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

Terdapat fungsi plot yang sangat mendasar. Terdapat koordinat layar, yang selalu berkisar 0 hingga 1024 di setiap sumbu, tidak berpengaruh jika layar itu persegi atau tidak. Selain itu ada koordinat plot yang dapat diatur dengan setplot(). Pemetaan antara koordinat bergantung pada jendela plot saat inI. Misalnya shrinkwindow() menyisakan ruangan tuntuk label sumbu dan judul plot

Sebagai contoh, kita hanya menggambar beberapa garis acak dengan berbagai warna. Untuk detail dalam fungsi fungsi ini.

```
> clg;
>window(0,0,1024,1024);
>setplot(0,1,0,1);
>hold on;
>n=5; X=random(n,2); Y=random(n,2);
>colors=rgb(random(n),random(n),random(n));
>loop 1 to n; color(colors[#]); plot(X[#],Y[#]); end;
>hold off;
>insimg;
```

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

Sebagai contoh lain, kita menggambar sebuah plot sebagai inset pada plot yang lain. Hal ini dilakukan dengan menentukan jendela plot yang lain. Perhatikan bahwa jendela ini tidak menyediakan ruang untuk label sumbu diluar jendela plot. kita harus menambahkan beberapa margin sesuai kebutuhan. Perhatikan bahwa kita menyimpan dan mengembalikan jendela penuh dan menahan plot saat kita memplot inset.

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

Plot dengan beberapa gambar dibuat dengan cara yang sama. Terdapat fungsi figure() fungsi untuk ini.

Aspek Plot

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

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

```
>aspect(2);
>plot2d("x^2-5x+6",1,4):
>reset;
```

Plot 2D di Euler

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

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

Euler dapat memplot plot 2D dari

- ekspresi

-fungsi, variabel, atau kurva berparameter,

-vektor nilai x,y,

-titik-titik pada bidang,

-kurva implisit dengan level atau wilayah level.

-Fungsi yang kompleks

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

Plot Ekspresi atau Variabel

Ekspresi dalam "x" (e.g. "4*x^2") atau nama dari sebuah fungsi (e.g. "f") menghasilakn grafik fungsi.

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

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

```
>plot2d("4x^2"):
>aspect(1.5); plot2d("x^3-x"):
>a:=5; plot2d("(-a*x^2-3x)"); insimg(50);
```

Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan nilai-nilai batas X (dan Y) di belakang ekspresi yang digambar.

```
>plot2d("x^3-x",-1,2):
>plot2d("sin(x)",-2*pi,2*pi):
>plot2d("cos(x)","sin(3*x)",xmin=0,xmax=2pi):
```

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

Dalam opsi, plot dapat diatur agar muncul

- di jendela terpisah yang dapat diubah ukurannya,
- di jendela buku catatan.

Lebih banyak gaya dapat dicapai dengan perintah plot tertentu.

Bagaimanapun, tekan tombol tabulator untuk melihat plotnya, jika tersembunyi.

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

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

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

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

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

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

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

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

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

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

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

```
>plot2d("sign(x)*exp(-x^2)",-1,1,<adaptive,n=1000):
>plot2d("x^x",r=1.2,cx=1,cy=1):
```

Perhatikan bahwa xˆx tidak ditentukan untuk x<=0. Fungsi plot2d menangkap kesalahan ini, dan mulai membuat plot segera setelah fungsinya ditentukan. Ini berfungsi untuk semua fungsi yang mengembalikan NAN di luar jangkauan definisinya.

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

The parameter square=true (or >square) selects the y-range automatically so that the result is a square plot window. Note that by default, Euler uses a square space inside the plot window.

```
>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 y, panggil shrinkwindow() dengan parameter lebih kecil, atau tetapkan nilai positif untuk "lebih kecil" 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; // definisi ekspresi
>plot2d(expr,-2,2): // plot dari -2 hingga 2
>plot2d(&diff(expr,x),>add,style="--",color=red): // menambahkan plot
>plot2d(&diff(expr,x,2),a=-2,b=2,c=-2,d=1): // plot didalam persgi
>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 di satu Parameter

Fungsi plot yang paling pentung untuk plot planar adalah plot2d(). Fungsi ini diimplementasikan dalam bahasa Euler dalam file “plot.e”, yang dimuat pada awal program.

Berikut ini beberapa contoh penggunaan fungsi. Seperti biasa dalam EMT, fungsi yang bekerja untuk fungsi atau ekspresi lain, Anda dapat mengoper parameter tambahan (selain x) yang bukan variabel global ke fungsi dengan parameter titik koma atau dengan koleksi panggilan.
on.

```
>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 ini adalah ringkasan dari fungsi yang diterima

- ekspresi atau ekspresi simbolik dalam x
- fungsi atau fungsi simbolik dengan nama sebagai “f”

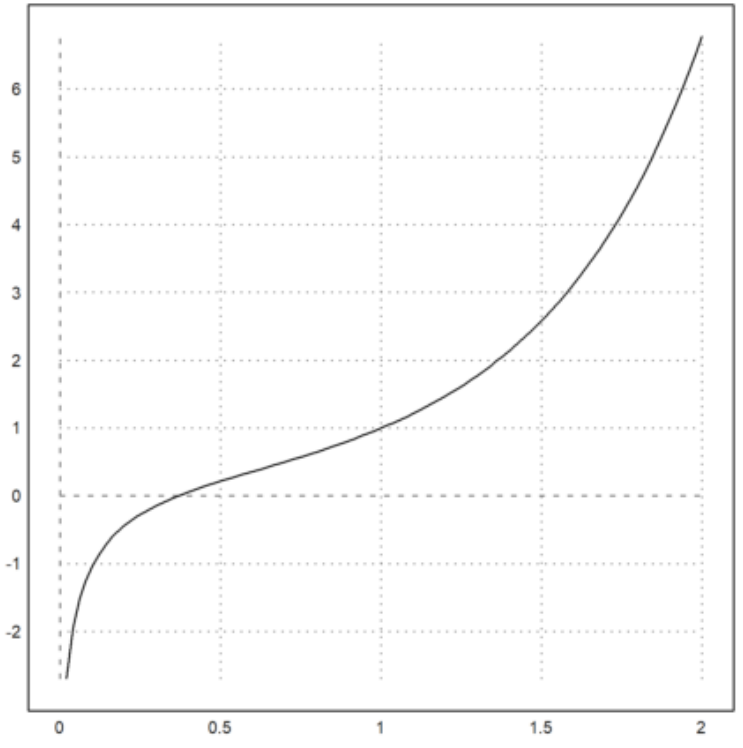
```
fungsi-fungsi simbolik hanya dengan nama f
```

ungsi 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):
```

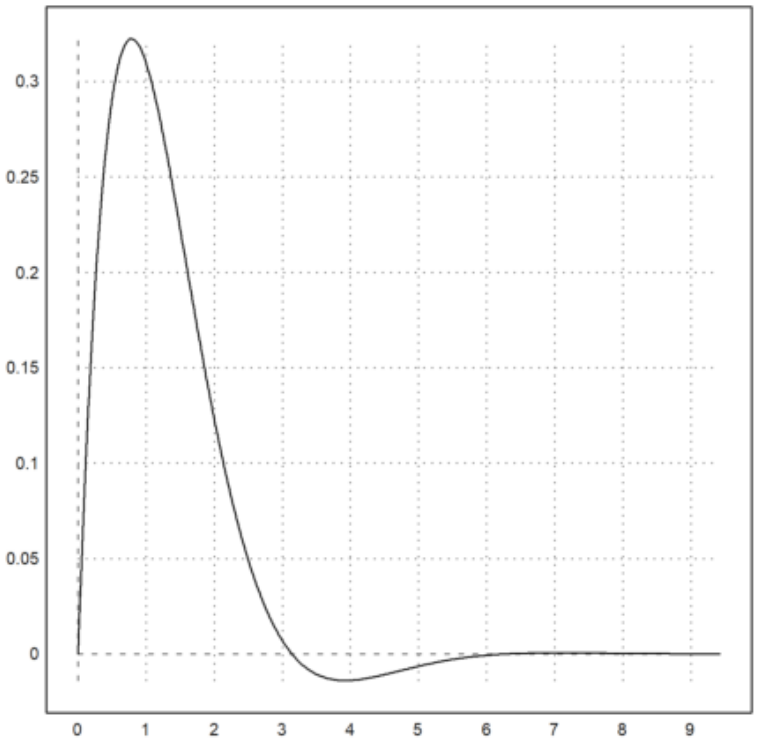


Tentu, untuk ekspresi atau ungkapan simbol nama variable sudah cukup untuk memplotnya

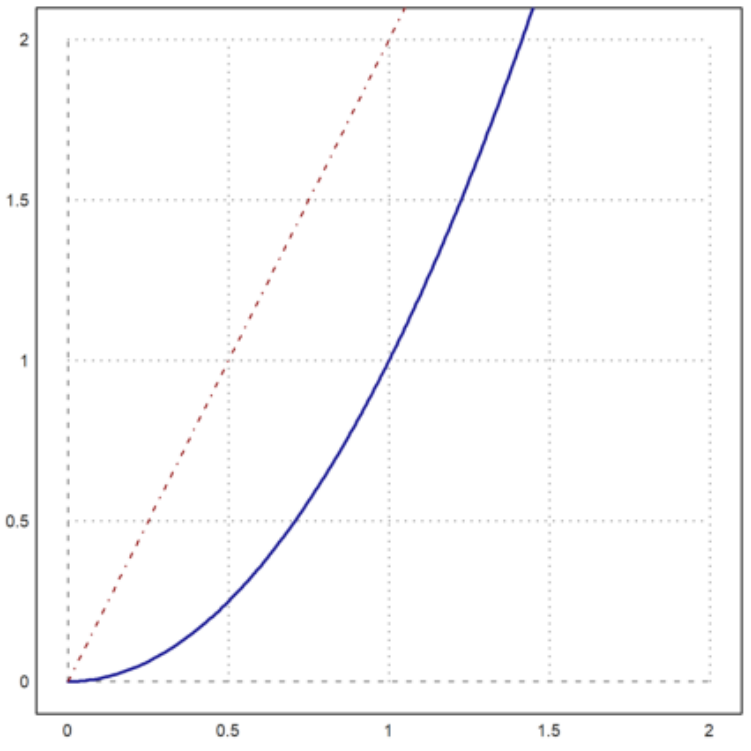
```
>expr &= sin(x)*exp(-x)
```

```
  - x
E    sin(x)
```

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



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



Untuk gaya garis terdapat beberapa pilihan.

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

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

- 0..15: indeks warna bawaan
- konstanta warna: putih, hitam, merah, hijau, biru, cyan, zaitun, abu-abu muda, abu-abu, abu-abu tua, oranye, hijau muda, pirus, biru muda, oranye muda, kuning
- rgb(red,green,blue): parameter asli dalam [0,1].

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

Berikut adalah warna EMT yang sudah diteteapkan sebelumnya.

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

Warna dapat diubah sesuai keinginan andaa

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

Menggambar Beberapa Kurva pada bidang koordinat yang sama

Plot yang terdiri lebih dari satu fungsi (multiple function) ke dalam satu jendela dapat dilakukan dengan berbagai cara. Salah satu metodenya adalah menggunakan >add untuk beberapa panggilan ke plot2d secara keseluruhan, kecuali panggilan pertama.

```
>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 tambahkan titik perpotongan dengan label (pada posisi "cl" untuk kiri tengah), dan masukkan hasilnya ke dalam buku catatan. Selain itu, 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 contoh berikut, kita memplot fungsi sin(x)=sin(x)/x dan ekspansi Taylor ke-8 dan ke-16. Dengan menghitung perluasan ini menggunakan Maxima melalui ekspresi simbolik.
Plot ini dilakukan dalam perintah multi-baris berikut dengan tiga panggilan ke plot2d(). Yang kedua dan ketiga memiliki kumpulan tanda >add, yang membuat plot menggunakan rentang sebelumnya.

Selain itu, dapat menambahkan kotak label yang menjelaskan fungsinya.

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

Cara kedua adalah dengan menggunakan pasangan matriks bernilai x dan matriks bernilai y yang berukuran sama.

Sehingga menghasilkan matriks nilai dengan satu Polinomial Bernstein di setiap baris. Untuk ini, kita cukup menggunakan vektor kolom i.

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

Perhatikan bahwa parameter warna dapat berupa vektor. Kemudian setiap warna digunakan untuk setiap baris 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 susunan warna, susunan gaya, dan susunan 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
```

Untuk dapat menghasilkan vektor seperti itu dari Maxima menggunakan makelist() dan mxm2str().tor from Maxima using makelist() and mxm2str().

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

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

```
>mxm2str(v) // 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
```

Dapat juga menggunakan bahasa matriks Euler.

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

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

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

Ekspresi dan fungsi satu baris dapat melihat variabel global.

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

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

```
>function f(x,a) := 1/a*exp(-x^2/a); ...
```

Contoh penggunaan loop untuk memplot beberapa fungsi

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

```
>plot2d("f",-10,10;5,thickness=2,title="a=5"):
>plot2d({{"f",1}},-10,10); ...
>for a=2:10; plot2d({{"f",a}},>add); end:
>plot2d({{"f",1}},-15,15); ...
>for a=2:15; plot2d({{"f",a}},>add); end:
>function f(x,a) := 1/a*exp(-x^2/a); ...
>x=-10:0.01:10; a=(1:10)'; plot2d(x,f(x,a),color=getspectral(a/10)):
```

Kita dapat mencapai hasil yang sama dengan cara berikut menggunakan bahasa matriks EMT. Setiap baris matriks f(x,a) merupakan satu fungsi. Selain itu, kita dapat mengatur warna untuk setiap baris matriks. Klik dua kali pada fungsi getspectral() untuk penjelasannya.

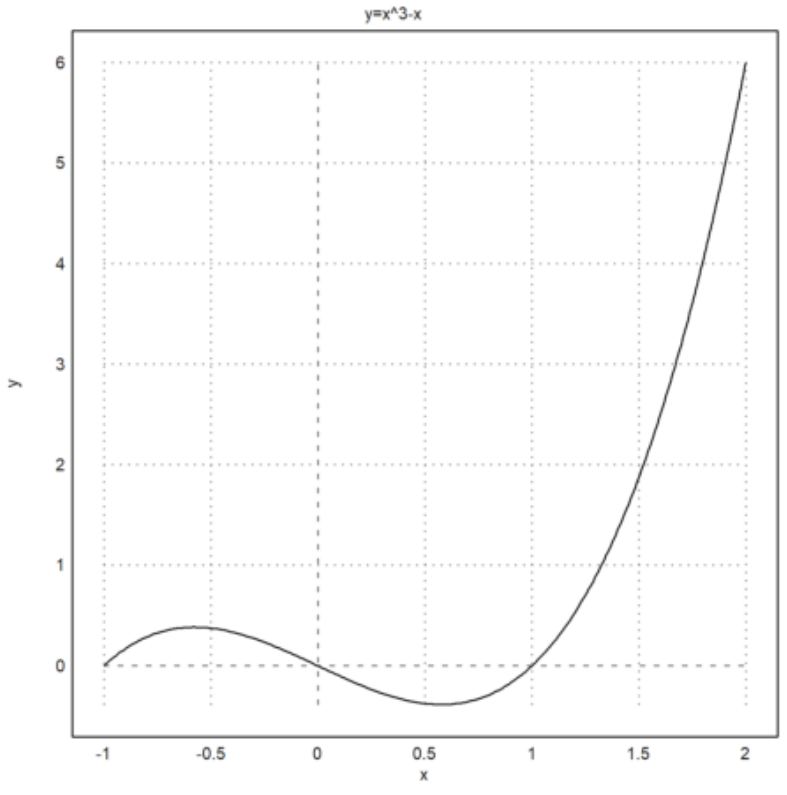
Menuliskan Label Kurva

Dekorasi sederhana pun bisa

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

Perintah label akan memplot ke plot saat ini pada koordinat plot (x,y). Hal ini memerlukan argumen posisional.

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




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

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

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

Kotak ini berada di bagian kanan atas secara default, tetapi >kiri berada di kiri atas. Anda dapat memindahkannya ke tempat mana pun yang Anda suka. Posisi jangkar berada di pojok kanan atas kotak, dan angkanya merupakan pecahan dari ukuran jendela grafis. Lebarnya otomatis.

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

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

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

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

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

Fitur serupa adalah fungsi textbox().

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

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

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

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

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

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

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

Pada plot berikut ini, 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):
```

Seringkali, kita menginginkan spasi dan label teks yang tidak sesuai pada sumbu x. Kita dapat menggunakan xaxis() dan yaxis() seperti yang akan kita tunjukkan nanti.

Cara termudah adalah dengan membuat plot kosong dengan sebuah frame menggunakan grid=4, dan kemudian menambahkan grid dengan ygrid() dan xgrid(). Pada contoh berikut, kita menggunakan tiga buah string LaTeX untuk label pada sumbu x dengan xtick().

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

Tentu saja, fungsi juga dapat digunakan.

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

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

Parameter “map” membantu menggunakan fungsi untuk vektor. Untuk plot, hal ini tidak diperlukan. Tetapi untuk mendemonstrasikan bahwa vektorisasi berguna, kami menambahkan beberapa titik kunci pada plot pada x=-1, x=0 dan x=1.

Pada plot berikut, kita juga memasukkan beberapa kode LaTeX. Kita menggunakannya untuk dua label dan sebuah kotak

```
>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

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

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

Tombol spasi akan mengatur ulang plot ke jendela plot awal.

Ketika memplot data, bendera >user hanya akan menunggu penekanan tombol.

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

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

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

Fungsi bawaan mousedrag() menunggu peristiwa mouse atau keyboard. Fungsi ini melaporkan mouse ke bawah, mouse bergerak atau mouse ke atas, dan penekanan tombol. Fungsi dragpoints() memanfaatkan hal ini, dan mengizinkan pengguna untuk menyeret titik manapun di dalam plot.

Kita membutuhkan fungsi plot terlebih dahulu. Sebagai contoh, kita melakukan interpolasi pada 5 titik dengan sebuah polinomial. Fungsi ini 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 pada plot2d (d dan xp), yang diteruskan ke evaluasi fungsi interp(). Tanpa ini, kita harus menulis fungsi plotinterp() terlebih dahulu, untuk mengakses nilai secara global.

Sekarang kita menghasilkan beberapa nilai acak, dan membiarkan pengguna menarik titik-titiknya.

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

Ada juga fungsi yang memplot fungsi lain tergantung pada vektor parameter, dan memungkinkan pengguna menyesuaikan 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 untuk parameter, nilai awal dan matriks rentang nx2, dan secara opsional, sebuah garis judul.
Terdapat slider interaktif, yang dapat mengatur nilai oleh pengguna. Fungsi dragvalues() menyediakan ini.

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

Anda dapat membatasi nilai yang diseret menjadi bilangan bulat. Sebagai contoh, kita menulis fungsi plot, yang memplot polinomial Taylor dengan derajat n ke fungsi kosinus.

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

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

Sekarang kita membiarkan derajat n bervariasi dari 0 sampai 20 dalam 20 stop. Hasil dari dragvalues() digunakan untuk memplot sketsa dengan n ini, dan untuk menyisipkan plot ke dalam buku catatan.

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

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

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

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

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

Gaya Plot 2D

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

```
>aspect();
>figure(3,4); ...
> figure(1); plot2d("x^3-x",grid=0); ... // no grid, frame or axis
> figure(2); plot2d("x^3-x",grid=1); ... // x-y-axis
> figure(3); plot2d("x^3-x",grid=2); ... // default ticks
> figure(4); plot2d("x^3-x",grid=3); ... // x-y- axis with labels inside
> figure(5); plot2d("x^3-x",grid=4); ... // no ticks, only labels
> figure(6); plot2d("x^3-x",grid=5); ... // default, but no margin
> figure(7); plot2d("x^3-x",grid=6); ... // axes only
> figure(8); plot2d("x^3-x",grid=7); ... // axes only, ticks at axis
> figure(9); plot2d("x^3-x",grid=8); ... // axes only, finer ticks at axis
> figure(10); plot2d("x^3-x",grid=9); ... // default, small ticks inside
> figure(11); plot2d("x^3-x",grid=10); ...// no ticks, axes only
> figure(0):
```

Parameter <frame mematikan bingkai, dan framecolor=blue menetapkan bingkai ke warna biru.

Jika Anda menginginkan tanda centang Anda sendiri, Anda dapat menggunakan style=0, dan menambahkan semuanya nanti.

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

Untuk judul plot dan label sumbu, lihat contoh berikut.

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

Sumbu dapat digambar secara terpisah dengan sumbu x() dan sumbu y().

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

Teks pada plot dapat diatur dengan label(). Pada contoh berikut ini, “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
```

Terdapat juga kotak teks.

```
>plot2d(&f(x),-1,1,-2,2); // function
>plot2d(&diff(f(x),x),>add,style="--",color=red); // derivative
>labelbox(["f","f'"],["-","--"],[black,red]): // label box
>plot2d(["exp(x)","1+x"],color=[black,blue],style=["-","-."]):
>gridstyle("->",color=gray,textcolor=gray,framecolor=gray);  ...
> plot2d("x^3-x",grid=1);   ...
> settitle("y=x^3-x",color=black); ...
> label("x",2,0,pos="bc",color=gray);  ...
> label("y",0,6,pos="cl",color=gray); ...
> reset():
```

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

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

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

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

```
>aspect(1.5);
>plot2d(["sin(x)","cos(x)"],0,2pi,color=[red,green],<grid,<frame); ...
> xaxis(-1.1,(0:2)*pi,xt=["0",u"&pi;",u"2&pi;"],style="-",>ticks,>zero);  ...
> xgrid((0:0.5:2)*pi,<ticks); ...
> yaxis(-0.1*pi,-1:0.2:1,style="-",>zero,>grid); ...
> labelbox(["sin","cos"],colors=[red,green],x=0.5,y=0.2,>left); ...
> xlabel(u"&phi;"); ylabel(u"f(&phi;)"):
```

If x and y are data vectors, these data will be used as x- and y-coordinates of a curve. In this case, a, b, c, and d, or a radius r can be specified, or the plot window will adjust automatically to the data. Alternatively, >square can be set to keep a square aspect ratio.

Plotting an expression is only an abbreviation for data plots. For data plots, you need one or more rows of x-values, and one or more rows of y-values. From the range and the x-values the plot2d function will compute the data to plot, by default with adaptive evaluation of the function. For point plots use ">points", for mixed lines and points use ">addpoints".

But you can enter data directly.

- Use row vectors for x and y for one function.
- Matrices for x and y are plotted line by line.

Here is an example with one row for x and y.

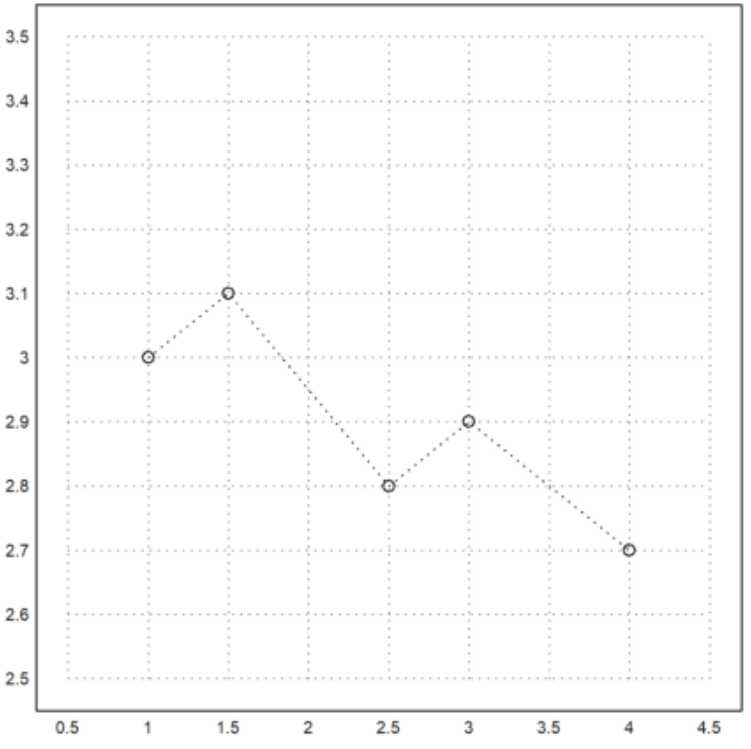
```
>x=-10:0.1:10; y=exp(-x^2)*x; plot2d(x,y):
```

Data can also be plotted as points. Use points=true for this. The plot works like polygons, but draws only the corners.

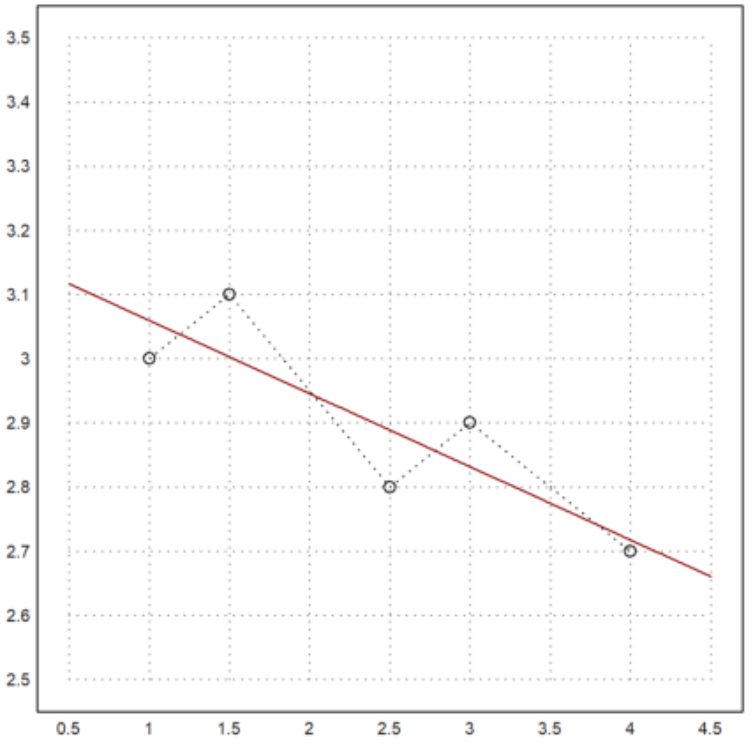
- style="...": Select from "[]", "<>", "o", ".", "..", "+", "*", "[]", "<>", "o", "..", "", "|".

To plot sets of points use >points. If the color is a vector of colors, each points
gets a different color. For a matrix of coordinates and a column vector, the color applies to the rows of the matrix.
The parameter >addpoints adds points to line segments for plots of 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

Data plots are really polygons. We can also plot curves or filled curves.

- filled=true fills the plot.
- style="...": Select from "", "/", "\", "\/".
- fillcolor: See above for available colors.

The fill color is determined by the argument "fillcolor", and on optional <outline prevents drawing the boundary for all styles but the default one.

```
>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):
```

In the following examples we plot a filled ellips and two filled hexagons using a closed curve with 6 points with different fill style.

```
>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="#"):
```

Another example is a septagon, which we create with 7 points on the unit circle.

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

The following is the set of the maximal value of four linear conditions less than or equal 3. This is A[k].v<=3 for all rows of A. To get nice corners, we use n relatively large.

```
>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):
```

The main point of the matrix language is that it allows to generate tables of functions easily.

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

We now have vectors x and y of values. plot2d() can plot these values as a curve connecting the points. The plot can be filled. In this case this yields a nice result due to the winding rule, which is used for the fill.

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

A vector of intervals is plotted against x values as a filled region between lower and upper values of the intervals.

This is can be useful to plot errors of the computation. But it can also be used to plot statistical errors.

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

If x is a sorted vector, and y is a vector of intervals, then plot2d will plot the filled ranges of the intervals in the plane.The fill styles are the same as the styles of polygons.

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

It is possible to fill regions of values for a specific function. For this, level must be a 2xn matrix. The first row are the lower bounds and the second row contains the upper bounds.

```
>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
```

We can also fill ranges of values like

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

```
>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

The x-values need not be sorted. (x,y) simply describes a curve. If x is sorted, the curve is a graph of a function.

In the following example, we plot the spiral

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

We either need to use very many points for a smooth look or the function adaptive() to evaluate the expressions (see the function adaptive() for more details).

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

Alternatively, it is possible to use two expressions for curves. The following plots the same curve as above.

```
>plot2d("x*cos(2*pi*x)","x*sin(2*pi*x)",xmin=0,xmax=1,r=1):
>t=linspace(0,1,1000); r=exp(-t); x=r*cos(2pi*t); y=r*sin(2pi*t);
>plot2d(x,y,r=1):
```

In the next example, we plot the curve

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

with

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

```
>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

An array of complex numbers can also be plotted. Then the grid points will be connected. If a number of grid lines is specified (or a 1x2 vector of grid lines) in the argument cgrid only those grid lines are visible.

A matrix of complex numbers will automatically plot as a grid in the complex plane.

In the following example, we plot the image of the unit circle under the exponential function. The cgrid parameter hides some of the grid curves.

```
>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):
```

A vector of complex numbers is automatically plotted as a curve in the complex plane with real part and imaginary part.

In the example, we plot the unit circle with

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

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

Statistical Plots

There are many functions which are specialized on statistical plots. One of the often used plots is a column plot.

A cumulative sum of a 0-1-normal distributed values produces a random walk.

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

Using two rows shows a walk in two dimensions.

```
>X=cumsum(randnormal(2,1000)); plot2d(X[1],X[2]):
>columnsplot(cumsum(random(10)),style="/",color=blue):
```

It can also show strings as labels.

```
>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):
```

To plot an experimental statistical distribution, you can use distribution=n with plot2d.

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

Or you can compute the distribution from the data and plot the result with >bar in plot3d, or with a 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):
```

The statplot() function sets the style with a simple string.

```
>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):
```

Moreover, data can be plotted as bars. In this case, x should be sorted and one element longer than y. The bars will extend from x[i] to x[i+1] with values y[i]. If x has the same size as y, it will be extended by one element with the last spacing.

Fill styles can be used just as above.

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

The data for bar plots (bar=1) and histograms (histogram=1) can either be explicitly given in xv and yv, or can be computed from an empirical distribution in xv with >distribution (or distribution=n). Histograms of xv values will be computed automatically with >histogram. If >even is specified, the xv values will be counted in integer intervals.

```
>plot2d(normal(10000),distribution=50):
>k=0:10; m=bin(10,k); x=(0:11)-0.5; plot2d(x,m,>bar):
>columnsplot(m,k):
>plot2d(random(600)*6,histogram=6):
```

For distributions, there is the parameter distribution=n, which counts values automatically and prints the relative distribution with n sub-intervals.

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

With the parameter even=true, this will use integer intervals.

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

Note that there are many statistical plots, which might be useful. Have a look at the tutorial about statistics.

```
>columnsplot(getmultiplicities(1:6,intrandom(1,6000,6))):
>plot2d(normal(1,1000),>distribution); ...
>  plot2d("qnormal(x)",color=red,thickness=2,>add):
```

There are also many special plots for statistics. A boxplot shows the quartiles of this distribution and lots of outliers. By definition, outliers in a boxplot are data which exceed 1.5 times the middle 50% range of the plot.

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

Fungsi Implisit

Plot implisit menunjukkan penyelesaian garis level f(x,y)=level, dengan "level" dapat berupa nilai tunggal atau vektor nilai. Jika level = "auto", akan ada garis level nc, yang akan tersebar antara fungsi minimum dan maksimum secara merata. Warna yang lebih gelap atau terang dapat ditambahkan dengan >hue untuk menunjukkan nilai fungsi. Untuk fungsi implisit, xv harus berupa fungsi atau ekspresi parameter x dan y, atau alternatifnya, xv dapat berupa matriks nilai.

Euler dapat menandai garis level

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

dari fungsi apa pun.

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

```
>aspect(1.5);
>plot2d("x^2+y^2-x*y-x",r=1.5,level=0,contourcolor=red):
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=0): // Solutions of f(x,y)=0
>plot2d(expr,level=0:0.5:20,>hue,contourcolor=white,n=200): // nice
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=200,grid=4): // nicer
```

Ini juga berfungsi untuk plot data. Namun, Anda harus menentukan rentangnya 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 set

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

dengan rentang level.

Dimungkinkan untuk mengisi wilayah 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
```

```
Variable or function expr not found.
Error in:
plot2d(expr,level=[0;1],style="-",color=blue): // 0 <= f(x,y)  ...
           ^
```

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

```
>plot2d("x^4+y^4",r=1.5,level=[0;1],color=blue,style="/"):
>plot2d("x^2+y^3+x*y",level=[0,2,4;1,3,5],style="/",r=2,n=100):
>plot2d("x^2+y^3+x*y",level=-10:20,r=2,style="-",dl=0.1,n=100):
>plot2d("sin(x)*cos(y)",r=pi,>hue,>levels,n=100):
```

Dimungkinkan juga untuk menandai suatu wilayah

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

Hal ini dilakukan dengan menambahkan level dengan dua baris.

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

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

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

```
>plot2d("x^3-x*y+x^2*y^2",r=6,level=1,n=100):
>function starplot1 (v, style="/", color=green, lab=none) ...
```

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

Tidak ada tanda centang kotak atau sumbu di sini. Selain itu, kami menggunakan jendela penuh untuk plotnya.

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

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

Terkadang, Anda mungkin ingin merencanakan sesuatu yang plot2d tidak bisa lakukan, tapi hampir.

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

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

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

Mari kita uji dengan nilai yang terdistribusi secara eksponensial.

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

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

Fungsi wait(0) memaksa plot muncul di jendela grafis. Jika tidak, pengundian ulang akan dilakukan dalam interval waktu yang jarang.

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

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

Tekan tombol apa saja untuk menghentikan animasi ini.

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

Plot Logaritma

EMT menggunakan parameter "logplot" untuk skala logaritmik.
Plot logaritma dapat diplot menggunakan skala logaritma di y dengan logplot=1, atau menggunakan skala logaritma di x dan y dengan logplot=2, atau di x dengan logplot=3.

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

```
>plot2d("exp(x^3-x)*x^2",1,5,logplot=1):
>plot2d("exp(x+sin(x))",0,100,logplot=1):
>plot2d("exp(x+sin(x))",10,100,logplot=2):
>plot2d("gamma(x)",1,10,logplot=1):
>plot2d("log(x*(2+sin(x/100)))",10,1000,logplot=3):
```

Ini juga berfungsi dengan plot data.

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

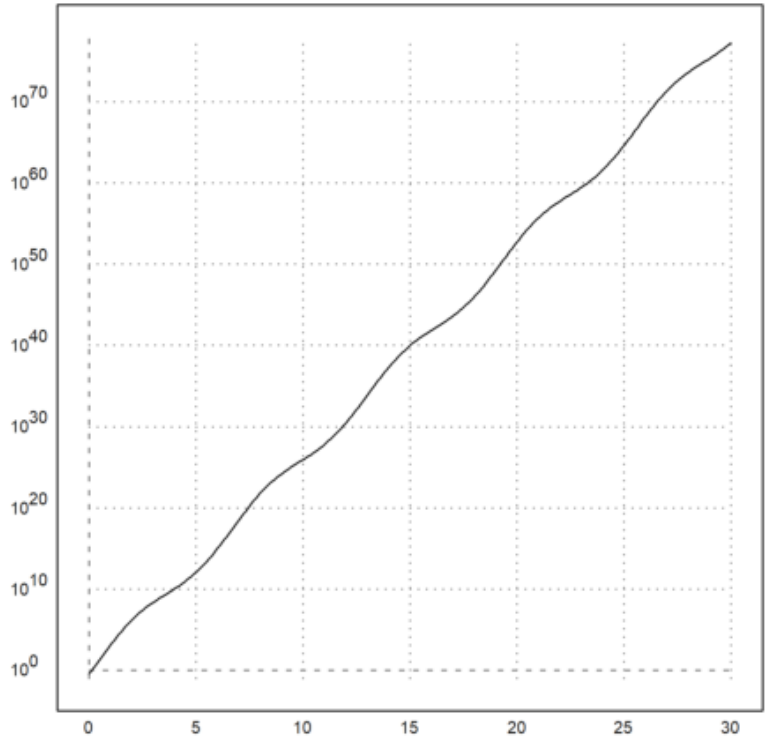
Latihan Soal

No.1

$$2sin(x) + 6x - cos(x)$$

Buatlah Plot Logaritma dengan persamaan diatas dengan rentan 0 hingga 30 dan logplot 1

```
>plot2d("exp(2*sin(x)+6*x-cos(x))",0,30,logplot=1):
```

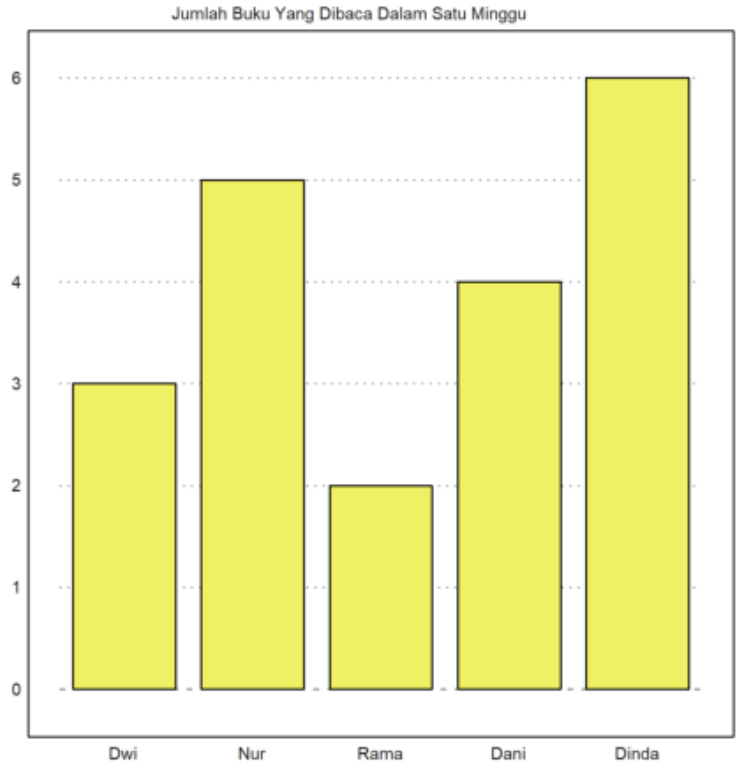


Penjelasan: Maka kurva yang didapatkan berbentuk kurva bergelombang dari titik x 0 hingga x 30.

No.2

Di sekolah Pelita Harapan terdapat 5 siswa yang gemar membaca buku. 5 siswa itu bernama Dwi, Nur, Rama, Dani, dan Dinda. Dalam seminggu Dwi membaca buku sebanyak 3, Nur sebanyak 5 buku, Rama 2 buku, Dani 4 buku dan Dinda sebanyak 6 buku. Buatlah diagram batang berwarna kuning lalu berikan judul diatas diagram.

```
>siswa=["Dwi","Nur","Rama","Dani","Dinda"];
>jumlahbuku=[3,5,2,4,6];
>columnsplot(jumlahbuku,lab=siswa,color=yellow,style="--");
>title("Jumlah Buku Yang Dibaca Dalam Satu Minggu"):
```



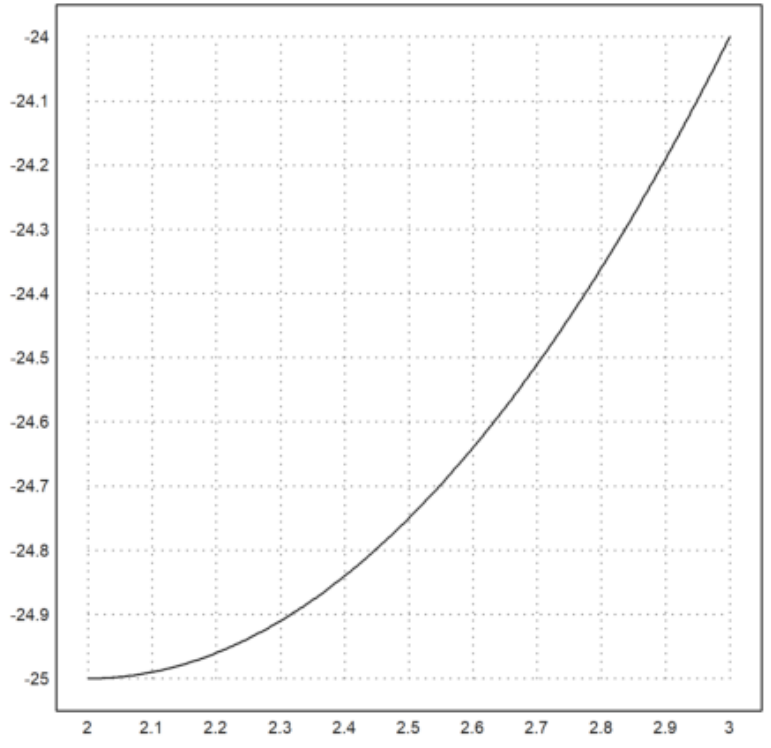
Penjelasan: Sesuai diagram diatas didapatkan bahwa siswa sekolah Pelita Harapan yang gemar membaca buku selama satu minggu paling banyak yaitu Dinda sebanyak 6 buku, Nur sebanyak 5 buku, Dani sebanyak 4 buku, Dwi sebanyak 3 buku dan yang terakhir yaitu Rama yang hanya membaca buku sebanyak 2.

No.3

Buatlah grafik fungsi dibawah dengan rentan x diantara 2 dan 3 dengan aspek 2

$$x^2 - 4x - 21$$

```
>aspect(2);
>plot2d("x^2-4*x-21",2,3):
```

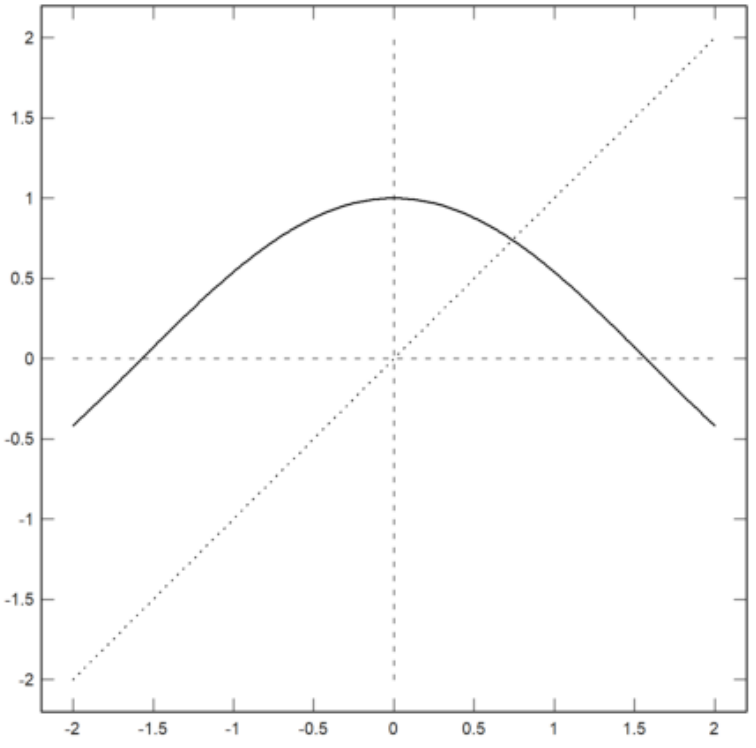


Penjelasan: Maka kurva diatas berbentuk melengkung keatas dari titik x 2 hingga x 3

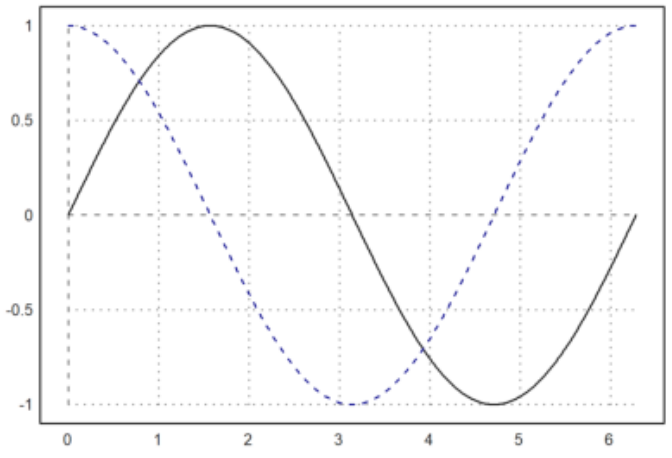
No.4

Gambar grafik dengan persamaan cos(x) kemudian gambar grafik lagi dengan persamaan cos x

```
>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):
```

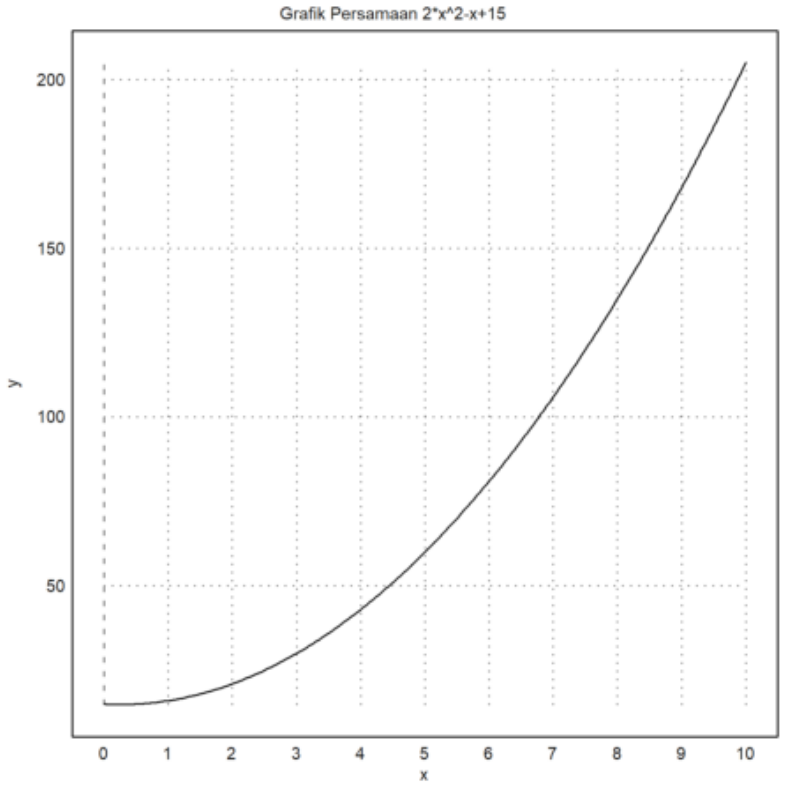


No.5

$$2x^2 - x + 15$$

Buat grafik dengan persamaan diatas dengan judul ”Grafik Persamaan 2*x^2-x+15”, label di sumbu x dan y serta gunakan rentan 0 hingga 10

```
>plot2d("2*x^2-x+15",0,10,title="Grafik Persamaan 2*x^2-x+15",yl="y",xl="x"):
```



Penjelasan: Maka kurva diatas berbentuk melengkung keatas dari titik x 0 hingga x 10

Rujukan

Lengkap Fungsi plot2d()

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

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

Parameters
x,y : equations, functions or data vectors
a,b,c,d : Plot area (default a=-2,b=2)
r : if r is set, then a=cx-r, b=cx+r, c=cy-r, d=cy+r

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

xmin,xmax : range of the parameter for curves
auto : Determine y-range automatically (default)
square : if true, try to keep square x-y-ranges
n : number of intervals (default is adaptive)
grid : 0 = no grid and labels,

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

frame : 0 = no frame
framecolor: color of the frame and the grid
margin : number between 0 and 0.4 for the margin around the plot
color : Color of curves. If this is a vector of colors,

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

thickness : line thickness for curves

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

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

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

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

```
value
```

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

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

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

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

own :

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

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

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

cgrid :

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

Overview

The function can plot

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

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

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

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