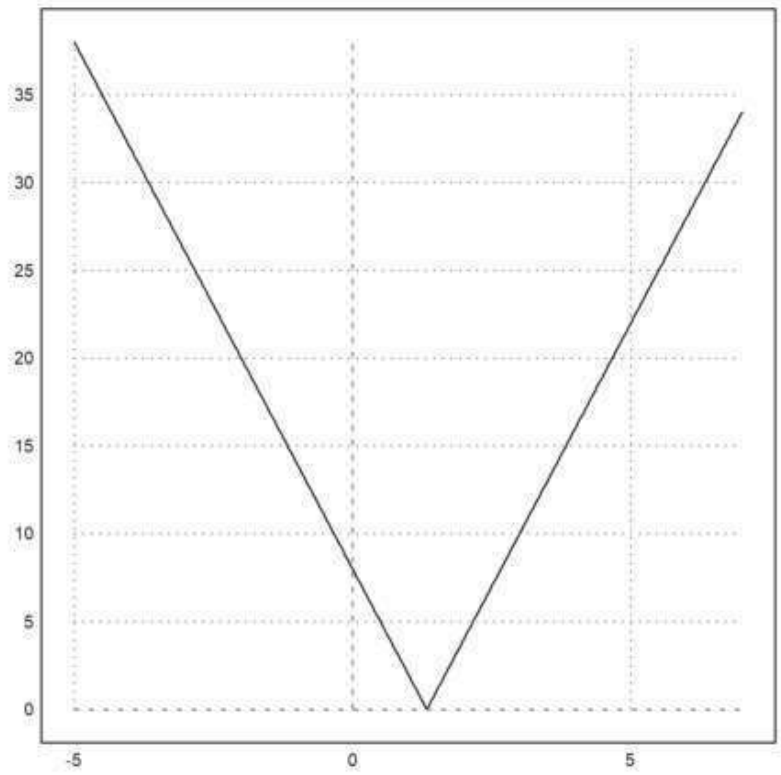

```
>//Contoh pada fungsi kuadrat
>plot2d("3*x^2+7*x-9",-3,6):
```



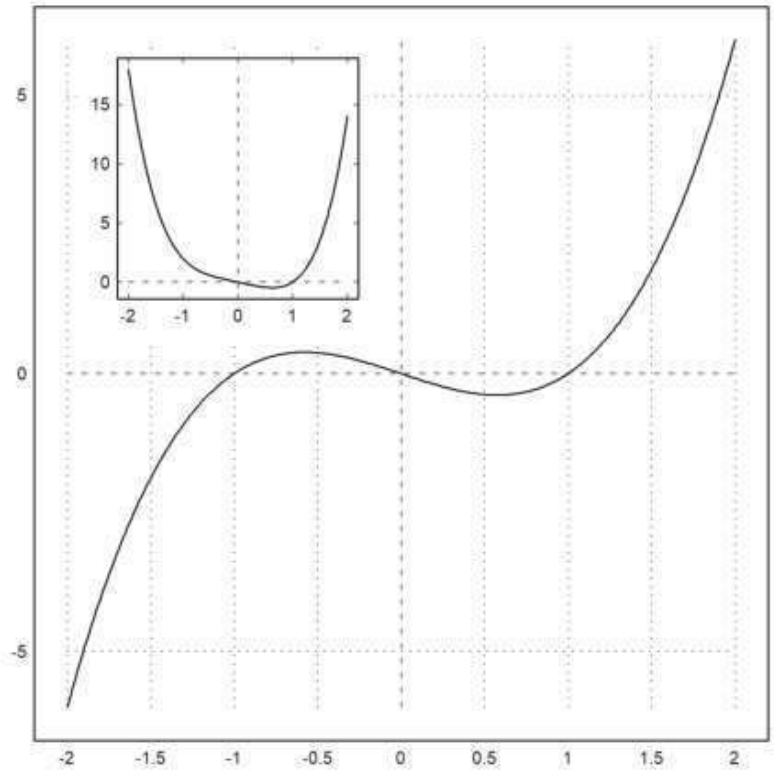

```
>plot2d("cos(x)",0,7*pi): //Contoh pada fungsi trigonometri
```




```
>plot2d("abs(6x-8)",-5,7):
```




```
>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);
```

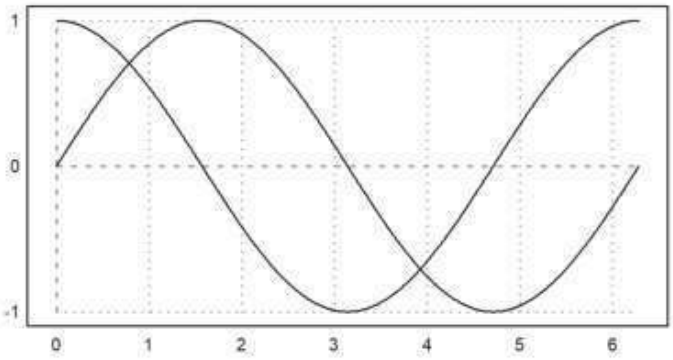
Sebuah plot dengan beberapa gambar dapat dibuat dengan cara yang sama. Untuk ini tersedia fungsi utilitas figure().

## Aspek Plot

Plot bawaan menggunakan jendela plot berbentuk persegi. Anda dapat mengubahnya dengan fungsi aspect(). Jangan lupa untuk mereset aspek tersebut nanti. Anda juga dapat mengubah bawaan ini di menu dengan "Set Aspect" ke rasio aspek tertentu atau ke ukuran saat ini dari jendela grafik.

Namun, Anda juga bisa mengubahnya hanya untuk satu plot. Untuk hal ini, ukuran area plot saat ini diubah, dan jendela diatur sehingga label memiliki ruang yang cukup.

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




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

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

## 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 maupun data.

Dimungkinkan juga untuk membuat plot di Maxima menggunakan Gnuplot atau di Python menggunakan Matplotlib.

Euler dapat memplot 2D untuk:

- ekspresi,
- fungsi, variabel, atau kurva terparametrisasi,
- vektor nilai x-y,
- kumpulan titik di bidang,
- kurva implisit dengan level atau daerah level,
- fungsi kompleks.

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

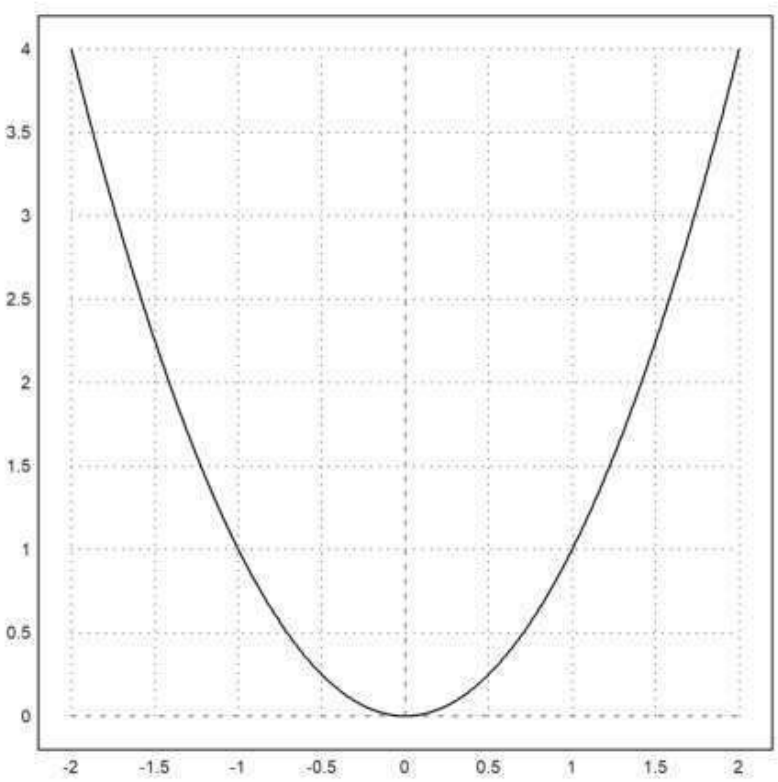
## Plot Ekspresi atau Variabel

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

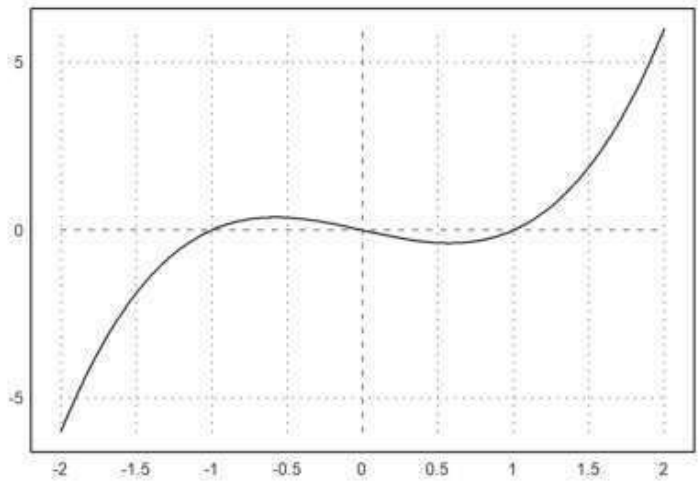
Berikut contoh paling dasar, yang menggunakan rentang bawaan dan mengatur rentang y yang sesuai agar grafik fungsi dapat ditampilkan dengan baik.

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

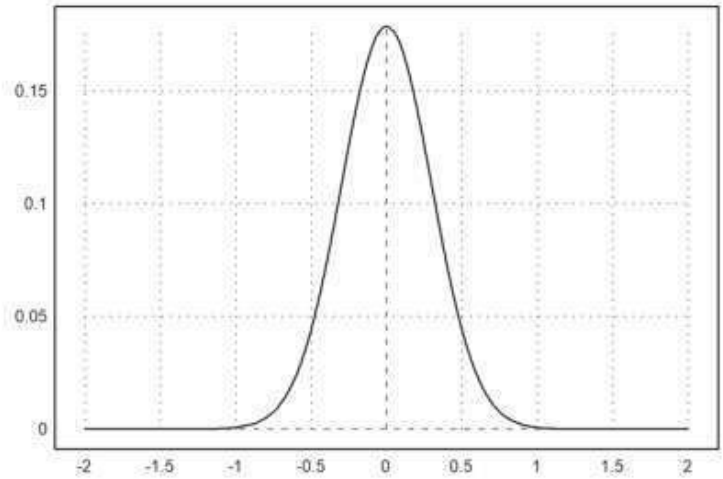
```
>plot2d("x^2"):
```




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




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

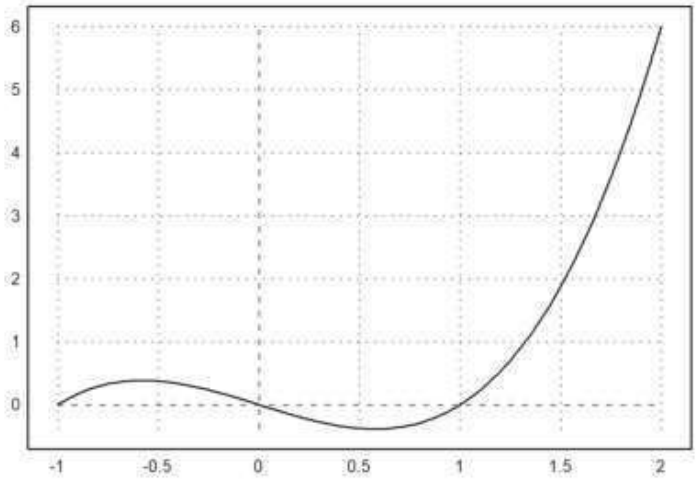



Dari beberapa contoh sebelumnya Anda dapat melihat bahwa secara bawaan gambar plot menggunakan sumbu X dengan rentang nilai dari -2 sampai dengan 2. Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan batas nilai X (dan Y) di belakang ekspresi yang digambar.

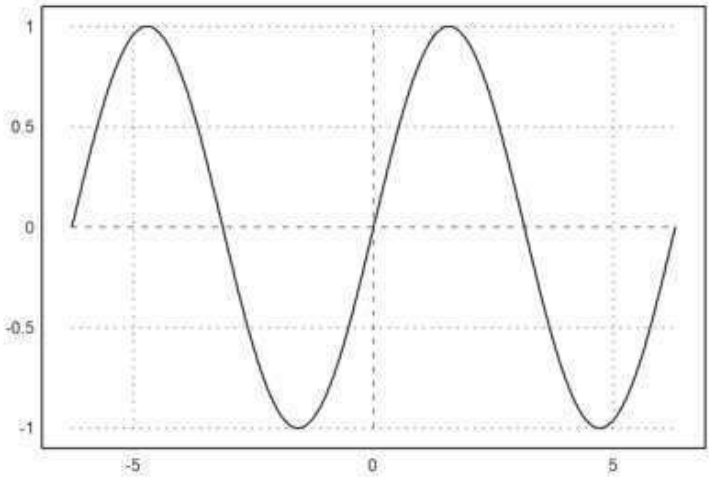
Rentang plot diatur dengan parameter yang ditentukan sebagai berikut:

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

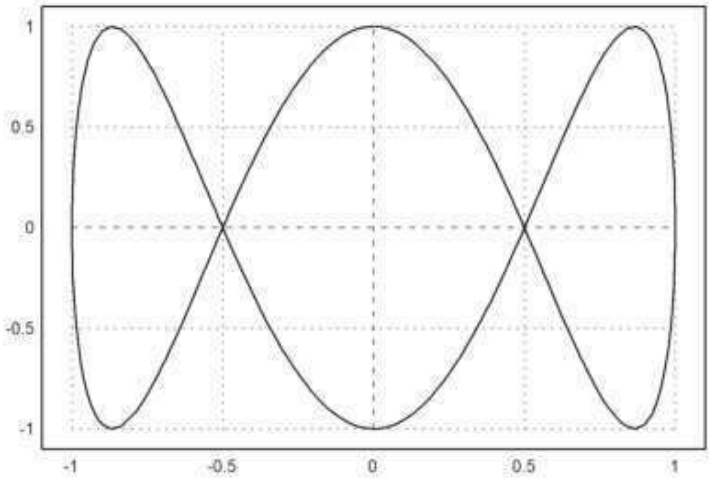
```
>plot2d("x^3-x",-1,2):
```




```
>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 dari tanda titik dua adalah perintah insimg(lines), yang menyisipkan plot dengan menempati sejumlah baris teks tertentu.

Dalam opsi, plot dapat diatur untuk tampil

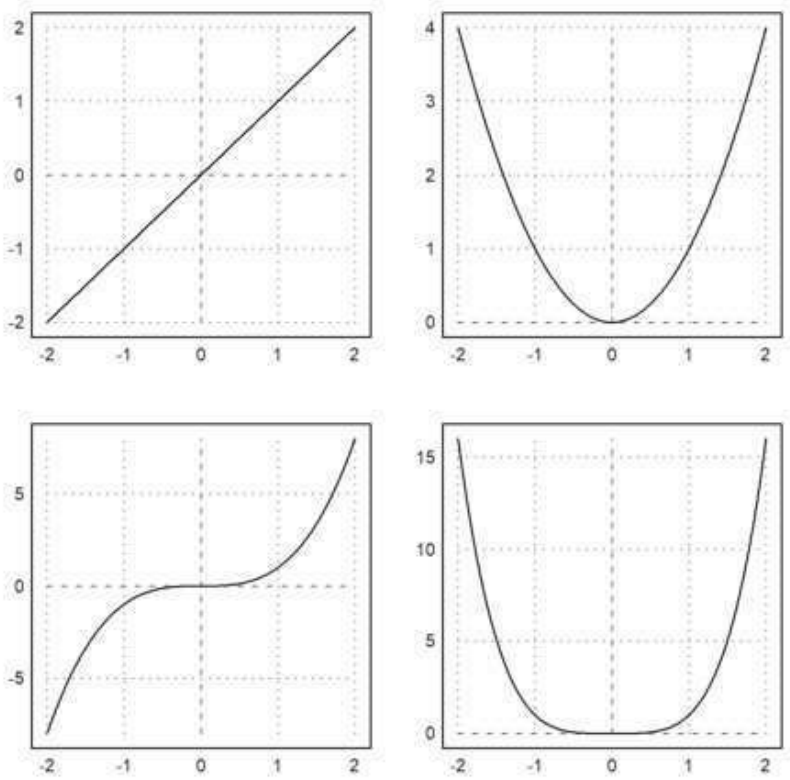
- di jendela terpisah yang dapat diubah ukurannya,
- di jendela notebook.

Lebih banyak gaya dapat dicapai dengan perintah plot khusus.

Dalam kasus apa pun, tekan tombol Tab untuk melihat plot jika plot tertutup.

Untuk membagi jendela menjadi beberapa plot, gunakan perintah figure(). Pada contoh, kita memplot x^1 hingga x^4 ke dalam 4 bagian jendela. figure(0) mengembalikan jendela ke bawaan.

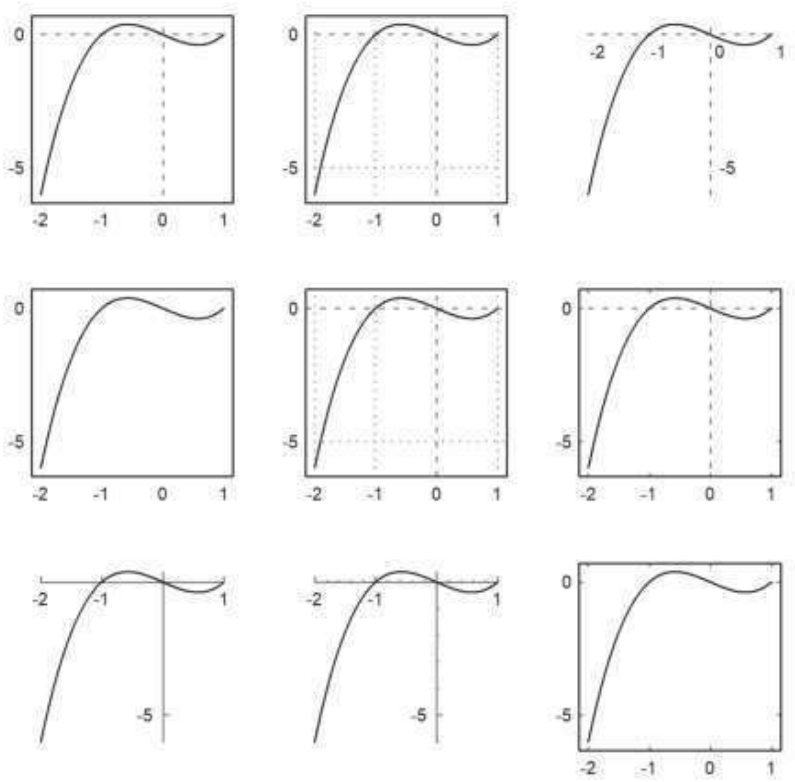
```
>reset;
>figure(2,2); ...
 for n=1 to 4; figure(n); plot2d("x^"+n); end; ...
 figure(0):
```




Dalam plot2d(), tersedia gaya alternatif dengan grid=x. Sebagai gambaran, berbagai gaya grid ditampilkan dalam satu gambar (lihat penggunaan perintah figure() di bawah).

Gaya grid=0 tidak disertakan. Gaya ini menampilkan tanpa grid dan tanpa bingkai.

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

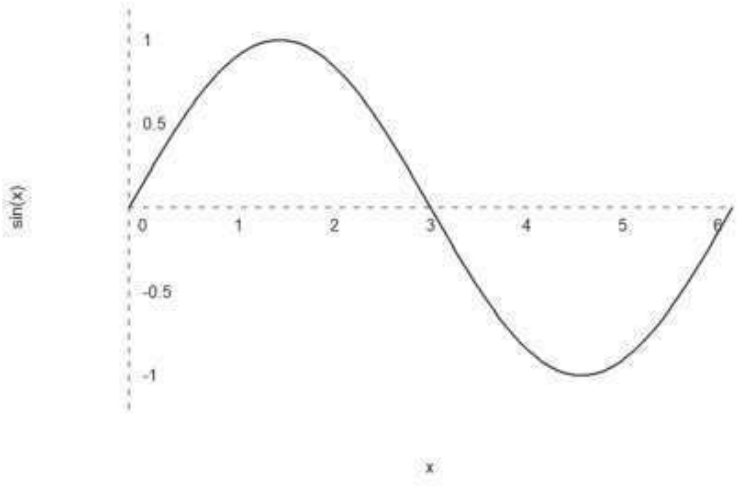


Jika argumen untuk plot2d() berupa sebuah ekspresi yang diikuti oleh empat angka, maka angka-angka tersebut adalah rentang x dan y untuk plot.

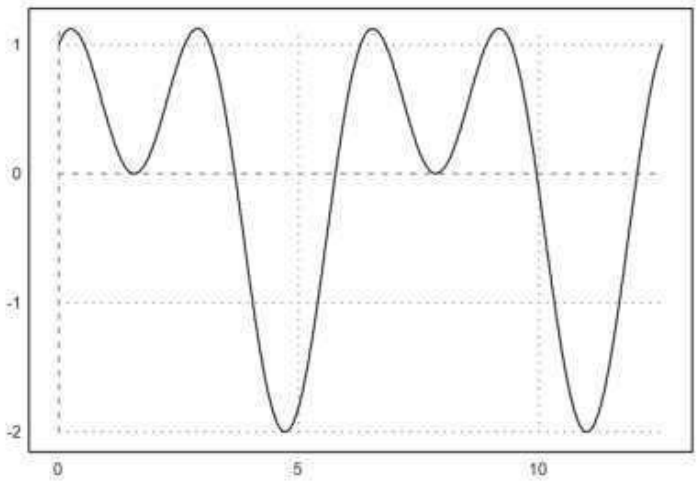
Sebagai alternatif, a, b, c, d dapat ditentukan sebagai parameter yang diberikan dengan bentuk a=... dan seterusnya.

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

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




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

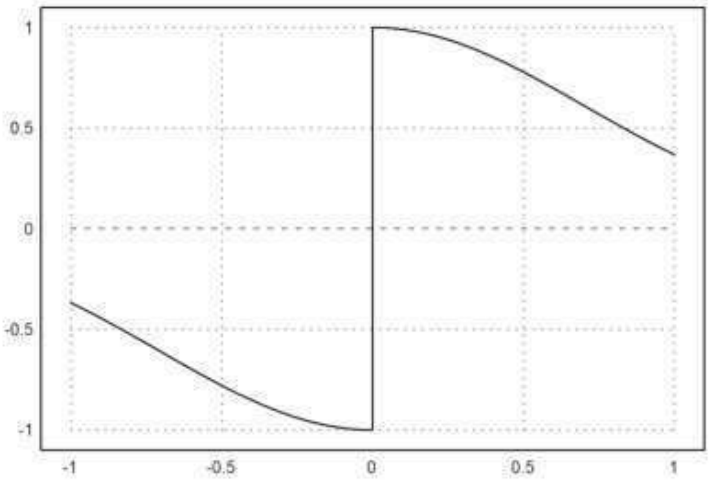



Gambar yang dihasilkan dengan menyisipkan plot ke dalam jendela teks akan disimpan di direktori yang sama dengan notebook, secara bawaan dalam subdirektori bernama "images". Gambar-gambar ini juga digunakan oleh fitur ekspor HTML.

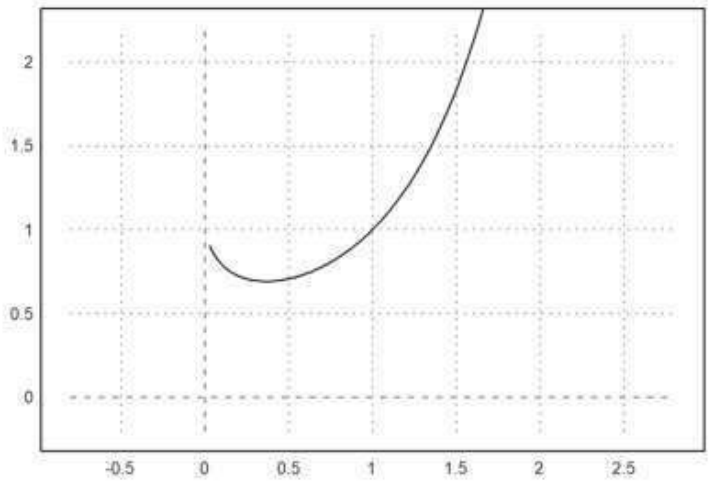
Anda dapat dengan mudah menandai gambar apa pun dan menyalinnya ke clipboard dengan Ctrl-C. Tentu saja, Anda juga dapat mengekspor grafik saat ini dengan fungsi yang ada di menu File.

Fungsi atau ekspresi dalam plot2d dievaluasi secara adaptif. Untuk meningkatkan kecepatan, Anda dapat mematikan plot adaptif dengan <adaptive dan menentukan jumlah subinterval dengan n=.... Hal ini biasanya hanya diperlukan dalam kasus-kasus tertentu saja.

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



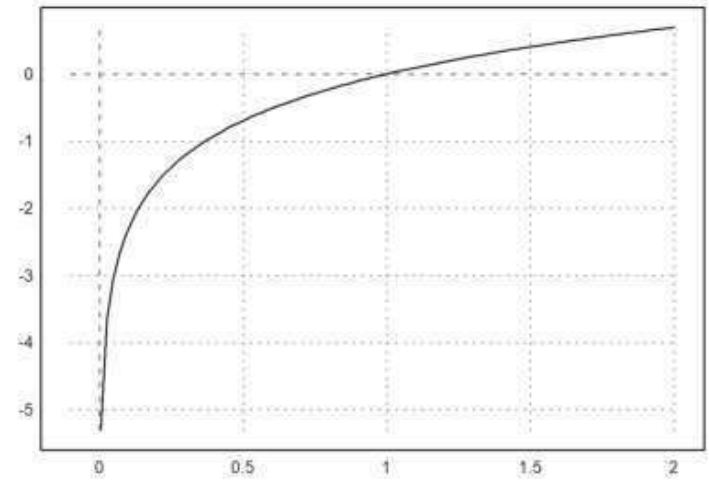

```
>plot2d("x^x",r=1.2,cx=1,cy=1):
```




Perlu dicatat bahwa x^x tidak terdefinisi untuk x = 0. Fungsi plot2d akan menangani kesalahan ini, dan mulai memplot segera setelah fungsi tersebut terdefinisi.

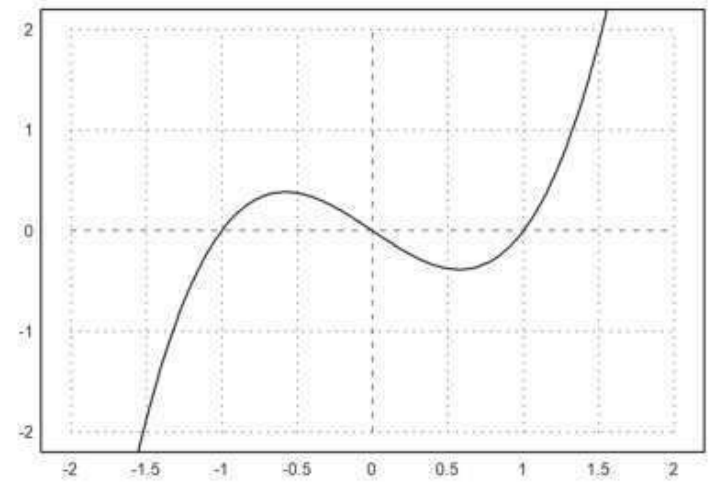
Hal ini berlaku untuk semua fungsi yang menghasilkan NAN di luar daerah definisinya.

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

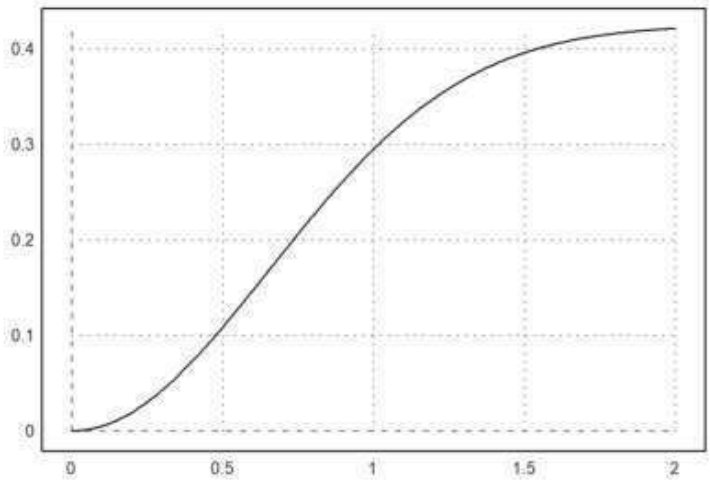



Parameter square=true (atau >square) secara otomatis memilih rentang y sehingga hasilnya berupa jendela plot berbentuk persegi. Perlu dicatat bahwa secara bawaan, Euler menggunakan ruang berbentuk persegi di dalam jendela plot.

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

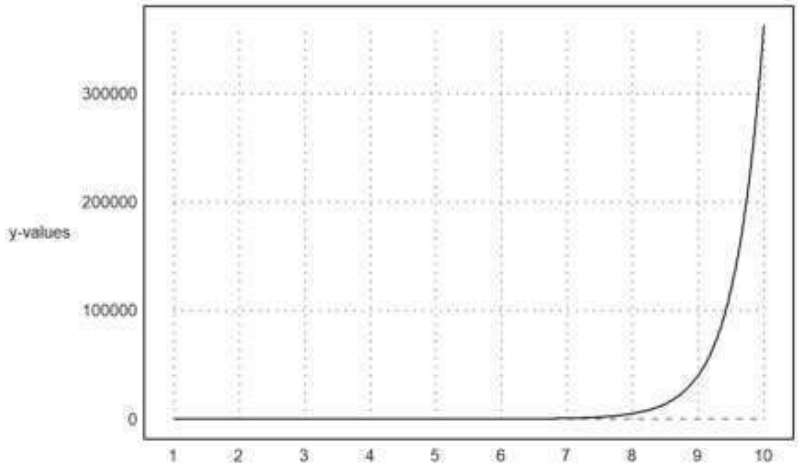


```
>plot2d(''integrate("sin(x)*exp(-x^2)",0,x)'',0,2): // plot integral
```



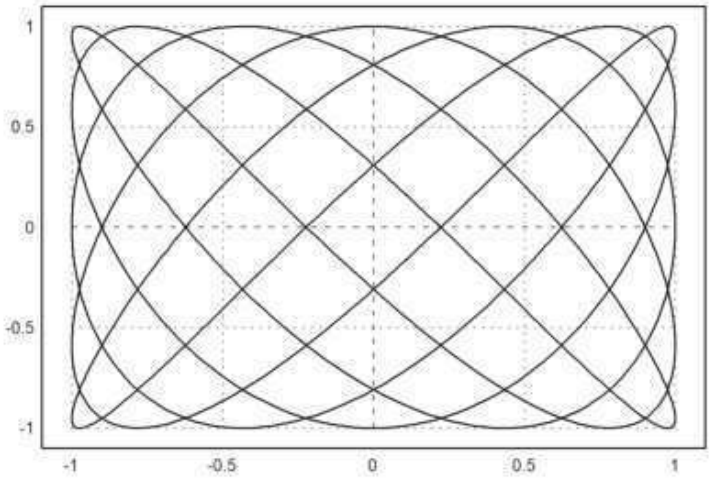

Jika Anda membutuhkan lebih banyak ruang untuk label sumbu-y, panggil shrinkwindow() dengan parameter smaller, atau atur nilai positif untuk "smaller" di plot2d().

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

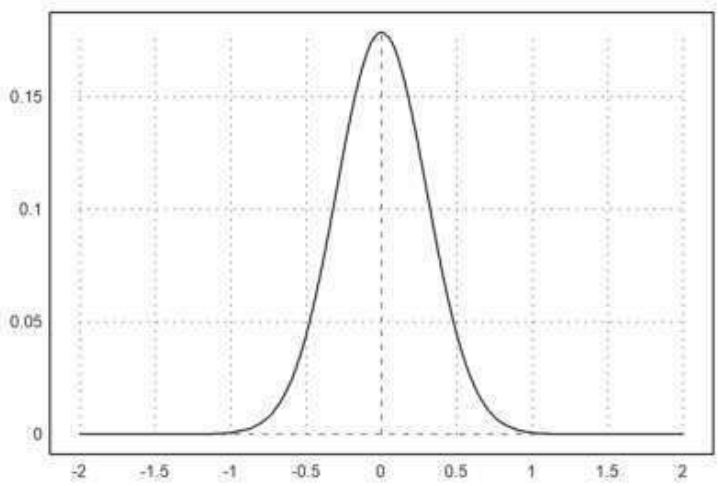



Ekspresi simbolik juga dapat digunakan, karena disimpan sebagai ekspresi string sederhana.

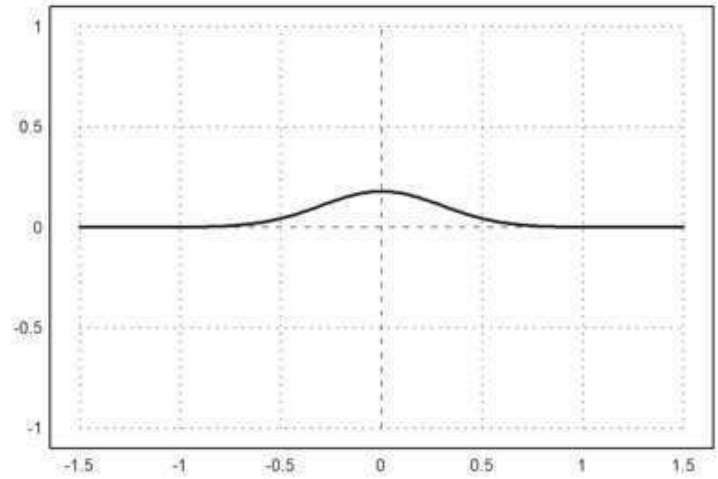
```
>x=linspace(0,2pi,1000); plot2d(sin(5x),cos(7x)):
```



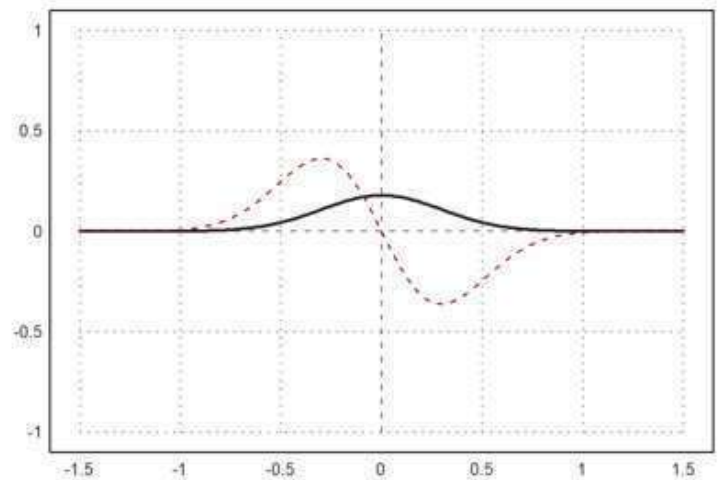
```
>a:=5.6; expr &= exp(-a*x^2)/a; // mendefinisikan ekspresi
>plot2d(expr,-2,2): // plot dari -2 ke 2
```



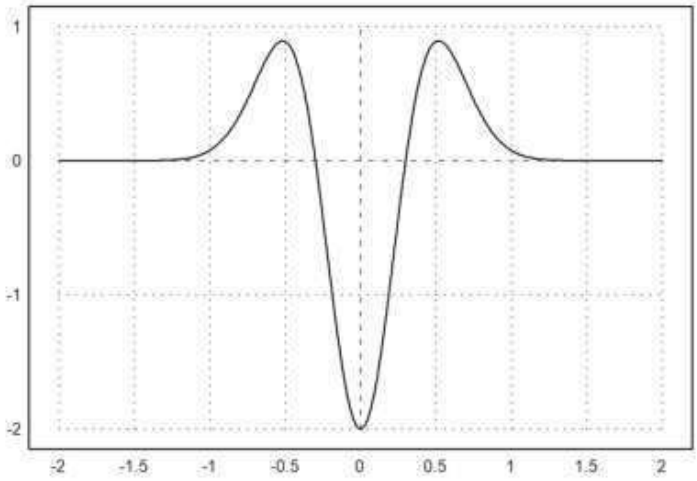

```
>plot2d(expr,r=1,thickness=2): // menggambar dalam bentuk persegi di sekitar (0,0)
```



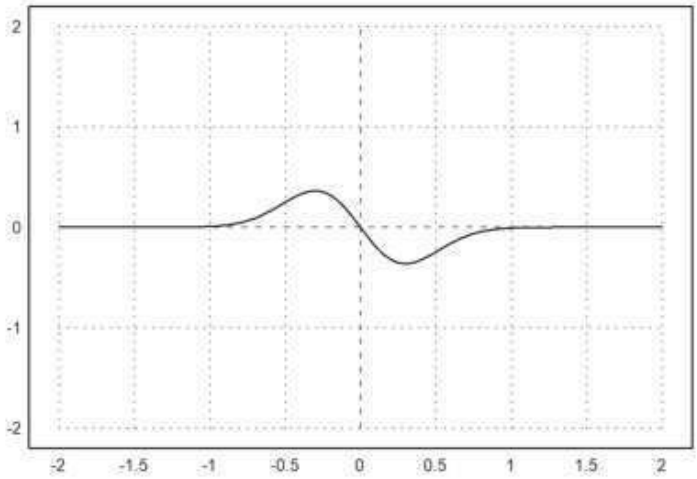

```
>plot2d(&diff(expr,x),>add,style="--",color=red): // menambahkan plot lain
```



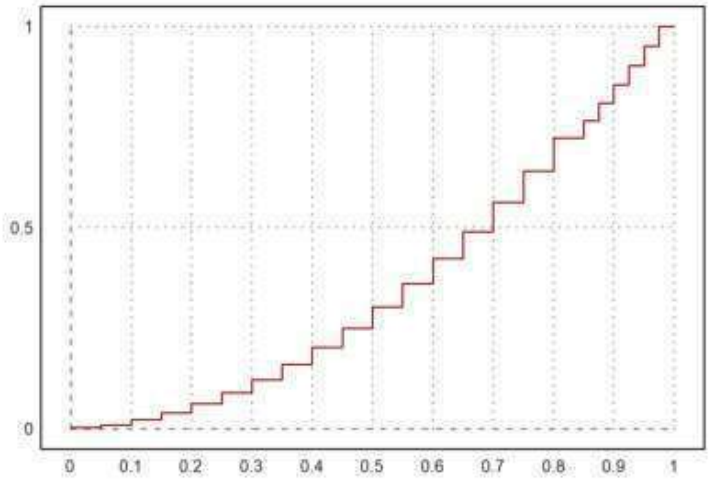

```
>plot2d(&diff(expr,x,2),a=-2,b=2,c=-2,d=1): // menggambar dalam persegi panjang
```



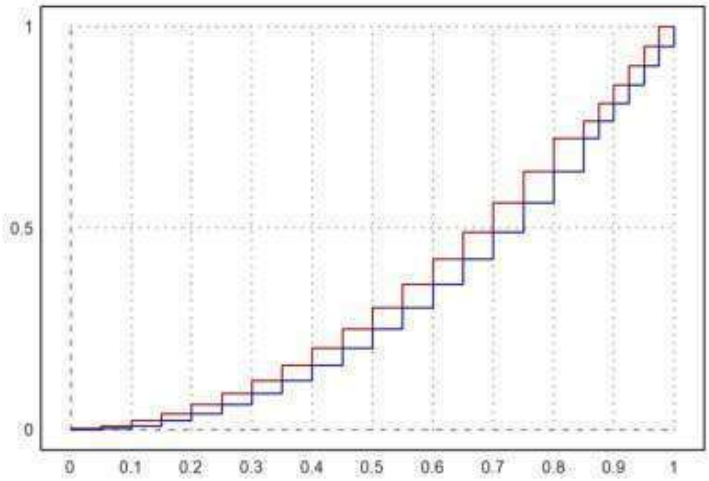

```
>plot2d(&diff(expr,x),a=-2,b=2,>square): // menjaga plot tetap persegi
```




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




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

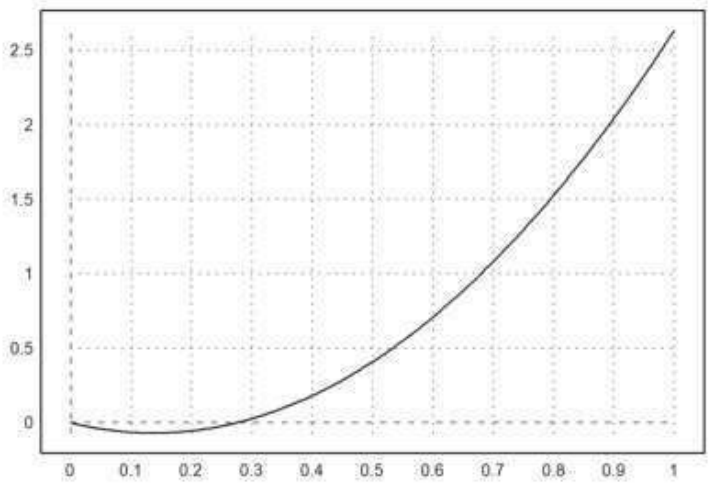



## Fungsi dengan satu parameter

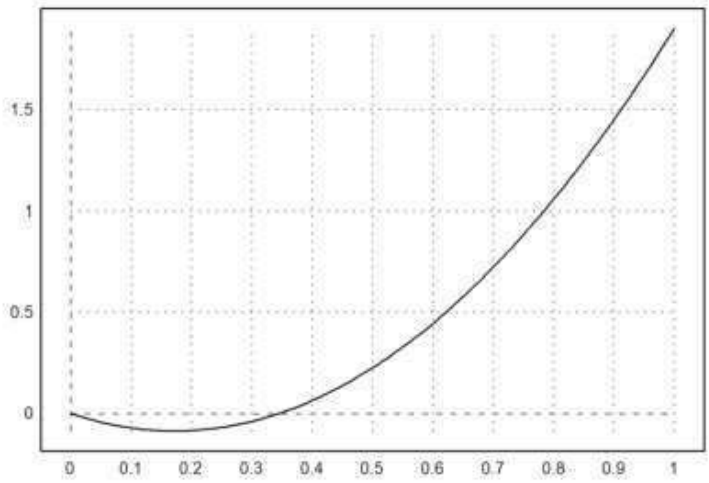
Fungsi plotting terpenting untuk plot bidang adalah plot2d(). Fungsi ini diimplementasikan dalam bahasa Euler di dalam file "plot.e", yang dimuat pada saat program dimulai.

Berikut beberapa contoh penggunaan fungsi. Seperti biasa di EMT, fungsi-fungsi yang bekerja untuk fungsi lain atau ekspresi dapat diberi parameter tambahan (selain x) yang bukan merupakan variabel global ke dalam fungsi tersebut dengan parameter titik koma atau dengan kumpulan pemanggilan.

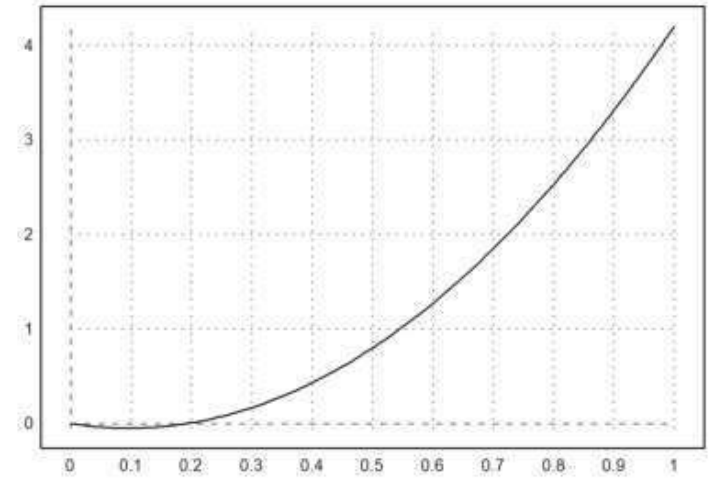
```
>function f(x,a) := x^2/a+a*x^2-x; // mendefinisikan fungsi
>a=0.3; plot2d("f",0,1;a): // plot dengan a=0.3
```



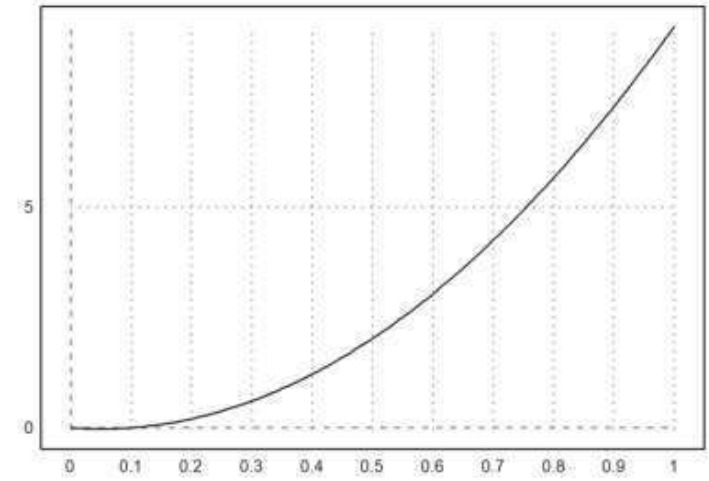

```
>plot2d("f",0,1;0.4): // plot dengan a=0.4
```



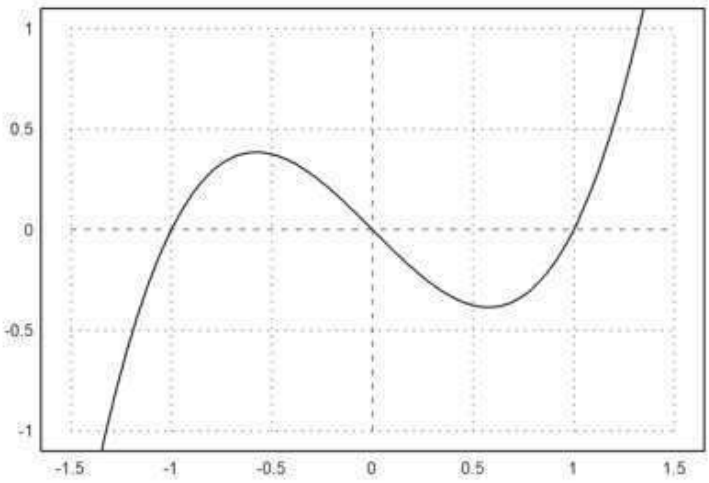

```
>plot2d({{"f",0.2}},0,1): // plot dengan a=0.2
```




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




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




Berikut ringkasan fungsi yang diterima:

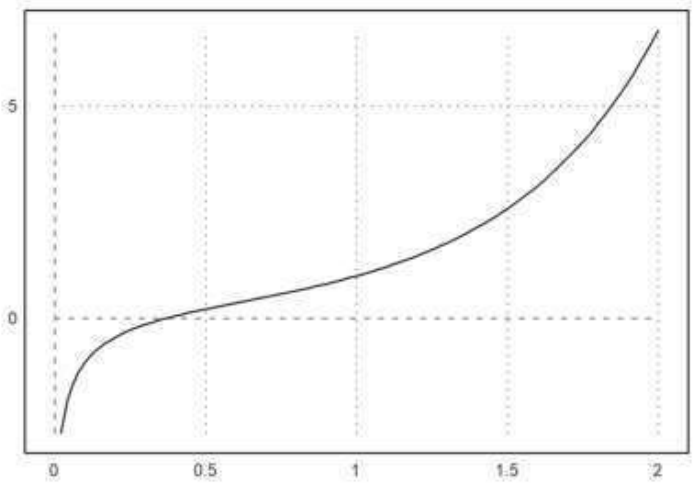
- ekspresi atau ekspresi simbolik dalam x
- fungsi atau fungsi simbolik dengan nama seperti "f"
- fungsi simbolik cukup dengan nama f

Fungsi plot2d() juga menerima fungsi simbolik. Untuk fungsi simbolik, cukup dengan menggunakan namanya saja.

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

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

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

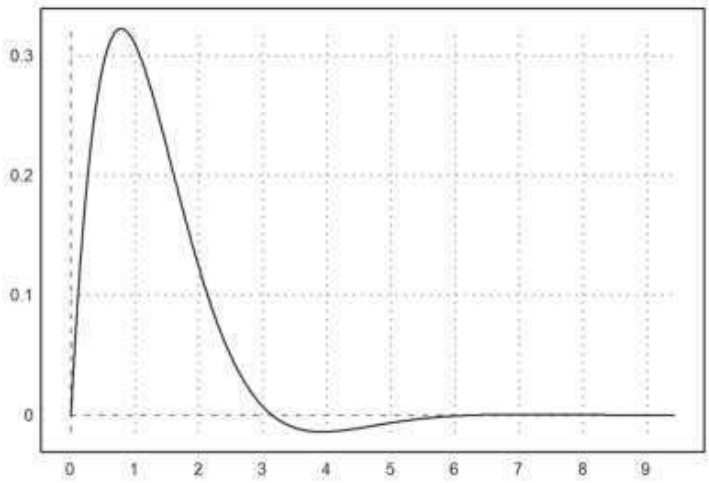



Tentu saja, untuk ekspresi atau ekspresi simbolik, nama variabel saja sudah cukup untuk memplotnya.

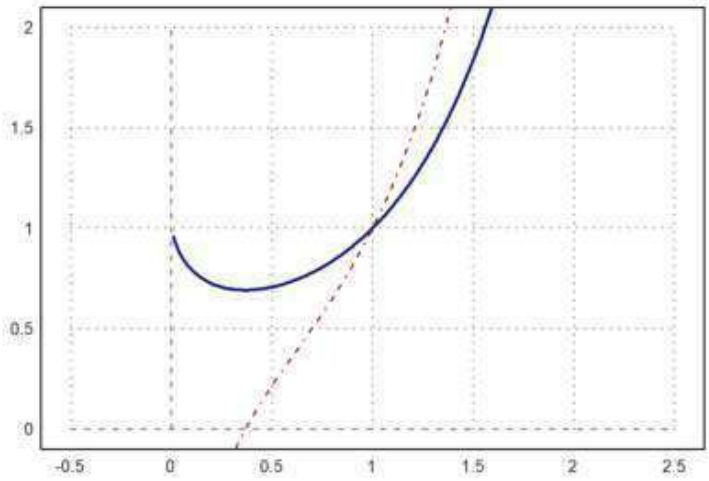
```
>expr &= sin(x)*exp(-x)
```

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

```
>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="-.-"):
```



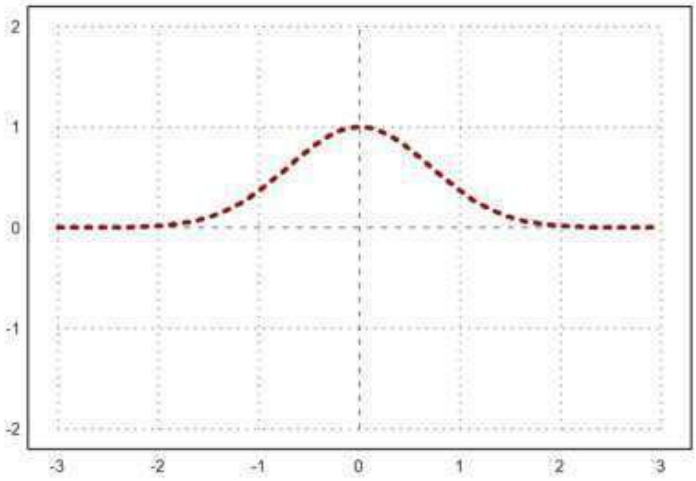

Untuk gaya garis tersedia berbagai opsi:

- style="...". Pilih dari "-", "--", "-.", ".", ".-.", "-.-".
- color: lihat penjelasan warna di bawah.
- thickness: bawaan adalah 1.

Warna dapat dipilih dari warna bawaan, atau sebagai warna RGB.

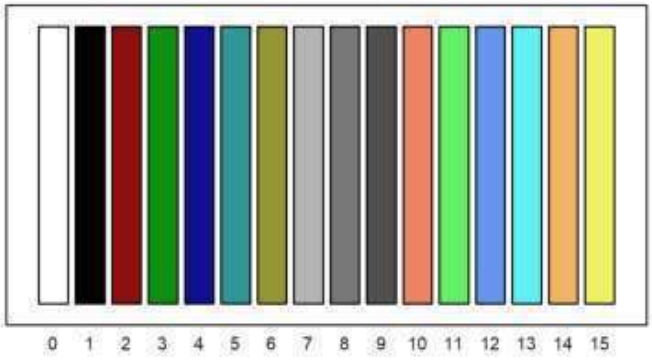
- 0..15: indeks warna bawaan
- konstanta warna: white, black, red, green, blue, cyan, olive, lightgray, gray, darkgray, orange, lightgreen, turquoise, lightblue, lightorange, yellow
- rgb(red,green,blue): parameternya berupa bilangan real dalam \[0,1]

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



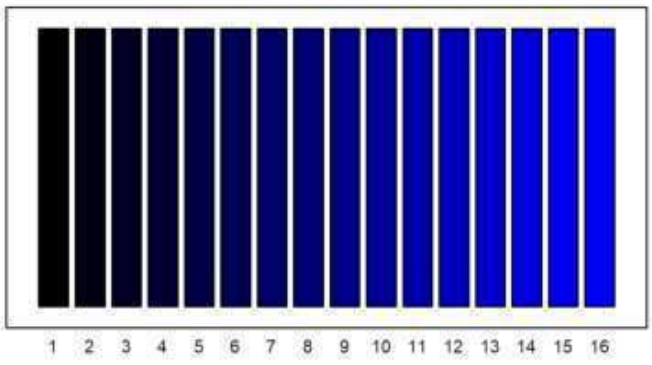

Berikut adalah tampilan warna-warna yang sudah didefinisikan sebelumnya di EMT.

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




Namun, Anda bisa menggunakan warna apa saja.

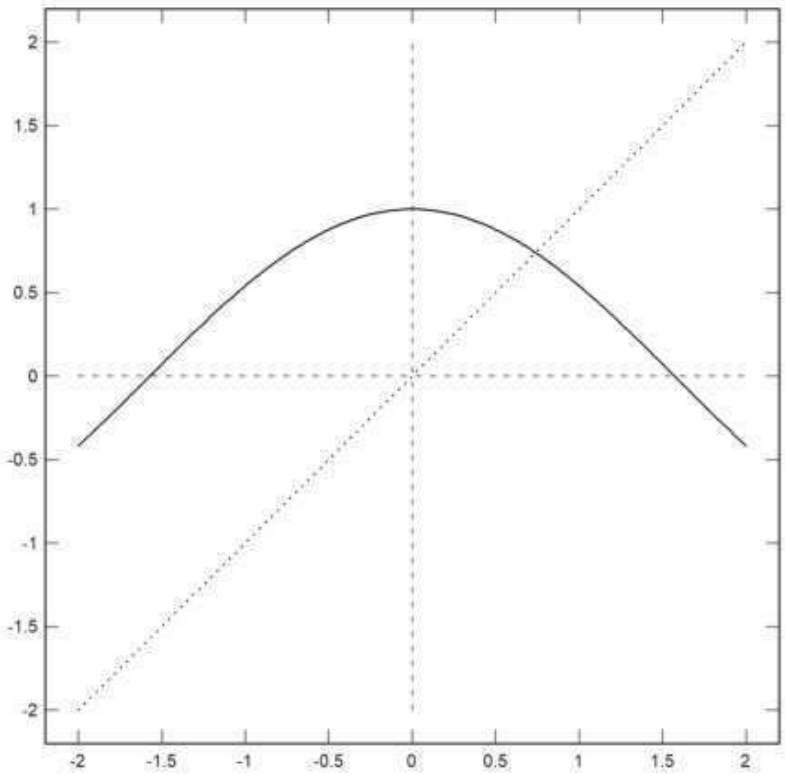
```
>columnsplot(ones(1,16),grid=0,color=rgb(0,0,linspace(0,1,15))):
```



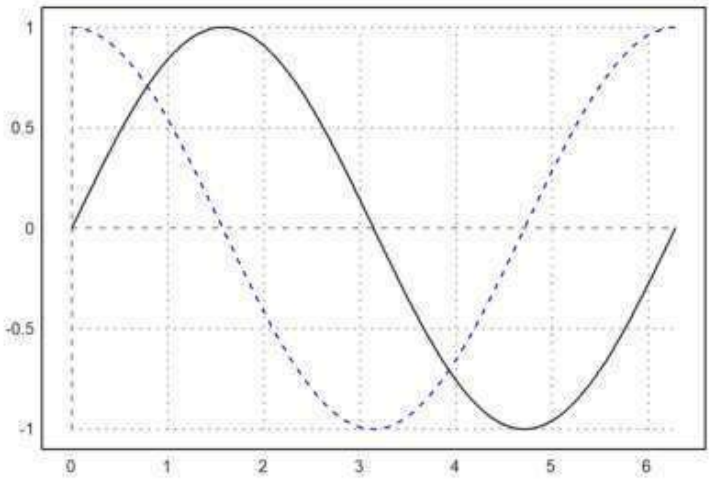

## Menggambar beberapa kurva pada bidang koordinat yang sama

Memplot lebih dari satu fungsi (fungsi ganda) dalam satu jendela dapat dilakukan dengan berbagai cara. Salah satu caranya adalah dengan menggunakan >add pada beberapa pemanggilan plot2d, kecuali pada pemanggilan pertama. Fitur ini sudah kita gunakan pada contoh-contoh sebelumnya.

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

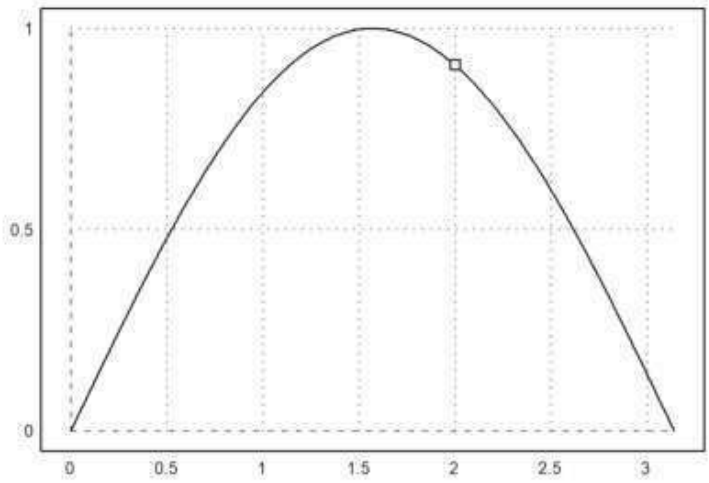



```
>aspect(1.5); plot2d("sin(x)",0,2pi); plot2d("cos(x)",color=blue,style="--",>add):
```



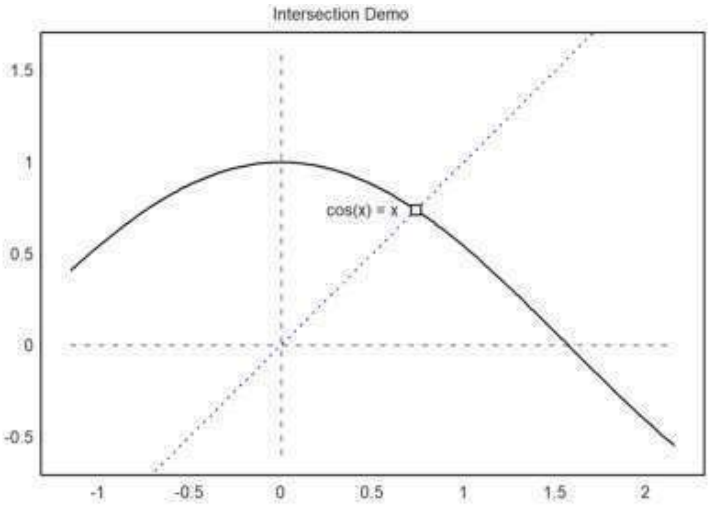

Salah satu kegunaan >add adalah untuk menambahkan titik pada kurva.

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




Kita menambahkan titik potong dengan sebuah label (pada posisi "cl" untuk center left), dan menyisipkan hasilnya ke dalam notebook. Kita 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, kita memplot fungsi sinc(x)=sin(x)/x serta ekspansi Taylor ke-8 dan ke-16-nya. Ekspansi ini dihitung menggunakan Maxima melalui ekspresi simbolik.

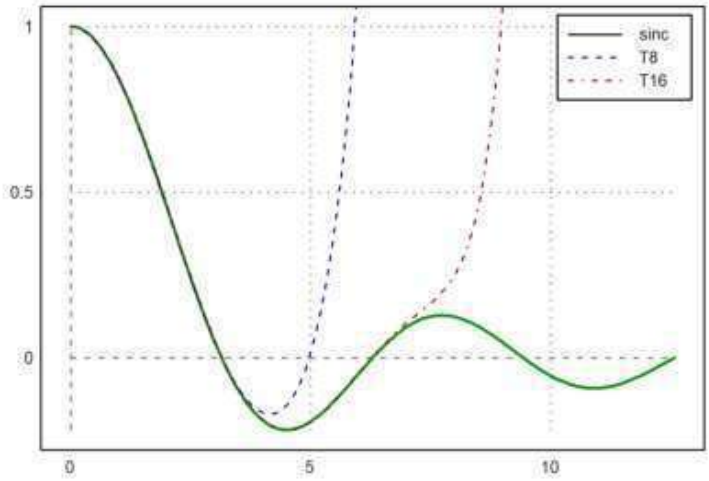
Plot ini dibuat dengan perintah multi-baris yang berisi tiga pemanggilan plot2d(). Pemanggilan kedua dan ketiga menggunakan flag >add, sehingga plot tersebut menggunakan rentang yang sama dengan plot sebelumnya.

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

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

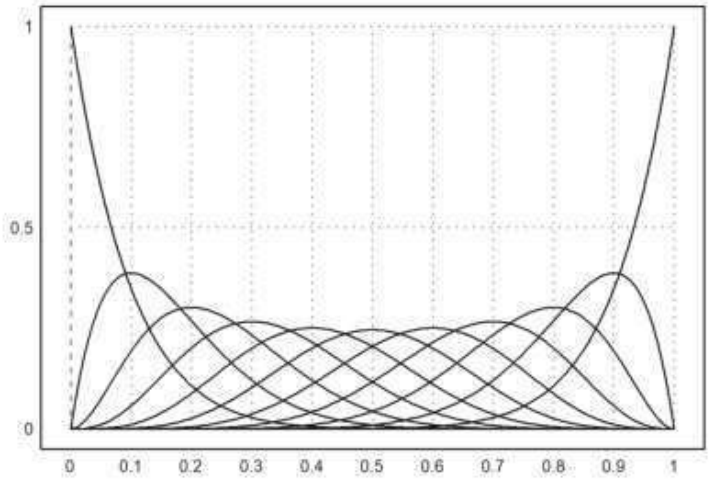
$$\frac{x^4}{120}-\frac{x^2}{6}+1$$

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




Pada contoh berikut, kita menghasilkan Polinomial Bernstein.

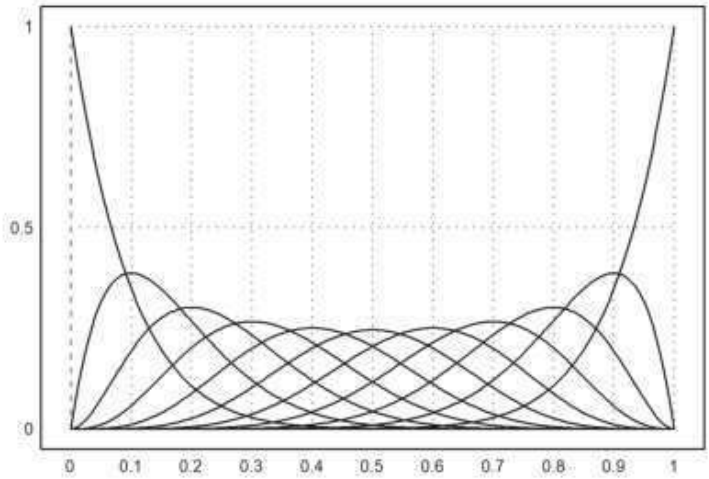
```
>plot2d("(1-x)^10",0,1); // plot fungsi pertama
>for i=1 to 10; plot2d("bin(10,i)*x^i*(1-x)^(10-i)",>add); end;
>insimg;
```




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

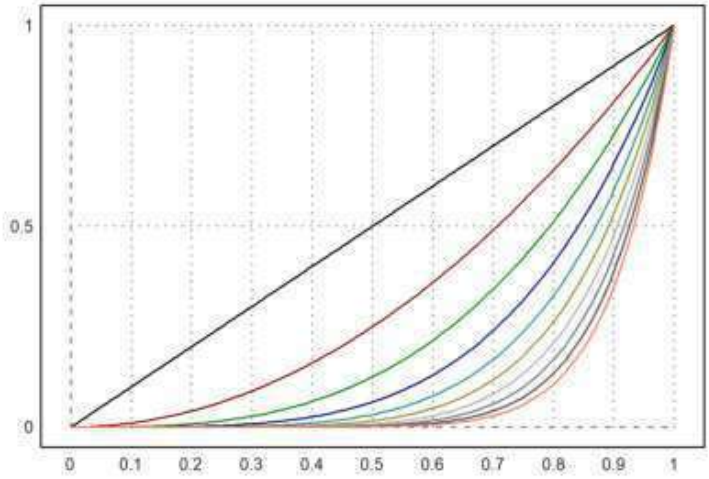
Kita menghasilkan sebuah matriks nilai dengan satu Polinomial Bernstein pada setiap barisnya. Untuk ini, kita cukup menggunakan vektor kolom i. Lihatlah bagian pengantar tentang bahasa matriks untuk mempelajari lebih detail.

```
>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):
```



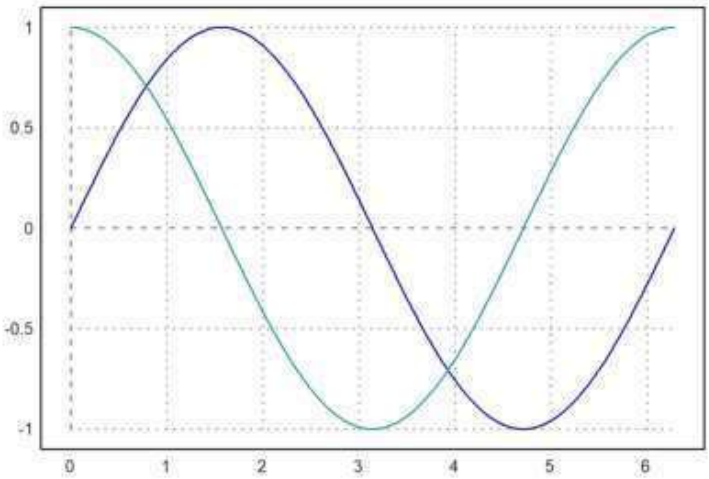

Perlu dicatat bahwa parameter color dapat berupa sebuah vektor. Dengan begitu, setiap warna akan digunakan untuk setiap baris dari matriks.

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

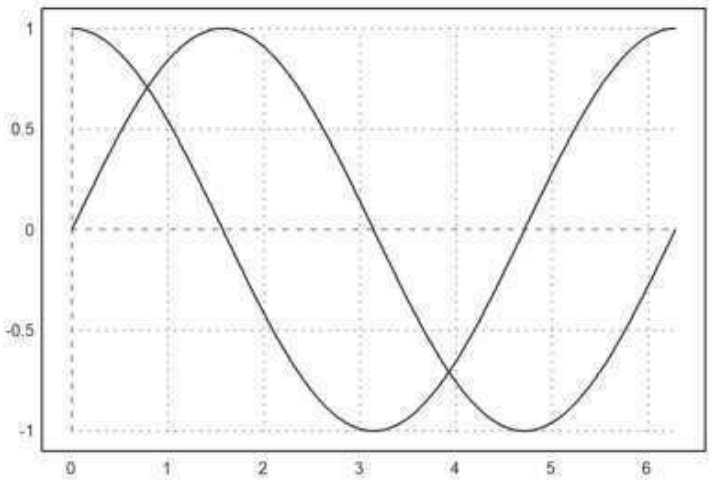



Metode lain adalah dengan menggunakan vektor ekspresi (string). Anda kemudian dapat menggunakan array warna, array gaya, dan array ketebalan dengan panjang yang sama.

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




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




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

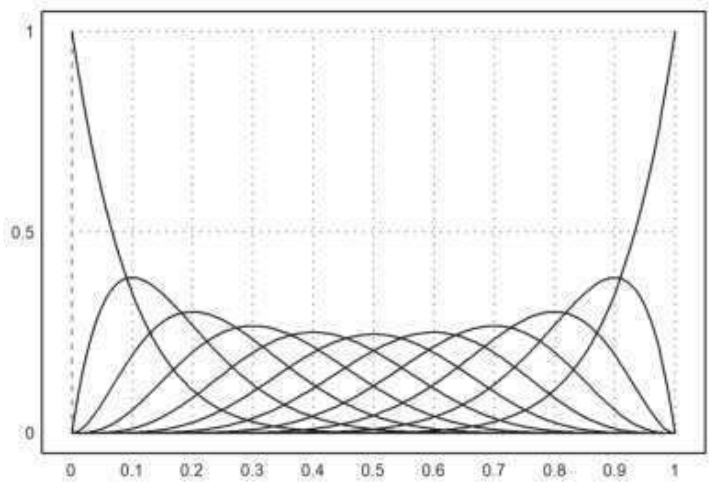
```
>v &= makelist(binomial(10,i)*x^i*(1-x)^(10-i),i,0,10) // membuat daftar
```

$$\left[(1-x)^{10},\ 10\,(1-x)^{9}\,x,\ 45\,(1-x)^{8}\,x^{2},\ 120\,(1-x)^{7}\,x^{3},\ 210\,(1-x)^{6}\,x^{4},\ 252\,(1-x)^{5}\,x^{5},\ 210\,(1-x)^{4}\,x^{6},\ 120\,(1-x)^{3}\,x^{7},\ 45\,(1-x)^{2}\,x^{8},\ 10\,(1-x)\,x^{9},\ x^{10}\right]$$

```
>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): // plot fungsi
```

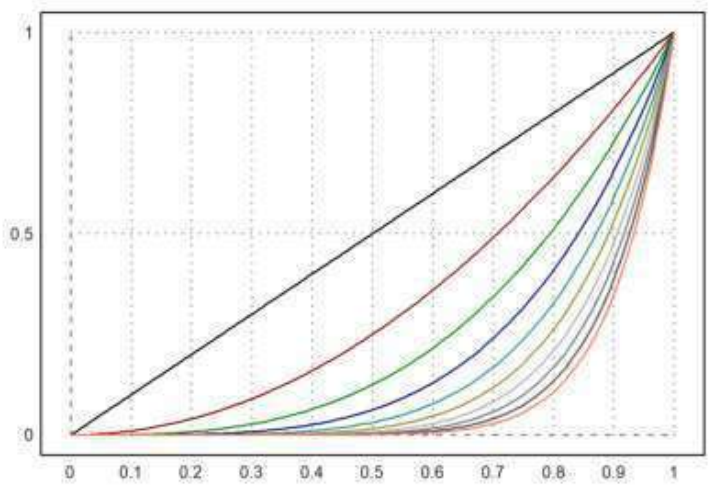



Alternatif lain adalah menggunakan bahasa matriks dari Euler.

Jika sebuah ekspresi menghasilkan matriks fungsi, dengan satu fungsi di setiap baris, maka semua fungsi tersebut akan diplot dalam satu plot.

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

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

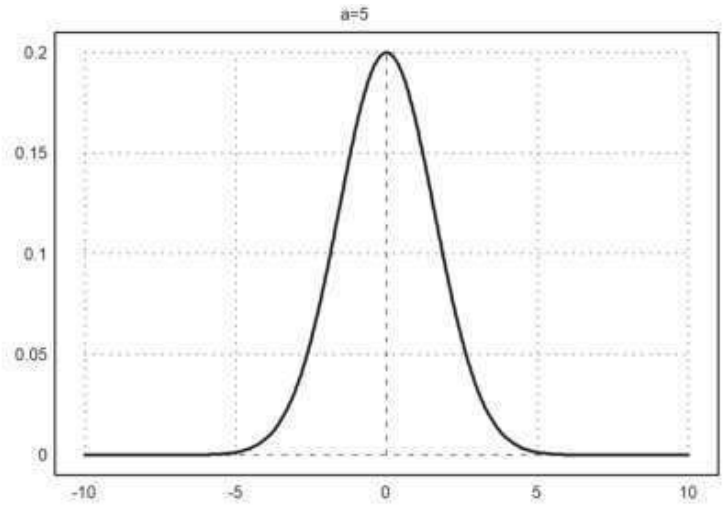



Ekspresi dan fungsi satu baris dapat melihat variabel global.

Jika Anda tidak bisa menggunakan variabel global, Anda perlu membuat fungsi dengan parameter tambahan, dan meneruskan parameter tersebut sebagai parameter titik koma.

Perhatikan agar semua parameter yang ditetapkan diletakkan di bagian akhir perintah plot2d. Pada contoh, kita meneruskan a=5 ke fungsi f, yang kemudian diplot dari -10 sampai 10.

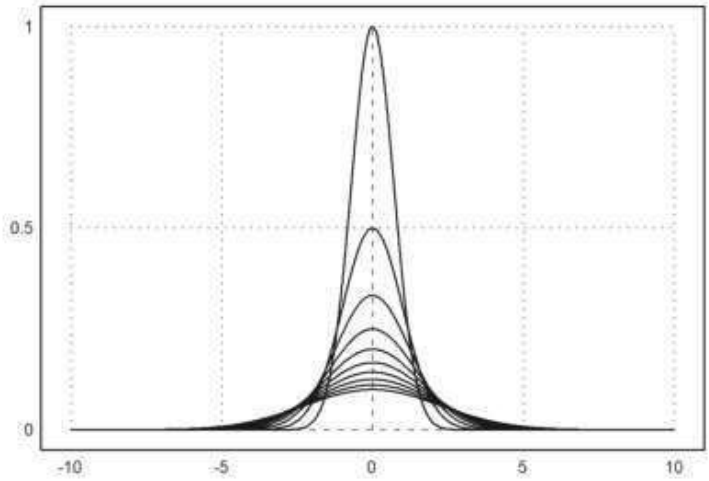
```
>function f(x,a) := 1/a*exp(-x^2/a); ...
 plot2d("f",-10,10;5,thickness=2,title="a=5"):
```




Sebagai alternatif, gunakan sebuah koleksi dengan nama fungsi dan semua parameter tambahan. Daftar khusus ini disebut call collections, dan ini merupakan cara yang lebih disarankan untuk meneruskan argumen ke sebuah fungsi yang dirinya sendiri diteruskan sebagai argumen ke fungsi lain.

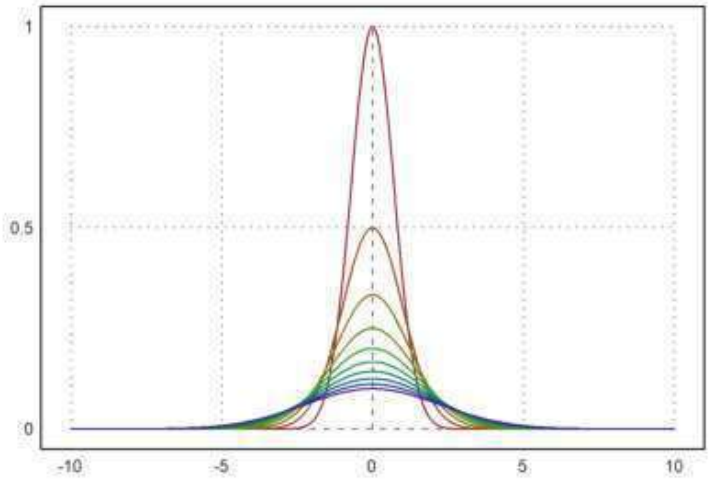
Pada contoh berikut, kita menggunakan sebuah loop untuk memplot beberapa fungsi (lihat tutorial tentang pemrograman untuk loop).

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




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

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



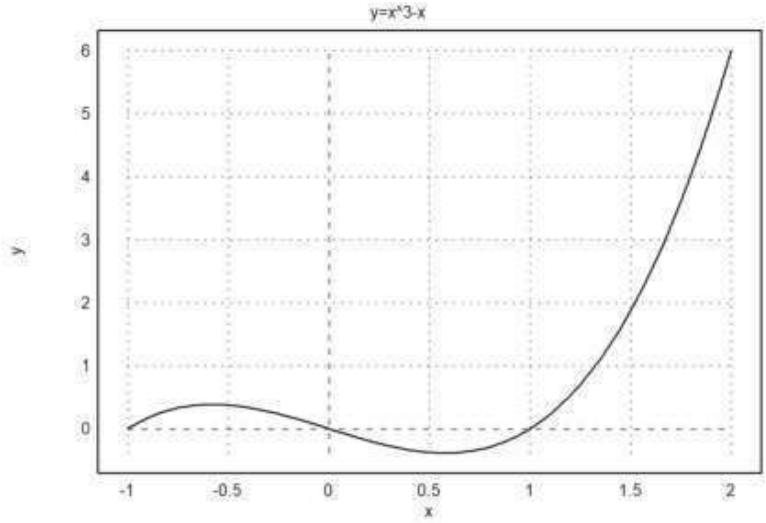

## Teks Label

Hiasan sederhana dapat berupa

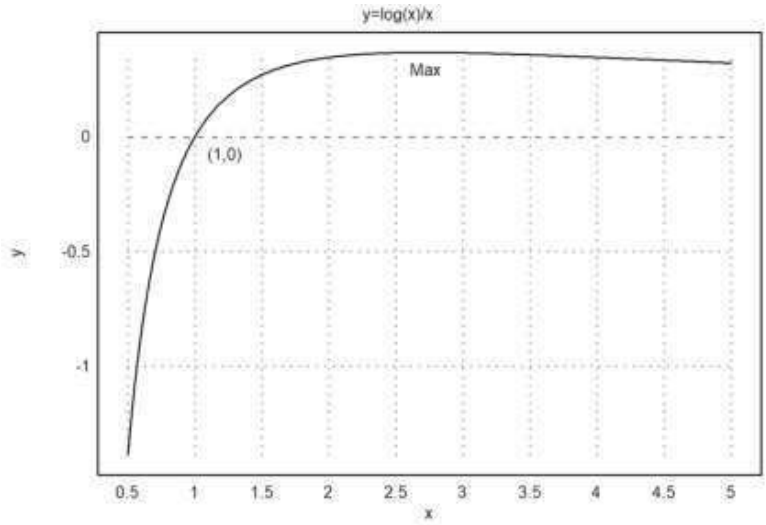
- judul dengan title="..."
- label sumbu-x dan sumbu-y dengan xl="...", yl="..."
- teks label lain dengan label("...",x,y)

Perintah label akan menambahkan teks ke plot saat ini pada koordinat plot (x,y). Perintah ini juga dapat menerima argumen posisi.

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

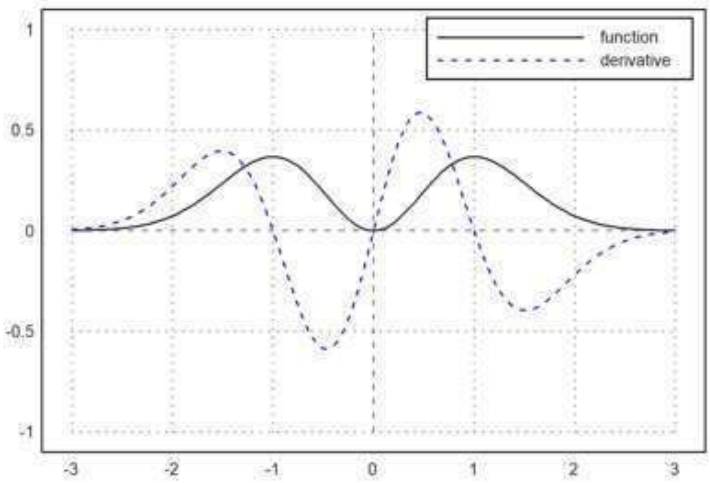



```
>expr := "log(x)/x"; ...
   plot2d(expr,0.5,5,title="y="+expr,xl="x",yl="y"); ...
   label("(1,0)",1,0); label("Max",E,expr(E),pos="lc"):
```




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

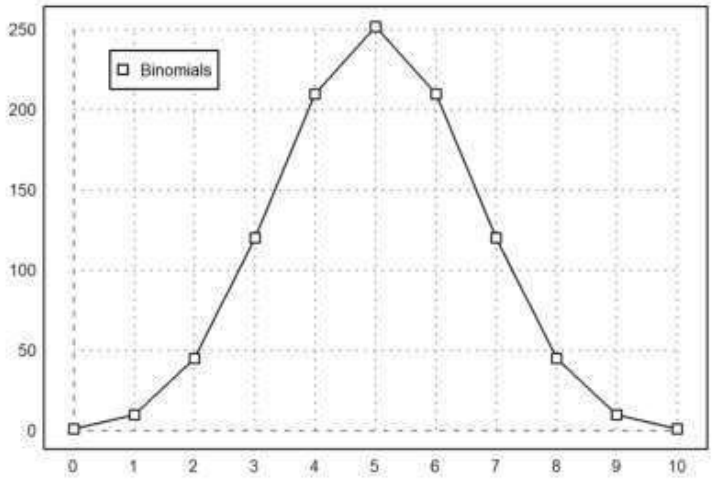


Kotak label secara bawaan ditempatkan di kanan atas, tetapi >left akan menempatkannya di kiri atas. Anda bisa memindahkannya ke posisi mana pun sesuai keinginan. Posisi acuan adalah sudut kanan atas kotak, dan angka-angka yang digunakan merupakan pecahan dari ukuran jendela grafik. Lebarnya ditentukan secara otomatis.

Untuk plot titik, kotak label juga dapat digunakan. Tambahkan parameter >points, atau sebuah vektor berisi flag, satu untuk setiap label.

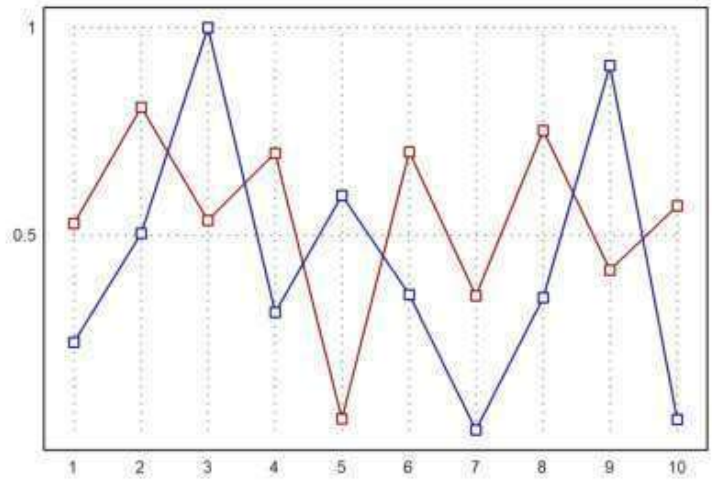
Pada contoh berikut, hanya ada satu fungsi. Jadi kita bisa menggunakan string tunggal, bukan vektor string. Kita juga mengatur warna teks menjadi hitam untuk contoh ini.

```
>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 seperti ini juga tersedia dalam statplot(). Seperti pada plot2d(), warna dapat diatur untuk setiap baris plot. Terdapat lebih banyak plot khusus untuk tujuan statistika (lihat tutorial tentang statistika).

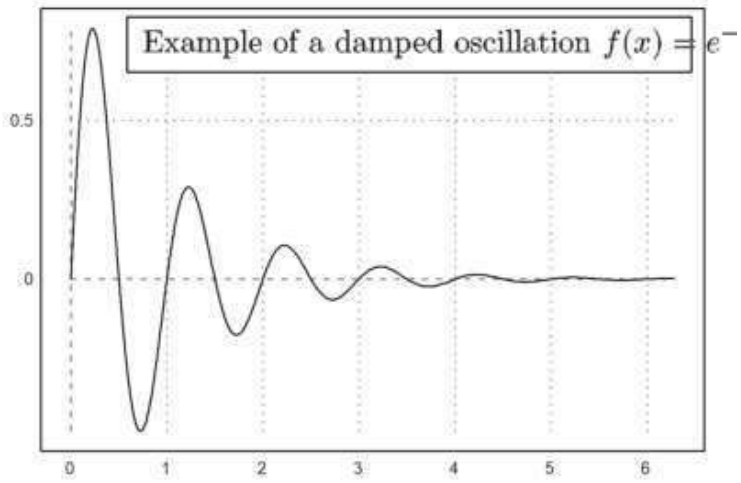
```
>statplot(1:10,random(2,10),color=[red,blue]):
```




Fitur serupa adalah fungsi textbox().

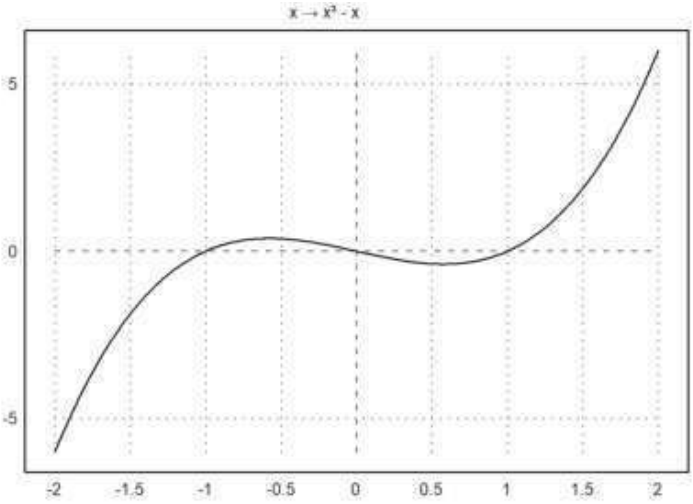
Lebarnya secara bawaan mengikuti lebar maksimal dari baris teks. Namun, lebar ini juga dapat diatur oleh pengguna.

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



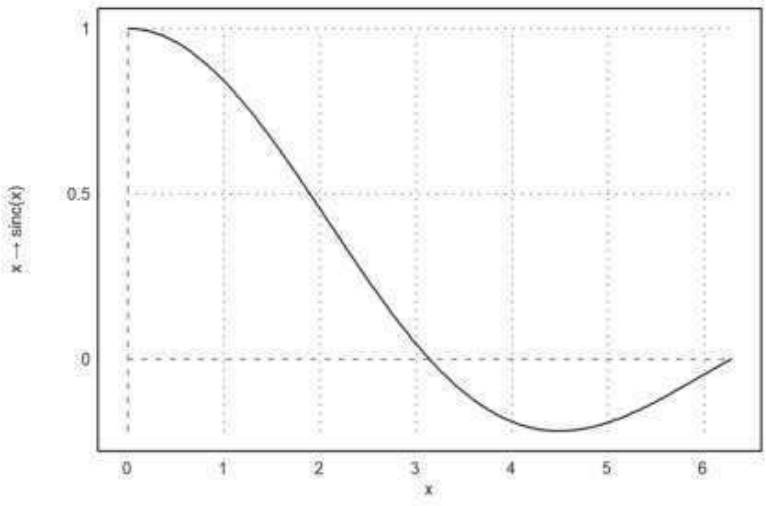

Label teks, judul, kotak label, dan teks lainnya dapat memuat string Unicode (lihat sintaks EMT untuk informasi lebih lanjut tentang string Unicode).

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




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

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



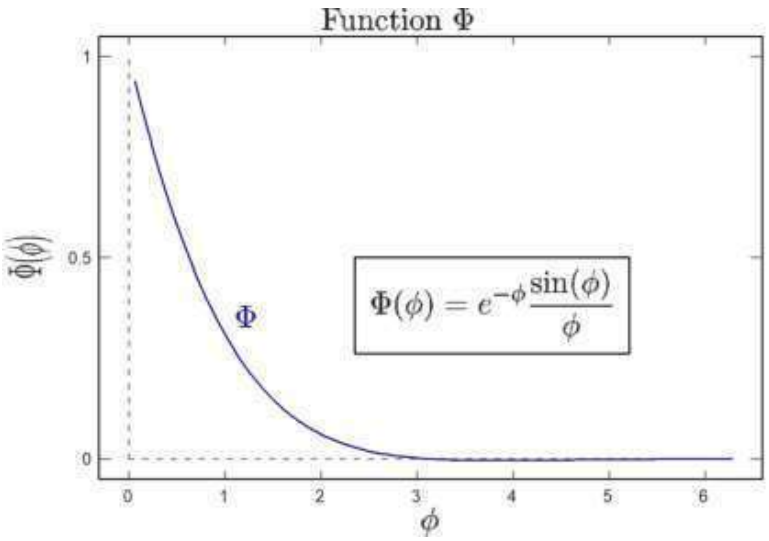

## LaTeX

Anda juga dapat memplot rumus LaTeX jika telah menginstal sistem LaTeX. Saya merekomendasikan MiKTeX. Path ke binary "latex" dan "dvipng" harus ada di system path, atau Anda perlu mengatur LaTeX di menu opsi.

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

Pada plot berikut, kita menggunakan LaTeX untuk label sumbu x dan y, sebuah label, sebuah kotak label, serta judul plot.

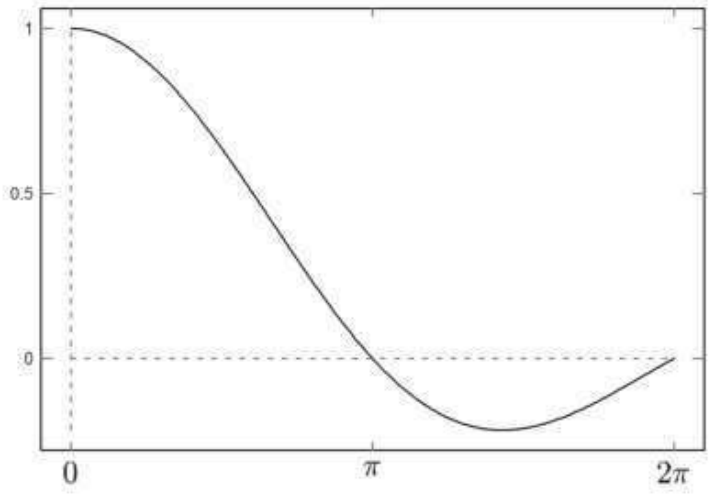
```
>plot2d("exp(-x)*sin(x)/x",a=0,b=2pi,c=0,d=1,grid=6,color=blue, ...
   title=latex("\text{Function $\Phi$}"), ...
   xl=latex("\phi"),yl=latex("\Phi(\phi)")); ...
 textbox( ...
   latex("\Phi(\phi) = e^{-\phi} \frac{\sin(\phi)}{\phi}"),x=0.8,y=0.5); ...
 label(latex("\Phi",color=blue),1,0.4):
```




Sering kali, kita menginginkan jarak yang tidak konformal dan label teks pada sumbu x. Kita dapat menggunakan xaxis() dan yaxis() seperti yang akan ditunjukkan nanti.

Cara termudah adalah membuat plot kosong dengan bingkai menggunakan grid=4, kemudian menambahkan garis grid dengan ygrid() dan xgrid(). Pada contoh berikut, kita menggunakan tiga string LaTeX sebagai 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):
```



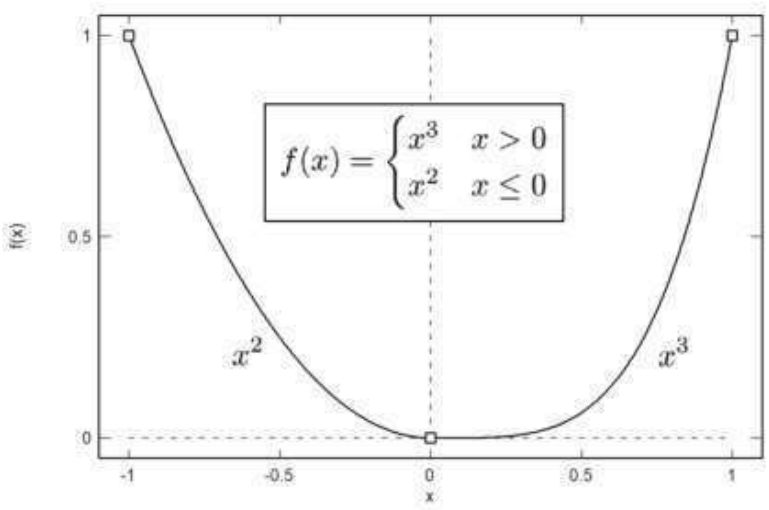

Tentu saja, fungsi juga dapat digunakan.

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

Parameter "map" membantu penggunaan fungsi untuk vektor. Untuk plot, sebenarnya hal ini tidak diperlukan. Namun, untuk menunjukkan bahwa vektorisasi itu berguna, kita menambahkan beberapa titik penting pada plot di x=-1, x=0, dan x=1.

Pada plot berikut, kita juga memasukkan beberapa kode LaTeX. Kode tersebut digunakan untuk dua label dan sebuah kotak teks. Tentu saja, Anda hanya bisa menggunakan LaTeX jika LaTeX telah terpasang dengan benar.

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




## Interaksi Pengguna

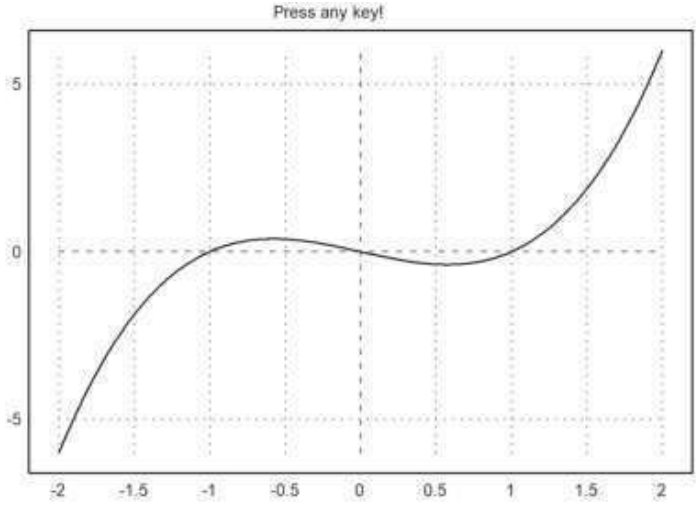
Saat memplot sebuah fungsi atau ekspresi, parameter >user memungkinkan pengguna untuk memperbesar dan menggeser plot dengan tombol panah atau mouse. Pengguna dapat

- memperbesar dengan tombol + atau -
- menggeser plot dengan tombol panah
- memilih jendela plot dengan mouse
- mereset tampilan dengan spasi
- keluar dengan tombol return

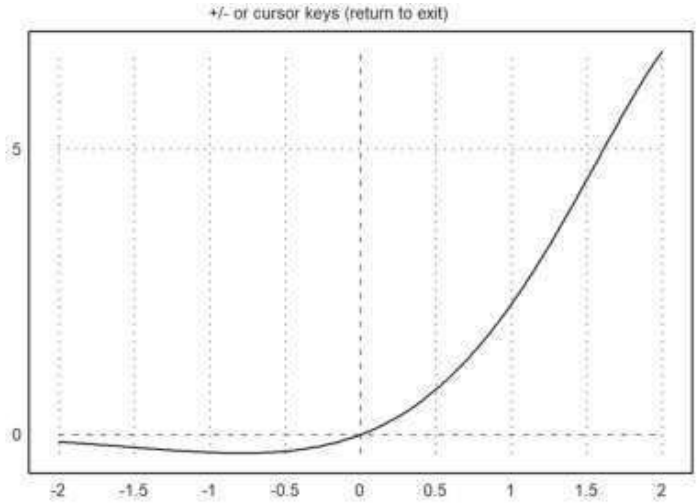
Tombol spasi akan mengembalikan plot ke jendela plot semula.

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 menunjukkan cara lanjutan interaksi pengguna (lihat tutorial tentang pemrograman untuk detail).

Fungsi bawaan mousedrag() menunggu kejadian mouse atau keyboard. Fungsi ini melaporkan mouse down, mouse moved, atau mouse up, serta penekanan tombol. Fungsi dragpoints() memanfaatkan ini, dan memungkinkan pengguna menyeret titik mana pun dalam plot.

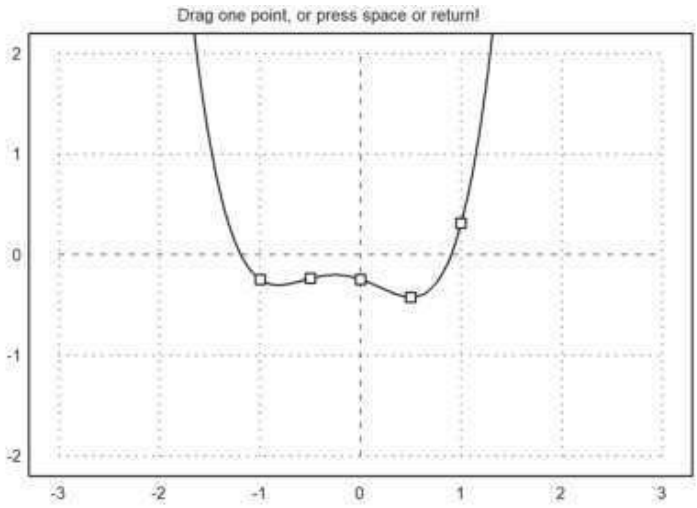
Kita memerlukan sebuah fungsi plot terlebih dahulu. Sebagai contoh, kita menginterpolasi pada 5 titik dengan sebuah polinomial. Fungsi tersebut harus memplot ke dalam area plot yang tetap.

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

Perhatikan parameter titik koma dalam plot2d (d dan xp), yang diteruskan ke evaluasi fungsi interp(). Tanpa ini, kita harus menulis fungsi plotinterp() terlebih dahulu, yang 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 sebuah fungsi yang memplot fungsi lain yang bergantung pada sebuah vektor parameter, dan memungkinkan pengguna menyesuaikan parameter-parameter tersebut.

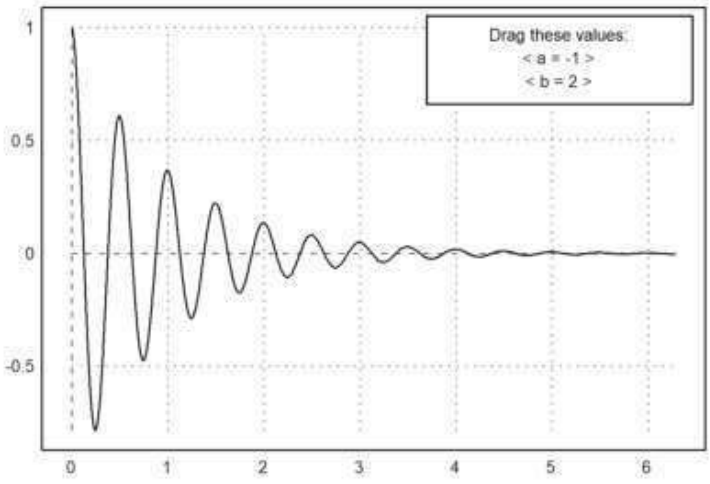
Pertama, kita memerlukan fungsi plotnya.

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

Lalu kita memerlukan nama-nama parameter, nilai awal, dan sebuah matriks nx2 yang berisi rentang nilai, serta opsional satu baris judul.

Terdapat slider interaktif yang memungkinkan pengguna mengatur nilai. Fungsi dragvalues() menyediakan fitur ini.

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



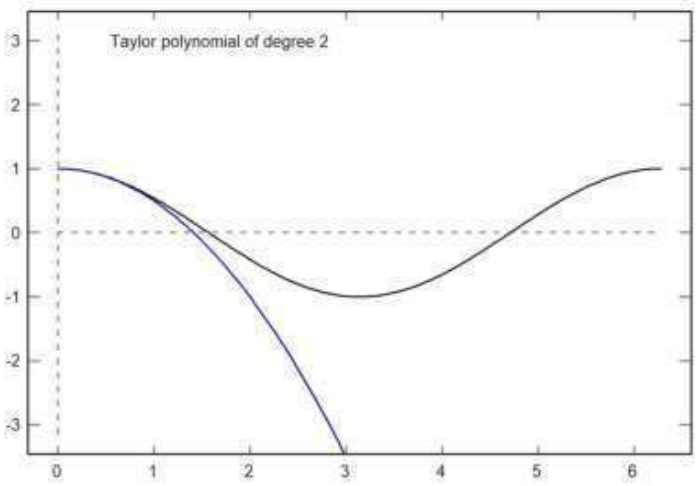

Nilai yang digeser dapat dibatasi hanya pada bilangan bulat. Sebagai contoh, kita menulis sebuah fungsi plot yang memplot polinomial Taylor berderajat n untuk fungsi cosinus.

```
>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 membolehkan derajat n bervariasi dari 0 hingga 20 dalam 20 langkah. Hasil dari dragvalues() digunakan untuk memplot sketsa dengan nilai n tersebut, dan menyisipkan plot ke dalam notebook.

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

```
plotf(nd):
```




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

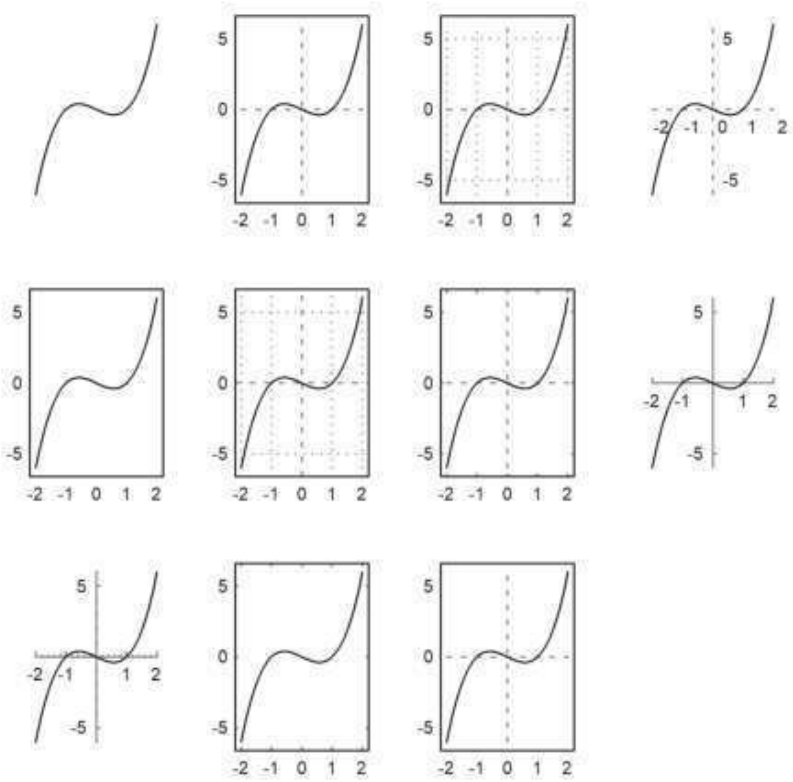
```
>function dragtest ...
   plot2d(none,r=1,title="Drag with the mouse, or press any key!");
   start=0;
   repeat
     {flag,m,time}=mousedrag();
     if flag==0 then return; endif;
     if flag==2 then
       hold on; mark(m[1],m[2]); hold off;
     endif;
   end
 endfunction
```

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

## Gaya Plot 2D

Secara bawaan, EMT menghitung tanda sumbu secara otomatis dan menambahkan label pada setiap tanda. Hal ini dapat diubah dengan parameter grid. Gaya bawaan dari sumbu dan label dapat dimodifikasi. Selain itu, label dan judul juga dapat ditambahkan secara manual. Untuk mengembalikan ke gaya bawaan, gunakan reset().

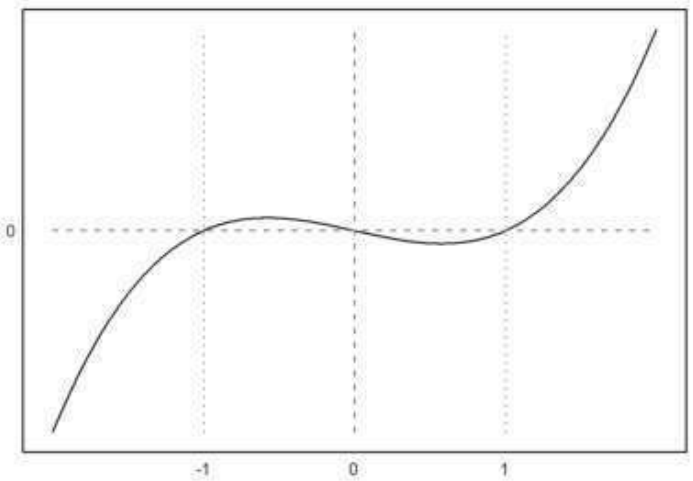
```
>aspect();
>figure(3,4); ...
  figure(1); plot2d("x^3-x",grid=0); ... // tanpa grid, bingkai, atau tanda sumbu
> figure(2); plot2d("x^3-x",grid=1); ... // sumbu x-y
> figure(3); plot2d("x^3-x",grid=2); ... // tanda bawaan
> figure(4); plot2d("x^3-x",grid=3); ... // sumbu x-y dengan label di dalam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



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

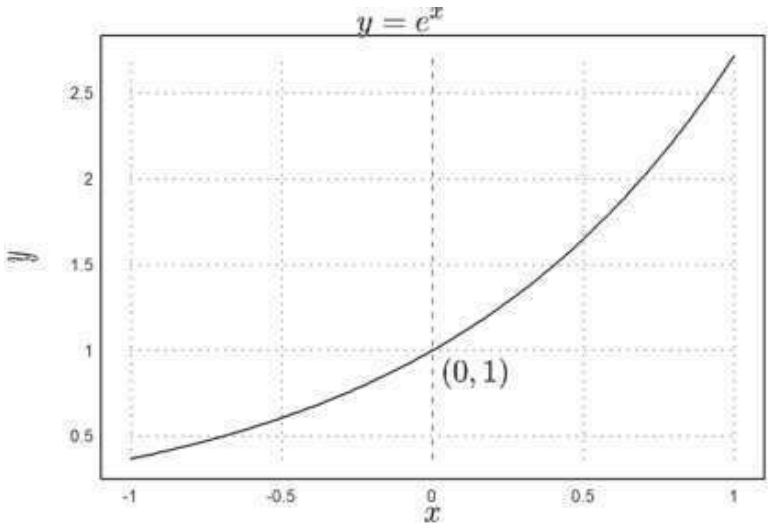
Jika Anda ingin menggunakan tanda sumbu sendiri, Anda dapat menggunakan style=0, lalu menambahkan semuanya secara manual.

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



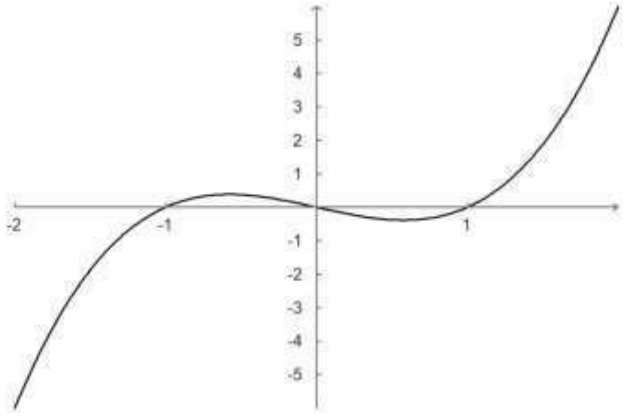
Untuk judul plot dan label sumbu, lihat contoh berikut.

```
>plot2d("exp(x)",-1,1);
>textcolor(black); // atur warna teks menjadi hitam
>title(latex("y=e^x")); // judul di atas plot
>xlabel(latex("x")); // "x" untuk sumbu x
>ylabel(latex("y"),>vertical); // "y" vertikal untuk sumbu y
>label(latex("(0,1)"),0,1,color=blue): // beri label pada sebuah 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 ditambahkan dengan label(). Pada contoh berikut, "lc" berarti lower center. Ini mengatur posisi label relatif terhadap koordinat plot.

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

$$x^3 - x$$

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