

TUGAS APLIKASI KOMPUTER

"Software Euler Math Toolbox"

Tugas ini disusun untuk memenuhi tugas Mata Kuliah Aplikasi Komputer

Dosen Pengampu: Drs. Sahid M.Sc. dan Thesa Adi Saputra Yusri M.Cs.



Disusun oleh:
RAIHAN HIJRI FIRJATULLAH
22305141034
MATEMATIKA E 2022

PENDIDIKAN MATEMATIKA
DEPARTEMEN PENDIDIKAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS NEGERI YOGYAKARTA
2023

DAFTAR ISI

1 Pengenalan Software Euler Math Toolbox	2
2 Penggunaan Software EMT untuk Aljabar	17
3 Latihan Soal dari BUKU ALJABAR dengan EMT	79
4 Penggunaan Software EMT untuk Plot 2D	105
5 Penggunaan Software EMT untuk Plot 3D	193
6 Penggunaan Software EMT untuk Kalkulus	223
7 Penggunaan Software EMT untuk Geometri	267
8 Penggunaan Software EMT untuk Statistika	308

BAB 1

PENGENALAN SOFTWARE EULER MATH TOOLBOX

Pendahuluan dan Pengenalan Cara Kerja EMT

Selamat datang! Ini adalah pengantar pertama ke Euler Math Toolbox (disingkat EMT atau Euler). EMT adalah sistem terintegrasi yang merupakan perpaduan kernel numerik Euler dan program komputer aljabar Maxima.

- Bagian numerik, GUI, dan komunikasi dengan Maxima telah dikembangkan oleh R. Grothmann, seorang profesor matematika di Universitas Eichstätt, Jerman. Banyak algoritma numerik dan pustaka software open source yang digunakan di dalamnya.

- Maxima adalah program open source yang matang dan sangat kaya untuk perhitungan simbolik dan aritmatika tak terbatas. Software ini dikelola oleh sekelompok pengembang di internet.

- Beberapa program lain (LaTeX, Povray, Tiny C Compiler, Python) dapat digunakan di Euler untuk memungkinkan perhitungan yang lebih cepat maupun tampilan atau grafik yang lebih baik.

Yang sedang Anda baca (jika dibaca di EMT) ini adalah berkas notebook di EMT. Notebook aslinya bawaan EMT (dalam bahasa Inggris) dapat dibuka melalui menu File, kemudian pilih "Open Tutorials and Examples", lalu pilih file "00 First Steps.en". Perhatikan, file notebook EMT memiliki ekstensi ".en". Melalui notebook ini Anda akan belajar menggunakan software Euler untuk menyelesaikan berbagai masalah matematika.

Panduan ini ditulis dengan Euler dalam bentuk notebook Euler, yang berisi teks (deskriptif), baris-baris perintah, tampilan hasil perintah (numerik, ekspresi matematika, atau gambar/plot), dan gambar yang disisipkan dari file gambar.

Untuk menambah jendela EMT, Anda dapat menekan [F11]. EMT akan menampilkan jendela grafik di layar desktop Anda. Tekan [F11] lagi untuk kembali ke tata letak favorit Anda. Tata letak disimpan untuk sesi berikutnya.

Anda juga dapat menggunakan [Ctrl]+[G] untuk menyembunyikan jendela grafik. Selanjutnya Anda dapat beralih antara grafik dan teks dengan tombol [TAB].

Seperti yang Anda baca, notebook ini berisi tulisan (teks) berwarna hijau, yang dapat Anda edit dengan meng-klik kanan teks atau tekan menu Edit -> Edit Comment atau tekan [F5], dan juga baris perintah EMT yang ditandai dengan ">" dan berwarna merah. Anda dapat menyisipkan baris perintah baru dengan cara menekan tiga tombol bersamaan: [Shift]+[Ctrl]+[Enter].

Komentar (Teks Uraian)

Komentar atau teks penjelasan dapat berisi beberapa "markup" dengan sintaks sebagai berikut.

- * Judul
- ** Sub-Judul
- latex: $F(x) = \int_a^x f(t) dt$
- mathjax: $\frac{x^2-1}{x-1} = x+1$
- maxima: 'integrate(x^3, x) = integrate(x^3, x) + "C"
- http://www.euler-math-toolbox.de
- See: http://www.google.de | Google
- image: IMAGE HATI.png
- ---

Hasil sintaks-sintaks di atas (tanpa diawali tanda strip) adalah sebagai berikut.

Judul

Sub-Judul

$$F(x) = \int_a^x f(t) dt$$

$$\frac{x^2 - 1}{x - 1} = x + 1$$

maxima: 'integrate(x^3, x) = integrate(x^3, x) + "C"

http://www.euler-math-toolbox.de

See: http://www.google.de | Google

image: IMAGE HATI.png

Gambar diambil dari folder images di tempat file notebook berada dan tidak dapat dibaca dari Web. Untuk "See:", tautan (URL)web lokal dapat digunakan.

Paragraf terdiri atas satu baris panjang di editor. Pergantian baris akan memulai baris baru. Paragraf harus dipisahkan dengan baris kosong.

```
>/> baris perintah diawali dengan >, komentar (keterangan) diawali dengan //
```

Baris Perintah

Mari kita tunjukkan cara menggunakan EMT sebagai kalkulator yang sangat canggih.

EMT berorientasi pada baris perintah. Anda dapat menuliskan satu atau lebih perintah dalam satu baris perintah. Setiap perintah harus diakhiri dengan koma atau titik koma.

- Titik koma menyembunyikan output (hasil) dari perintah.
- Sebuah koma mencetak hasilnya.
- Setelah perintah terakhir, koma diasumsikan secara otomatis (boleh tidak ditulis).

Dalam contoh berikut, kita mendefinisikan variabel r yang diberi nilai 1,25. Output dari definisi ini adalah nilai variabel. Tetapi karena tanda titik koma, nilai ini tidak ditampilkan. Pada kedua perintah di belakangnya, hasil kedua perhitungan tersebut ditampilkan.

```
>r=1.25; pi*r^2, 2*pi*r
```

4.90873852123
7.85398163397

Latihan untuk Anda

- Sisipkan beberapa baris perintah baru
- Tulis perintah-perintah baru untuk melakukan suatu perhitungan yang Anda inginkan, boleh menggunakan variabel, boleh tanpa variabel.

JAWABAN

NAMA : RAIHAN HIJRI FIRJATULLAH

KELAS : MATEMATIKA E

NIM : 22305141034

- PERINTAH LATIHAN :

```
>A=3; B=4; C=5; A*B*C, A+B+C
```

60
12

SATUAN

Beberapa catatan yang harus Anda perhatikan tentang penulisan sintaks perintah EMT.

- Pastikan untuk menggunakan titik desimal, bukan koma desimal untuk bilangan!
- Gunakan * untuk perkalian dan ^ untuk eksponen (pangkat).
- Seperti biasa, * dan / bersifat lebih kuat daripada + atau -.
- ^ mengikat lebih kuat dari *, sehingga $\pi \cdot r^2$ merupakan rumus luas lingkaran.
- Jika perlu, Anda harus menambahkan tanda kurung, seperti pada 2^3 (2^3).

Perintah $r = 1.25$ adalah menyimpan nilai ke variabel di EMT. Anda juga dapat menulis $r := 1.25$ jika mau. Anda dapat menggunakan spasi sesuka Anda.

Anda juga dapat mengakhiri baris perintah dengan komentar yang diawali dengan dua garis miring (//).

```
>r := 1.25 // Komentar: Menggunakan := sebagai ganti =
```

1.25

Argumen atau input untuk fungsi ditulis di dalam tanda kurung.

```
>sin(45°), cos(pi), log(sqrt(E))
```

0.707106781187
-1
0.5

Seperti yang Anda lihat, fungsi trigonometri bekerja dengan radian, dan derajat dapat diubah dengan \circ . Jika keyboard Anda tidak memiliki karakter derajat tekan [F7], atau gunakan fungsi deg() untuk mengonversi. EMT menyediakan banyak sekali fungsi dan operator matematika. Hampir semua fungsi matematika sudah tersedia di EMT. Anda dapat melihat daftar lengkap fungsi-fungsi matematika di EMT pada berkas Referensi (klik menu Help -> Reference)

Untuk membuat rangkaian komputasi lebih mudah, Anda dapat merujuk ke hasil sebelumnya dengan "%". Cara ini sebaiknya hanya digunakan untuk merujuk hasil perhitungan dalam baris perintah yang sama.

```
>(sqrt(5)+1)/2, %^2-%+1 // Memeriksa solusi x^2-x+1=0
```

1.61803398875

2

Latihan untuk Anda

- Buka berkas Reference dan baca fungsi-fungsi matematika yang tersedia di EMT.
- Sisipkan beberapa baris perintah baru.
- Lakukan contoh-contoh perhitungan menggunakan fungsi-fungsi matematika di EMT.

JAWABAN

NAMA : RAIHAN HIJRI FIRJATULLAH

KELAS : MATEMATIKA E

NIM : 22305141034

CONTOH PERHITUNGAN :

```
>sin(90°)/(sqrt(4)) + cos(90°) , cos(60°)/(sqrt(4))
```

0.5

0.25

Satuan

EMT dapat mengubah unit satuan menjadi sistem standar internasional (SI). Tambahkan satuan di belakang angka untuk konversi sederhana.

```
>1miles // 1 mil = 1609,344 m
```

1609.344

Beberapa satuan yang sudah dikenal di dalam EMT adalah sebagai berikut. Semua unit diakhiri dengan tanda dolar (\$), namun boleh tidak perlu ditulis dengan mengaktifkan easyunits.

```
kilometer$:=1000;  
km$:=kilometer$;  
cm$:=0.01;  
mm$:=0.001;  
minute$:=60;  
min$:=minute$;  
minutes$:=minute$;  
hour$:=60*minute$;  
h$:=hour$;  
hours$:=hour$;  
day$:=24*hour$;  
days$:=day$;  
d$:=day$;  
year$:=365.2425*day$;  
years$:=year$;  
y$:=year$;  
inch$:=0.0254;  
in$:=inch$;  
feet$:=12*inch$;  
foot$:=feet$;  
ft$:=feet$;  
yard$:=3*feet$;  
yards$:=yard$;  
yd$:=yard$;  
mile$:=1760*yard$;  
miles$:=mile$;  
kg$:=1;  
sec$:=1;  
ha$:=10000;  
Ar$:=100;  
Tagwerk$:=3408;  
Acre$:=4046.8564224;  
pt$:=0.376mm;
```

Untuk konversi ke dan antar unit, EMT menggunakan operator khusus, yakni ->.

```
>4km -> miles, 4inch -> " mm"
```

```
2.48548476895  
101.6 mm
```

Format Tampilan Nilai

Akurasi internal untuk nilai bilangan di EMT adalah standar IEEE, sekitar 16 digit desimal. Aslinya, EMT tidak mencetak semua digit suatu bilangan. Ini untuk menghemat tempat dan agar terlihat lebih baik. Untuk mengatramilan satu bilangan, operator berikut dapat digunakan.

```
>pi
```

```
3.14159265359
```

```
>longest pi
```

```
3.141592653589793
```

```
>long pi
```

```
3.14159265359
```

```
>short pi
```

```
3.1416
```

```
>shortest pi
```

```
3.1
```

```
>fraction pi
```

```
312689/99532
```

```
>short 1200*1.03^10, long E, longest pi
```

```
1612.7
```

```
2.71828182846
```

```
3.141592653589793
```

Format aslinya untuk menampilkan nilai menggunakan sekitar 10 digit. Format tampilan nilai dapat diatur secara global atau hanya untuk satu nilai.

Anda dapat mengganti format tampilan bilangan untuk semua perintah selanjutnya. Untuk mengembalikan ke format aslinya dapat digunakan perintah "deformat" atau "reset".

```
>longestformat; pi, deformat; pi
```

```
3.141592653589793
```

```
3.14159265359
```

Kernel numerik EMT bekerja dengan bilangan titik mengambang (floating point) dalam presisi ganda IEEE (berbeda dengan bagian simbolik EMT). Hasil numerik dapat ditampilkan dalam bentuk pecahan.

```
>1/7+1/4, fraction %
```

```
0.392857142857
```

```
11/28
```

Perintah Multibaris

Perintah multi-baris membentang di beberapa baris yang terhubung dengan "..." di setiap akhir baris, kecuali baris terakhir. Untuk menghasilkan tanda pindah baris tersebut, gunakan tombol [Ctrl]+[Enter]. Ini akan menyambung perintah ke baris berikutnya dan menambahkan "..." di akhir baris sebelumnya. Untuk menggabungkan suatu baris ke baris sebelumnya, gunakan [Ctrl]+[Backspace].

Contoh perintah multi-baris berikut dapat dijalankan setiap kali kursor berada di salah satu barisnya. Ini juga menunjukkan bahwa ... harus berada di akhir suatu baris meskipun baris tersebut memuat komentar.

```
>a=4; b=15; c=2; // menyelesaikan a*x^2+b*x+c=0 secara manual ...
>D=sqrt (b^2/(a^2*4)-c/a); ...
>-b/(2*a) + D, ...
>-b/(2*a) - D
```

```
-0.138444501319
-3.61155549868
```

Menampilkan Daftar Variabel

Untuk menampilkan semua variabel yang sudah pernah Anda definisikan sebelumnya (dan dapat dilihat kembali nilainya), gunakan perintah "listvar".

```
>listvar
```

A	3
r	1.25
B	4
C	5
a	4
b	15
c	2
D	1.73655549868123

Perintah listvar hanya menampilkan variabel buatan pengguna. Dimungkinkan untuk menampilkan variabel lain, dengan menambahkan string termuat di dalam nama variabel yang diinginkan.

Perlu Anda perhatikan, bahwa EMT membedakan huruf besar dan huruf kecil. Jadi variabel "d" berbeda dengan variabel "D".

Contoh berikut ini menampilkan semua unit yang diakhiri dengan "m" dengan mencari semua variabel yang berisi "m\$".

```
>listvar m$
```

km\$	1000
cm\$	0.01
mm\$	0.001
nm\$	1853.24496
gram\$	0.001
m\$	1
hquantum\$	6.62606957e-34
atm\$	101325

Untuk menghapus variabel tanpa harus memulai ulang EMT gunakan perintah "remvalue".

```
>remvalue a,b,c,D  
>D
```

Variable D not found!

Error in:

D ...
^

Menampilkan Panduan

Untuk mendapatkan panduan tentang penggunaan perintah atau fungsi di EMT, buka jendela panduan dengan menekan [F1] dan cari fungsinya. Anda juga dapat mengklik dua kali pada fungsi yang tertulis di baris perintah atau di teks untuk membuka jendela panduan.

Coba klik dua kali pada perintah "intrandom" berikut ini!

```
>intrandom(10,6)
```

[4, 2, 6, 2, 4, 2, 3, 2, 2, 6]

Di jendela panduan, Anda dapat mengklik kata apa saja untuk menemukan referensi atau fungsi.

Misalnya, coba klik kata "random" di jendela panduan. Kata tersebut boleh ada dalam teks atau di bagian "See:" pada panduan. Anda akan menemukan penjelasan fungsi "random", untuk menghasilkan bilangan acak berdistribusi uniform antara 0,0 dan 1,0. Dari panduan untuk "random" Anda dapat menampilkan panduan untuk fungsi "normal", dll.

```
>random(10)
```

[0.270906, 0.704419, 0.217693, 0.445363, 0.308411, 0.914541, 0.193585, 0.463387,

```
>normal(10)
```

[-0.495418, 1.6463, -0.390056, -1.98151, 3.44132, 0.308178, -0.733427, -0.526167,

Matriks dan Vektor

EMT merupakan suatu aplikasi matematika yang mengerti "bahasa matriks". Artinya, EMT menggunakan vektor dan matriks untuk perhitungan-perhitungan tingkat lanjut. Suatu vektor atau matriks dapat didefinisikan dengan tanda kurung siku. Elemen-elemennya dituliskan di dalam tanda kurung siku, antar elemen dalam satu baris dipisahkan oleh koma(,), antar baris dipisahkan oleh titik koma (;).

Vektor dan matriks dapat diberi nama seperti variabel biasa.

```
>v=[4,5,6,3,2,1]
```

[4, 5, 6, 3, 2, 1]

```
>A=[1,2,3;4,5,6;7,8,9]
```

1	2	3
4	5	6
7	8	9

Karena EMT mengerti bahasa matriks, EMT memiliki kemampuan yang sangat canggih untuk melakukan perhitungan matematis untuk masalah-masalah aljabar linier, statistika, dan optimisasi.

Vektor juga dapat didefinisikan dengan menggunakan rentang nilai dengan interval tertentu menggunakan tanda titik dua (:), seperti contoh berikut ini.

```
>c=1:5
```

```
[1, 2, 3, 4, 5]
```

```
>w=0:0.1:1
```

```
[0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
```

```
>mean(w^2)
```

```
0.35
```

Bilangan Kompleks

EMT juga dapat menggunakan bilangan kompleks. Tersedia banyak fungsi untuk bilangan kompleks di EMT. Bilangan imaginer

$$i = \sqrt{-1}$$

dituliskan dengan huruf I (huruf besar I), namun akan ditampilkan dengan huruf i (i kecil).

```
re(x) : bagian riil pada bilangan kompleks x.  
im(x) : bagian imaginer pada bilangan kompleks x.  
complex(x) : mengubah bilangan riil x menjadi bilangan kompleks.  
conj(x) : Konjugat untuk bilangan bilangan kompleks x.  
arg(x) : argumen (sudut dalam radian) bilangan kompleks x.  
real(x) : mengubah x menjadi bilangan riil.
```

Apabila bagian imaginer x terlalu besar, hasilnya akan menampilkan pesan kesalahan.

```
>sqrt(-1) // Error!  
>sqrt(complex(-1))
```

```
>z=2+3*I, re(z), im(z), conj(z), arg(z), deg(arg(z)), deg(arctan(3/2))
```

```
2+3i  
2  
3  
2-3i  
0.982793723247  
56.309932474  
56.309932474
```

```
>deg(arg(I)) // 90°
```

90

```
>sqrt(-1)
```

```
Floating point error!
Error in sqrt
Error in:
sqrt(-1) ...
^
```

```
>sqrt(complex(-1))
```

0+1i

EMT selalu menganggap semua hasil perhitungan berupa bilangan riil dan tidak akan secara otomatis mengubah ke bilangan kompleks.

Jadi akar kuadrat -1 akan menghasilkan kesalahan, tetapi akar kuadrat kompleks didefinisikan untuk bidang koordinat dengan cara seperti biasa. Untuk mengubah bilangan riil menjadi kompleks, Anda dapat menambahkan 0i atau menggunakan fungsi "complex".

```
>complex(-1), sqrt(%)
```

-1+0i
0+1i

Matematika Simbolik

EMT dapat melakukan perhitungan matematika simbolis (eksak) dengan bantuan software Maxima. Software Maxima otomatis sudah terpasang di komputer Anda ketika Anda memasang EMT. Meskipun demikian, Anda dapat juga memasang software Maxima tersendiri (yang terpisah dengan instalasi Maxima di EMT).

Pengguna Maxima yang sudah mahir harus memperhatikan bahwa terdapat sedikit perbedaan dalam sintaks antara sintaks asli Maxima dan sintaks ekspresi simbolik di EMT.

Untuk melakukan perhitungan matematika simbolis di EMT, awali perintah Maxima dengan tanda "&". Setiap ekspresi yang dimulai dengan "&" adalah ekspresi simbolis dan dikerjakan oleh Maxima.

```
>& (a+b)^2
```

$$(a + b)^2$$

```
>&expand((a+b)^2), &factor(x^2+5*x+6)
```

$$\frac{2}{a^2 + 2ab} + \frac{2}{b^2}$$

$$(x+2)(x+3)$$

```
>&solve(a*x^2+b*x+c,x) // rumus abc
```

$$[x = \frac{-\sqrt{b^2 - 4ac} - b}{2a}, x = \frac{\sqrt{b^2 - 4ac} - b}{2a}]$$

```
>&(a^2-b^2)/(a+b), &ratsimp(%) // ratsimp menyederhanakan bentuk pecahan
```

$$\frac{a^2 - b^2}{b^2 + a^2}$$

$$a^2 - b^2$$

```
>10! // nilai faktorial (modus EMT)
```

3628800

```
>&10! //nilai faktorial (simbolik dengan Maxima)
```

3628800

Untuk menggunakan perintah Maxima secara langsung (seperti perintah pada layar Maxima) awali perintahnya dengan tanda ":" pada baris perintah EMT. Sintaks Maxima disesuaikan dengan sintaks EMT (disebut "modus kompatibilitas").

```
>factor(1000) // mencari semua faktor 1000 (EMT)
```

[2, 2, 2, 5, 5, 5]

```
>::: factor(1000) // faktorisasi prima 1000 (dengan Maxima)
```

$$\begin{matrix} 3 & 3 \\ 2 & 5 \end{matrix}$$

```
>::: factor(20!)
```

$$\begin{matrix} 18 & 8 & 4 & 2 \\ 2 & 3 & 5 & 7 & 11 & 13 & 17 & 19 \end{matrix}$$

Jika Anda sudah mahir menggunakan Maxima, Anda dapat menggunakan sintaks asli perintah Maxima dengan menggunakan tanda ":::" untuk mengawali setiap perintah Maxima di EMT. Perhatikan, harus ada spasi antara ":::" dan perintahnya.

```
>::: binomial(5,2); // nilai C(5,2)
```

$$10$$

```
>::: binomial(m,4); // C(m,4)=m!/(4!(m-4)!)
```

$$\frac{(m - 3)(m - 2)(m - 1)m}{24}$$

```
>::: trigexpand(cos(x+y)); // rumus cos(x+y)=cos(x) cos(y)-sin(x)sin(y)
```

$$\cos(x) \cos(y) - \sin(x) \sin(y)$$

```
>::: trigexpand(sin(x+y));
```

$$\cos(x) \sin(y) + \sin(x) \cos(y)$$

```
>::: trigsimp(((1-sin(x)^2)*cos(x))/cos(x)^2+tan(x)*sec(x)^2) //menyederhanakan fungsi tri-
```

$$\frac{\sin(x)^4 + \cos(x)^4}{\cos(x)^3}$$

Untuk menyimpan ekspresi simbolik ke dalam suatu variabel digunakan tanda "&=".

```
>p1 &= (x^3+1) / (x+1)
```

$$\frac{x^3 + 1}{x + 1}$$

```
>&ratsimp(p1)
```

$$\frac{x^2 - x + 1}{x}$$

Untuk mensubstitusikan suatu nilai ke dalam variabel dapat digunakan perintah "with".

```
>&p1 with x=3 // (3^3+1) / (3+1)
```

7

```
>&p1 with x=a+b, &ratsimp(%) //substitusi dengan variabel baru
```

$$\frac{(b + a)^3 + 1}{b + a + 1}$$

$$\frac{b^2 + (2ab - 1)b^2 + a^2 - a + 1}{b^2}$$

```
>&diff(p1,x) //turunan p1 terhadap x
```

$$\begin{array}{r}
 & 2 & 3 \\
 3 & x & x + 1 \\
 \hline
 x + 1 & & 2 \\
 & (x + 1)
 \end{array}$$

```
>&integrate(p1,x) // integral p1 terhadap x
```

$$\begin{array}{r}
 & 3 & 2 \\
 2 & x - 3 & x + 6 & x \\
 \hline
 & & & 6
 \end{array}$$

Tampilan Matematika Simbolik dengan LaTeX

Anda dapat menampilkan hasil perhitungan simbolik secara lebih bagus menggunakan LaTeX. Untuk melakukannya, tambahkan tanda dolar (\$) di depan tanda & pada setiap perintah Maxima. Perhatikan, hal ini hanya dapat menghasilkan tampilan yang diinginkan apabila komputer Anda sudah terpasang software LaTeX.

```
>$& (a+b)^2
```

$$(a + b)^2$$

```
>$&expand( (a+b)^2), $&factor(x^2+5*x+6)
```

$$(x + 2) (x + 3)$$

```
>$&solve(a*x^2+b*x+c,x) // rumus abc
```

$$\left[x = \frac{-\sqrt{b^2 - 4ac} - b}{2a}, x = \frac{\sqrt{b^2 - 4ac} - b}{2a} \right]$$

```
>$&(a^2-b^2)/(a+b), $&ratsimp(%)
```

$$a - b$$

Selamat Belajar dan Berlatih!

Baik, itulah sekilas pengantar penggunaan software EMT. Masih banyak kemampuan EMT yang akan Anda pelajari dan praktikkan.

Sebagai latihan untuk memperlancar penggunaan perintah-perintah EMT yang sudah dijelaskan di atas, silakan Anda lakukan hal-hal sebagai berikut.

- Carilah soal-soal matematika dari buku-buku Matematika.
- Tambahkan beberapa baris perintah EMT pada notebook ini.
- Selesaikan soal-soal matematika tersebut dengan menggunakan EMT.

Pilih soal-soal yang sesuai dengan perintah-perintah yang sudah dijelaskan dan dicontohkan di atas.

JAWABAN

NAMA : RAIHAN HIJRI FIRJATULLAH

KELAS : MATEMATIKA E

NIM : 22305141034

- CONTOH SOAL-SOAL MATEMATIKA dari BUKU KALKULUS JILID 1:

```
>function f(x) &= sin(x)*tan(x)
```

sin(x) tan(x)

```
>function f(x) &= integrate(f(x),x)
```

$$\frac{\log(\sin(x) + 1)}{2} - \frac{\log(\sin(x) - 1)}{2} - \sin(x)$$

BAB 2

PENGGUNAAN SOFTWARE EMT UNTUK ALJABAR

TUGAS INDIVIDU 1

EMT untuk Perhitungan Aljabar

Pada notebook ini Anda belajar menggunakan EMT untuk melakukan berbagai perhitungan terkait dengan materi atau topik dalam Aljabar. Kegiatan yang harus Anda lakukan adalah sebagai berikut:

- Membaca secara cermat dan teliti notebook ini;
- Menerjemahkan teks bahasa Inggris ke bahasa Indonesia;
- Mencoba contoh-contoh perhitungan (perintah EMT) dengan cara meng ENTER setiap perintah EMT yang ada (pindahkan kursor ke baris perintah)
- Jika perlu Anda dapat memodifikasi perintah yang ada dan memberikan keterangan/penjelasan tambahan terkait hasilnya.
- Menyisipkan baris-baris perintah baru untuk mengerjakan soal-soal Aljabar dari file PDF yang saya berikan;
- Memberi catatan hasilnya.
- Jika perlu tuliskan soalnya pada teks notebook (menggunakan format LaTeX).
- Gunakan tampilan hasil semua perhitungan yang eksak atau simbolik dengan format LaTeX. (Seperti contoh-contoh pada notebook ini.)

Contoh pertama

Menyederhanakan bentuk aljabar:

$$6x^{-3}y^5 \times -7x^2y^{-9}$$

```
> $& 6*x^(-3)*y^5*-7*x^2*y^(-9)
```

$$-\frac{42}{x y^4}$$

Menyederhanakan fungsi :

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

```
> $& 2*y^2+2*x^2-3*y^2+2*x^2
```

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

Menjabarkan:

$$(6x^{-3} + y^5)(-7x^2 - y^{-9})$$

```
> $& showev (' expand( (6*x^(-3)+y^5)*(-7*x^2-y^(-9)) ) )
```

$$\text{expand} \left(\left(-\frac{1}{y^9} - 7x^2 \right) \left(y^5 + \frac{6}{x^3} \right) \right) = -7x^2y^5 - \frac{1}{y^4} - \frac{6}{x^3y^9} - \frac{42}{x}$$

Baris Perintah

Baris perintah Euler terdiri dari satu atau beberapa perintah Euler diikuti dengan titik koma ";" atau koma ",". Titik koma mencegah pencetakan hasil. Koma setelah perintah terakhir dapat dihilangkan.

Baris perintah berikut hanya akan mencetak hasil ekspresi, bukan tugas atau perintah format.

```
>r:=4; h:=4; pi*r^2*h/3
```

67.0206432766

Perintah harus dipisahkan dengan yang kosong. Baris perintah berikut mencetak dua hasilnya.

```
>pi*2*r*h, %+2*pi*r*h // Ingat tanda % menyatakan hasil perhitungan terakhir sebelumnya
```

100.530964915
201.06192983

Baris perintah dieksekusi dalam urutan yang ditekan pengguna kembali. Jadi Anda mendapatkan nilai baru setiap kali Anda menjalankan baris kedua.

```
>x := 2;  
>x := cos(x) // nilai cosinus (x dalam radian)
```

-0.416146836547

```
>x := cos(x)
```

0.914653325852

Jika dua garis terhubung dengan "..." kedua garis akan selalu dieksekusi secara bersamaan.

```
>x := 1.5; ...  
>x := (x+2/x)/2, x := (x+2/x)/2, x := (x+2/x)/2,
```

1.41666666667

1.41421568627

1.41421356237

Ini juga merupakan cara yang baik untuk menyebarkan perintah panjang pada dua atau lebih baris. Anda dapat menekan Ctrl+Return untuk membagi garis menjadi dua pada posisi kursor saat ini, atau Ctrl+Back untuk menggabungkan garis.

Untuk melipat semua multi-garis tekan Ctrl + L. Kemudian garis-garis berikutnya hanya akan terlihat, jika salah satunya memiliki fokus. Untuk melipat satu multi-baris, mulailah baris pertama dengan "%+".

```
>%+ x=4+5; ...
```

Garis yang dimulai dengan %% tidak akan terlihat sama sekali.

81

```
>
```

Euler mendukung loop di baris perintah, selama mereka masuk ke dalam satu baris atau multi-baris. Dalam program, pembatasan ini tidak berlaku, tentu saja. Untuk informasi lebih lanjut lihat pengantar berikut.

```
>x=4; for i=1 to 10; x := (x+2/x)/2, end; // menghitung akar 2
```

2.25
1.56944444444
1.42189036382
1.41423428594
1.41421356252
1.41421356237
1.41421356237
1.41421356237
1.41421356237
1.41421356237

Tidak apa-apa untuk menggunakan multi-line. Pastikan baris diakhiri dengan "...".

```
>x := 1.5; // comments go here before the ...
>repeat xnew:=(x+2/x)/2; until xnew~≈x; ...
>    x := xnew; ...
>end; ...
>x,
```

1.41421356237

Struktur bersyarat juga berfungsi.

```
>if E^pi>pi^E; then "Thought so!", endif;
```

Thought so!

Saat Anda menjalankan perintah, kursor dapat berada di posisi mana pun di baris perintah. Anda dapat kembali ke perintah sebelumnya atau melompat ke perintah berikutnya dengan tombol panah. Atau Anda dapat mengklik ke bagian komentar di atas perintah untuk menuju ke perintah.

Saat Anda menggerakkan kursor di sepanjang garis, pasangan tanda kurung atau kurung buka dan tutup akan disorot. Juga, perhatikan baris status. Setelah kurung buka fungsi `sqrt()`, baris status akan menampilkan teks bantuan untuk fungsi tersebut. Jalankan perintah dengan tombol kembali.

```
>sqrt(sin(90°)/cos(20°))
```

1.03158992457

Untuk melihat bantuan untuk perintah terbaru, buka jendela bantuan dengan F1. Di sana, Anda dapat memasukkan teks untuk dicari. Pada baris kosong, bantuan untuk jendela bantuan akan ditampilkan. Anda dapat menekan escape untuk menghapus garis, atau untuk menutup jendela bantuan.

Anda dapat mengklik dua kali pada perintah apa pun untuk membuka bantuan untuk perintah ini. Coba klik dua kali perintah `exp` di bawah ini di baris perintah.

```
>exp(log(2.4))
```

2.4

Anda dapat menyalin dan menempel di Euler ke. Gunakan Ctrl-C dan Ctrl-V untuk ini. Untuk menandai teks, seret mouse atau gunakan shift bersama dengan tombol kursor apa pun. Selain itu, Anda dapat menyalin tanda kurung yang disorot.

Sintaks Dasar

Euler tahu fungsi matematika biasa. Seperti yang Anda lihat di atas, fungsi trigonometri bekerja dalam radian atau derajat. Untuk mengonversi ke derajat, tambahkan simbol derajat (dengan tombol F7) ke nilainya, atau gunakan fungsi rad(x). Fungsi akar kuadrat disebut kuadrat dalam Euler. Tentu saja, $x^{(1/2)}$ juga dimungkinkan.

Untuk menyetel variabel, gunakan "=" atau "=". Demi kejelasan, pengantar ini menggunakan bentuk yang terakhir. Spasi tidak masalah. Tapi ruang antara perintah diharapkan.

Beberapa perintah dalam satu baris dipisahkan dengan "," atau ";". Titik koma menekan output dari perintah. Di akhir baris perintah "," diasumsikan, jika ";" hilang.

```
>g:=9.81; t:=2.5; 1/2*g*t^2
```

30.65625

EMT menggunakan sintaks pemrograman untuk ekspresi. Memasuki

$$e^2 \cdot \left(\frac{1}{3 + 4 \log(0.6)} + \frac{1}{7} \right)$$

Anda harus mengatur tanda kurung yang benar dan menggunakan / untuk pecahan. Perhatikan tanda kurung yang disorot untuk bantuan. Perhatikan bahwa konstanta Euler e diberi nama E dalam EMT.

```
>E^2*(1/(3+4*log(0.6))+1/7)
```

8.77908249441

Untuk menghitung ekspresi rumit seperti

$$\left(\frac{\frac{1}{7} + \frac{1}{8} + 2}{\frac{1}{3} + \frac{1}{2}} \right)^2 \pi$$

Anda harus memasukkannya dalam bentuk baris.

```
>((1/7 + 1/8 + 2) / (1/3 + 1/2))^2 * pi
```

23.2671801626

Letakkan tanda kurung dengan hati-hati di sekitar sub-ekspresi yang perlu dihitung terlebih dahulu. EMT membantu Anda dengan menyorot ekspresi bahwa braket penutup selesai. Anda juga harus memasukkan nama "pi" untuk huruf Yunani pi.

Hasil dari perhitungan ini adalah bilangan floating point. Secara default dicetak dengan akurasi sekitar 12 digit. Di baris perintah berikut, kita juga belajar bagaimana kita bisa merujuk ke hasil sebelumnya dalam baris yang sama.

```
>1/3+1/7, fraction %
```

0.47619047619

10/21

Perintah Euler dapat berupa ekspresi atau perintah primitif. Ekspresi terbuat dari operator dan fungsi. Jika perlu, itu harus berisi tanda kurung untuk memaksa urutan eksekusi yang benar. Jika ragu, memasang braket adalah ide yang bagus. Perhatikan bahwa EMT menunjukkan tanda kurung buka dan tutup saat mengedit baris perintah.

```
> (cos(pi/4)+1)^3*(sin(pi/4)+1)^2
```

14.4978445072

Operator numerik Euler meliputi

- + unary atau operator plus
- unary atau operator minus
- *, /
- produk matriks
- a^b daya untuk positif a atau bilangan bulat b ($a^{**}b$ juga berfungsi)
- $n!$ operator faktorial

dan masih banyak lagi.

Berikut adalah beberapa fungsi yang mungkin Anda butuhkan. Ada banyak lagi.

```
sin,cos,tan,atan,asin,acos,rad,deg  
log,exp,log10,sqrt,logbase  
bin,logbin,logfac,mod,lantai,ceil,bulat,abs,tanda  
conj,re,im,arg,conj,nyata,kompleks  
beta,betai,gamma,complexgamma,ellrf,ellf,ellrd,elle  
bitand, bitor, bitxor, bitnot
```

Beberapa perintah memiliki alias, mis. Untuk log.

```
>ln(E^2), arctan(tan(0.5)), logbase(30,30)
```

2
0.5
1

```
>sin(30°)
```

0.5

Pastikan untuk menggunakan tanda kurung (kurung bulat), setiap kali ada keraguan tentang urutan eksekusi! Berikut ini tidak sama dengan $(2^3)^4$, yang merupakan default untuk 2^3^4 di EMT (beberapa sistem numerik melakukannya dengan cara lain).

```
>2^3^4, (2^3)^4, 2^(3^4)
```

2.41785163923e+24
4096
2.41785163923e+24

Bilangan Asli

Tipe data utama dalam Euler adalah bilangan real. Real direpresentasikan dalam format IEEE dengan akurasi sekitar 16 digit desimal.

```
>longest 1/3
```

```
0.3333333333333333
```

```
>longest 1/9
```

```
0.1111111111111111
```

Representasi ganda internal membutuhkan 8 byte.

```
>printdual(1/3)
```

```
1.01010101010101010101010101010101010101010101010101010101010101*2^-2
```

```
>printhex(1/3)
```

```
5.555555555554*16^-1
```

```
>printdual(1/9)
```

```
1.1100011100011100011100011100011100011100011100011100011100*2^-4
```

```
>printhex(1/9)
```

```
1.C71C71C71C71C*16^-1
```

```
>
```

String

Sebuah string dalam Euler didefinisikan dengan "...".

```
>"A string can contain anything."
```

```
A string can contain anything.
```

String dapat digabungkan dengan | atau dengan +. Ini juga berfungsi dengan angka, yang dikonversi menjadi string dalam kasus itu.

```
>"The area of the circle with radius " + 2 + " cm is " + pi*4 + " cm^2."
```

The area of the circle with radius 2 cm is 12.5663706144 cm².

Fungsi print juga mengonversi angka menjadi string. Ini dapat mengambil sejumlah digit dan sejumlah tempat (0 untuk keluaran padat), dan secara optimal satu unit.

```
>"Golden Ratio : " + print((1+sqrt(5))/2,5,0)
```

Golden Ratio : 1.61803

Ada string khusus tidak ada, yang tidak dicetak. Itu dikembalikan oleh beberapa fungsi, ketika hasilnya tidak masalah. (Ini dikembalikan secara otomatis, jika fungsi tidak memiliki pernyataan pengembalian.)

```
>none
```

Untuk mengonversi string menjadi angka, cukup evaluasi saja. Ini juga berfungsi untuk ekspresi (lihat di bawah).

```
>"1234.5"()
```

1234.5

Untuk mendefinisikan vektor string, gunakan notasi vektor [...].

```
>v := ["affe", "charlie", "bravo"]
```

```
affe  
charlie  
bravo
```

Vektor string kosong dilambangkan dengan [none]. Vektor string dapat digabungkan.

```
>w := [none]; w | v | v
```

```
affe  
charlie  
bravo  
affe  
charlie  
bravo
```

String dapat berisi karakter Unicode. Secara internal, string ini berisi kode UTF-8. Untuk menghasilkan string seperti itu, gunakan u"..." dan salah satu entitas HTML.

String Unicode dapat digabungkan seperti string lainnya.

```
>u"&alpha; = " + 45 + u"&deg;" // pdfLaTeX mungkin gagal menampilkan secara benar  
= 45°
```

I

Dalam komentar, entitas yang sama seperti alpha;, beta; dll dapat digunakan. Ini mungkin alternatif cepat untuk Lateks. (Lebih detail di komentar di bawah).

Ada beberapa fungsi untuk membuat atau menganalisis string unicode. Fungsi strtochar() akan mengenali string Unicode, dan menerjemahkannya dengan benar.

```
>v=strtochar(u"&Auml; is a German letter")
```

```
[196, 32, 105, 115, 32, 97, 32, 71, 101, 114, 109, 97, 110, 32, 108, 101,
```

Hasilnya adalah vektor angka Unicode. Fungsi kebalikannya adalah chartoutf().

```
>v[1]=strtochar(u"&Uuml;") [1]; chartoutf(v)
```

Ü is a German letter

Fungsi utf() dapat menerjemahkan string dengan entitas dalam variabel menjadi string Unicode.

```
>s="We have &alpha;=&beta; ."; utf(s) // pdfLaTeX mungkin gagal menampilkan secara benar
```

We have =.

Dimungkinkan juga untuk menggunakan entitas numerik.

```
>u"#196;hnliches"
```

Ähnliches

Nilai Boolean

Nilai Boolean direpresentasikan dengan 1=true atau 0=false dalam Euler. String dapat dibandingkan, seperti halnya angka.

```
>2<1, "apel"<"banana"
```

```
0  
1
```

"dan" adalah operator "&&" dan "atau" adalah operator "||", seperti dalam bahasa C. (Kata-kata "dan" dan "atau" hanya dapat digunakan dalam kondisi untuk "jika".)

```
>2<E && E<3
```

1

Operator Boolean mematuhi aturan bahasa matriks.

```
>(1:10)>5, nonzeros(%)
```

```
[0, 0, 0, 0, 0, 1, 1, 1, 1, 1]  
[6, 7, 8, 9, 10]
```

Anda dapat menggunakan fungsi bukan nol() untuk mengekstrak elemen tertentu dari vektor. Dalam contoh, kami menggunakan isprime bersyarat(n).

```
>N=2|3:2:99 // N berisi elemen 2 dan bilangan2 ganjil dari 3 s.d. 99
```

```
[2, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37  
53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 75, 77, 79, 81, 83, 85, 87,
```

```
>N[nonzeros(isprime(N))] //pilih anggota2 N yang prima
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 63, 67, 69, 71, 73, 75, 77, 79, 81, 83, 85, 87, 91, 97]
```

Format Keluaran

Format output default EMT mencetak 12 digit. Untuk memastikan bahwa kami melihat default, kami mengatur ulang format.

```
>defformat; pi
```

3.14159265359

Secara internal, EMT menggunakan standar IEEE untuk bilangan ganda dengan sekitar 16 digit desimal. Untuk melihat jumlah digit penuh, gunakan perintah "format terpanjang", atau kita gunakan operator "terpanjang" untuk menampilkan hasil dalam format terpanjang.

```
>longest pi
```

3.141592653589793

Berikut adalah representasi heksadesimal internal dari bilangan ganda.

```
>printhex(pi)
```

3.243F6A8885A30*16^0

Format output dapat diubah secara permanen dengan perintah format.

```
>format(12,5); 1/3, pi, sin(1)
```

```
0.33333  
3.14159  
0.84147
```

Standarnya adalah format (12).

```
>format(12); 1/3
```

```
0.333333333333
```

Fungsi seperti "shortestformat", "shortformat", "longformat" bekerja untuk vektor dengan cara berikut.

```
>shortestformat; random(3,8)
```

```
0.66    0.2    0.89    0.28    0.53    0.31    0.44    0.3  
0.28    0.88    0.27    0.7     0.22    0.45    0.31    0.91  
0.19    0.46    0.095   0.6     0.43    0.73    0.47    0.32
```

Format default untuk skalar adalah format (12). Tapi ini bisa diubah.

```
>setscalarformat(5); pi
```

```
3.1416
```

Fungsi "format terpanjang" mengatur format skalar juga.

```
>longestformat; pi
```

```
3.141592653589793
```

Untuk referensi, berikut adalah daftar format output yang paling penting.

```
format terpendek format pendek format panjang, format terpanjang  
format(panjang,digit) format baik(panjang)  
fracformat (panjang)  
mengubah bentuk
```

Akurasi internal EMT adalah sekitar 16 tempat desimal, yang merupakan standar IEEE. Angka disimpan dalam format internal ini.

Tetapi format output EMT dapat diatur dengan cara yang fleksibel.

```
>longestformat; pi,
```

```
3.141592653589793
```

```
>format(10,5); pi
```

3.14159

Standarnya adalah defformat().

```
>defformat; // default
```

Ada operator pendek yang hanya mencetak satu nilai. Operator "terpanjang" akan mencetak semua digit angka yang valid.

```
>longest pi^2/2
```

4.934802200544679

Ada juga operator pendek untuk mencetak hasil dalam format pecahan. Kami sudah menggunakannya di atas.

```
>fraction 1+1/2+1/3+1/4
```

25/12

Karena format internal menggunakan cara biner untuk menyimpan angka, nilai 0,1 tidak akan direpresentasikan dengan tepat. Kesalahan bertambah sedikit, seperti yang Anda lihat dalam perhitungan berikut.

```
>longest 0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1-1
```

-1.110223024625157e-16

Tetapi dengan "format panjang" default Anda tidak akan melihat ini. Untuk kenyamanan, output dari angka yang sangat kecil adalah 0.

```
>0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1-1
```

0

Ekspresi

String atau nama dapat digunakan untuk menyimpan ekspresi matematika, yang dapat dievaluasi oleh EMT. Untuk ini, gunakan tanda kurung setelah ekspresi. Jika Anda bermaksud menggunakan string sebagai ekspresi, gunakan konvensi untuk menamakannya "fx" atau "fxy" dll. Ekspresi lebih diutamakan daripada fungsi. Variabel global dapat digunakan dalam evaluasi.

```
>r:=2; fx:="pi*r^2"; longest fx()
```

12.56637061435917

Parameter ditetapkan ke x , y , dan z dalam urutan itu. Parameter tambahan dapat ditambahkan menggunakan parameter yang ditetapkan.

```
>fx := "a*sin(x)^2"; fx(5, a=-1)
```

-0.919535764538

Perhatikan bahwa ekspresi akan selalu menggunakan variabel global, bahkan jika ada variabel dalam fungsi dengan nama yang sama. (Jika tidak, evaluasi ekspresi dalam fungsi dapat memberikan hasil yang sangat membingungkan bagi pengguna yang memanggil fungsi tersebut.)

```
>at := 4; function f(expr, x, at) := expr(x); ...
>f("at*x^2", 3, 5) // computes 4*3^2 not 5*3^2
```

36

Jika Anda ingin menggunakan nilai lain untuk "at" daripada nilai global, Anda perlu menambahkan "at=value".

```
>at := 4; function f(expr, x, a) := expr(x, at=a); ...
>f("at*x^2", 3, 5)
```

45

Untuk referensi, kami berkomentar bahwa koleksi panggilan (dibahas di tempat lain) dapat berisi ekspresi. Jadi kita bisa membuat contoh di atas sebagai berikut.

```
>at := 4; function f(expr, x) := expr(x); ...
>f({{"at*x^2", at=5}}, 3)
```

45

Ekspresi dalam x sering digunakan seperti fungsi.

Perhatikan bahwa mendefinisikan fungsi dengan nama yang sama seperti ekspresi simbolik global menghapus variabel ini untuk menghindari kebingungan antara ekspresi simbolik dan fungsi.

```
>f &= 6*x;
>function f(x) := 6*x;
>f(5)
```

30

```
>f &= 4*x;
>f(10)
```

40

```
>function f(x) := 9*x;  
>f(2)
```

18

Dengan cara konvensi, ekspresi simbolik atau numerik harus diberi nama fx, fxy dll. Skema penamaan ini tidak boleh digunakan untuk fungsi.

```
>fx &= diff(x^x,x); $&fx
```

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

Bentuk khusus dari ekspresi memungkinkan variabel apa pun sebagai parameter tanpa nama untuk evaluasi ekspresi, bukan hanya "x", "y" dll. Untuk ini, mulai ekspresi dengan "@(variabel) ...".

```
>"@(a,b) a^2+b^2", %(4,5)
```

```
@(a,b) a^2+b^2  
41
```

Ini memungkinkan untuk memanipulasi ekspresi dalam variabel lain untuk fungsi EMT yang membutuhkan ekspresi dalam "x".

Cara paling dasar untuk mendefinisikan fungsi sederhana adalah dengan menyimpan rumusnya dalam ekspresi simbolis atau numerik. Jika variabel utama adalah x, ekspresi dapat dievaluasi seperti fungsi.

Seperti yang Anda lihat dalam contoh berikut, variabel global terlihat selama evaluasi.

```
>fx &= x^3-a*x; ...  
>a=1.2; fx(0.5)
```

-0.475

Semua variabel lain dalam ekspresi dapat ditentukan dalam evaluasi menggunakan parameter yang ditetapkan.

```
>fx(0.5,a=1.1)
```

-0.425

Sebuah ekspresi tidak perlu simbolis. Ini diperlukan, jika ekspresi berisi fungsi, yang hanya diketahui di kernel numerik, bukan di Maxima.

EMT melakukan matematika simbolis dengan bantuan Maxima. Untuk detailnya, mulailah dengan tutorial berikut, atau telusuri referensi untuk Maxima. Para ahli di Maxima harus mencatat bahwa ada perbedaan sintaks antara sintaks asli Maxima dan sintaks default ekspresi simbolik di EMT.

Matematika simbolik terintegrasi dengan mulus ke dalam Euler dengan &. Ekspresi apa pun yang dimulai dengan & adalah ekspresi simbolis. Itu dievaluasi dan dicetak oleh Maxima.

Pertama-tama, Maxima memiliki aritmatika "tak terbatas" yang dapat menangani angka yang sangat besar.

```
> $& 44 !
```

```
2658271574788448768043625811014615890319638528000000000
```

Dengan cara ini, Anda dapat menghitung hasil yang besar dengan tepat. Mari kita hitung laeks: $C(44,10) = \frac{44!}{34! \cdot 10!}$

```
> $& 44! / (34!*10!) // nilai C(44,10)
```

```
2481256778
```

Tentu saja, Maxima memiliki fungsi yang lebih efisien untuk ini (seperti halnya bagian numerik dari EMT).

```
> $binomial(44,10) //menghitung C(44,10) menggunakan fungsi binomial()
```

```
2481256778
```

Untuk mempelajari lebih lanjut tentang fungsi tertentu klik dua kali di atasnya. Misalnya, coba klik dua kali pada "&binomial" di baris perintah sebelumnya. Ini membuka dokumentasi Maxima seperti yang disediakan oleh penulis program itu.

Anda akan belajar bahwa yang berikut ini juga berfungsi.

$$C(x, 3) = \frac{x!}{(x-3)!3!} = \frac{(x-2)(x-1)x}{6}$$

```
> $binomial(x, 3) // C(x, 3)
```

$$\frac{(x-2)(x-1)x}{6}$$

Jika Anda ingin mengganti x dengan nilai tertentu, gunakan "dengan".

```
>${&binomial(x, 3) with x=10 // substitusi x=10 ke C(x, 3)}
```

120

Dengan begitu Anda dapat menggunakan solusi persamaan dalam persamaan lain.

Ekspresi simbolik dicetak oleh Maxima dalam bentuk 2D. Alasan untuk ini adalah bendera simbolis khusus dalam string.

Seperti yang akan Anda lihat pada contoh sebelumnya dan berikut, jika Anda telah menginstal LaTeX, Anda dapat mencetak ekspresi simbolis dengan Lateks. Jika tidak, perintah berikut akan mengeluarkan pesan kesalahan.

Untuk mencetak ekspresi simbolis dengan LaTeX, gunakan \$ di depan & (atau Anda dapat menghilangkan &) sebelum perintah. Jangan menjalankan perintah Maxima dengan \$, jika Anda tidak menginstal LaTeX.

```
>${(3+x) / (x^2+1)}
```

$$\frac{x + 3}{x^2 + 1}$$

Ekspresi simbolik diuraikan oleh Euler. Jika Anda membutuhkan sintaks yang kompleks dalam satu ekspresi, Anda dapat menyertakan ekspresi dalam "...". Untuk menggunakan lebih dari ekspresi sederhana adalah mungkin, tetapi sangat tidak disarankan.

```
>&"v := 5; v^2"
```

25

Untuk kelengkapan, kami menyatakan bahwa ekspresi simbolik dapat digunakan dalam program, tetapi perlu diapit dalam tanda kutip. Selain itu, jauh lebih efektif untuk memanggil Maxima pada waktu kompilasi jika memungkinkan.

```
>${&expand((1+x)^4), ${&factor(diff(% ,x)) // diff: turunan, factor: faktor}}
```

$$4 (x + 1)^3$$

Sekali lagi, % mengacu pada hasil sebelumnya.

Untuk mempermudah, kami menyimpan solusi ke variabel simbolik. Variabel simbolik didefinisikan dengan "&=".

```
>fx &= (x+1) / (x^4+1); $&fx
```

$$\frac{x + 1}{x^4 + 1}$$

Ekspresi simbolik dapat digunakan dalam ekspresi simbolik lainnya.

```
>$factor(diff(fx,x))
```

$$\frac{-3x^4 - 4x^3 + 1}{(x^4 + 1)^2}$$

Masukan langsung dari perintah Maxima juga tersedia. Mulai baris perintah dengan "::". Sintaks Maxima disesuaikan dengan sintaks EMT (disebut "mode kompatibilitas").

```
>&factor(20!)
```

2432902008176640000

```
>::: factor(10!)
```

$$\begin{matrix} 8 & 4 & 2 \\ 2 & 3 & 5 & 7 \end{matrix}$$

```
>::: factor(20!)
```

$$\begin{matrix} 18 & 8 & 4 & 2 \\ 2 & 3 & 5 & 7 & 11 & 13 & 17 & 19 \end{matrix}$$

Jika Anda ahli dalam Maxima, Anda mungkin ingin menggunakan sintaks asli Maxima. Anda dapat melakukannya dengan ":::".

```
>::: av:g$ av^2;
```

$$\begin{matrix} 2 \\ g \end{matrix}$$

```
>fx &= x^3*exp(x), $fx
```

$$\begin{array}{r} 3 \quad x \\ x \quad E \end{array}$$

$$x^3 e^x$$

Variabel tersebut dapat digunakan dalam ekspresi simbolik lainnya. Perhatikan, bahwa dalam perintah berikut sisi kanan &= dievaluasi sebelum penugasan ke Fx.

```
>&(fx with x=5), $%, &float(%)
```

$$\begin{array}{r} 5 \\ 125 \quad E \end{array}$$

$$125 e^5$$

$$18551.64488782208$$

```
>fx(5)
```

$$18551.6448878$$

Untuk evaluasi ekspresi dengan nilai variabel tertentu, Anda dapat menggunakan operator "with".

Baris perintah berikut juga menunjukkan bahwa Maxima dapat mengevaluasi ekspresi secara numerik dengan float().

```
>&(fx with x=10)-(fx with x=5), &float(%)
```

$$\begin{array}{r} 10 \quad 5 \\ 1000 \quad E \quad - \quad 125 \quad E \end{array}$$

$$2.20079141499189e+7$$

```
>$factor(diff(fx,x,2))
```

$$x (x^2 + 6x + 6) e^x$$

Untuk mendapatkan kode Lateks untuk ekspresi, Anda dapat menggunakan perintah `tex`.

```
>tex(fx)
```

$$x^3 \backslash, e^{\{x\}}$$

Ekspresi simbolik dapat dievaluasi seperti ekspresi numerik.

```
>fx(0.5)
```

$$0.206090158838$$

Dalam ekspresi simbolis, ini tidak berfungsi, karena Maxima tidak mendukungnya. Sebagai gantinya, gunakan sintaks "with" (bentuk yang lebih bagus dari perintah `at(...)` dari Maxima).

```
>${&}fx with x=1/2
```

$$\frac{\sqrt{e}}{8}$$

Penugasan juga bisa bersifat simbolis.

```
>${&}fx with x=1+t
```

$$(t + 1)^3 e^{t+1}$$

Perintah `solve` memecahkan ekspresi simbolik untuk variabel di Maxima. Hasilnya adalah vektor solusi.

```
>${&}solve(x^2+x=4, x)
```

$$\left[x = \frac{-\sqrt{17} - 1}{2}, x = \frac{\sqrt{17} - 1}{2} \right]$$

Bandingkan dengan perintah numerik "selesaikan" di Euler, yang membutuhkan nilai awal, dan secara opsional nilai target.

```
>solve("x^2+x", 1, y=4)
```

$$1.56155281281$$

Nilai numerik dari solusi simbolik dapat dihitung dengan evaluasi hasil simbolis. Euler akan membaca tugas `x=` dll. Jika Anda tidak memerlukan hasil numerik untuk perhitungan lebih lanjut, Anda juga dapat memberikan Maxima menemukan nilai numerik.

```
>sol &= solve(x^2+2*x=4,x); $&sol, sol(), $&float(sol)
```

$$\left[x = -\sqrt{5} - 1, x = \sqrt{5} - 1 \right]$$

```
[-3.23607, 1.23607]
```

```
[x = -3.23606797749979, x = 1.23606797749979]
```

Untuk mendapatkan solusi simbolis tertentu, seseorang dapat menggunakan "with" dan index.

```
>$&solve(x^2+x=1,x), x2 &= x with %[2]; $&x2
```

$$\frac{\sqrt{5} - 1}{2}$$

$$\frac{\sqrt{5} - 1}{2}$$

Untuk menyelesaikan sistem persamaan, gunakan vektor persamaan. Hasilnya adalah vektor solusi.

```
>sol &= solve([x+y=3,x^2+y^2=5],[x,y]); $&sol, $&x*y with sol[1]
```

2

Ekspresi simbolis dapat memiliki bendera, yang menunjukkan perlakuan khusus di Maxima. Beberapa flag dapat digunakan sebagai perintah juga, yang lain tidak. Bendera ditambahkan dengan "|" (bentuk yang lebih bagus dari "ev(...,flags)")

```
>$& diff((x^3-1)/(x+1),x) //turunan bentuk pecahan
```

$$\frac{3x^2}{x+1} - \frac{x^3-1}{(x+1)^2}$$

```
>$& diff((x^3-1)/(x+1),x) | ratsimp //menyederhanakan pecahan
```

$$\frac{2x^3 + 3x^2 + 1}{x^2 + 2x + 1}$$

```
>$&factor(%)
```

$$\frac{2x^3 + 3x^2 + 1}{(x+1)^2}$$

Fungsi

Dalam EMT, fungsi adalah program yang didefinisikan dengan perintah "fungsi". Ini bisa berupa fungsi satu baris atau fungsi multibaris.

Fungsi satu baris dapat berupa numerik atau simbolis. Fungsi satu baris numerik didefinisikan oleh ":=".

```
>function f(x) := x*sqrt(x^2+1)
```

Untuk gambaran umum, kami menunjukkan semua kemungkinan definisi untuk fungsi satu baris. Suatu fungsi dapat dievaluasi sama seperti fungsi Euler bawaan lainnya.

```
>f(2)
```

4.472135955

Fungsi ini akan bekerja untuk vektor juga, dengan mematuhi bahasa matriks Euler, karena ekspresi yang digunakan dalam fungsi divektorkan.

```
>f(0:0.1:1)
```

[0, 0.100499, 0.203961, 0.313209, 0.430813, 0.559017, 0.699714, 0.854459, 1.0245]

Fungsi dapat diplot. Alih-alih ekspresi, kita hanya perlu memberikan nama fungsi.

Berbeda dengan ekspresi simbolik atau numerik, nama fungsi harus diberikan dalam string.

```
>solve("f",1,y=1)
```

0.786151377757

Secara default, jika Anda perlu menimpa fungsi bawaan, Anda harus menambahkan kata kunci "menimpa". Menimpa fungsi bawaan berbahaya dan dapat menyebabkan masalah untuk fungsi lain tergantung pada fungsi tersebut.

Anda masih dapat memanggil fungsi bawaan sebagai "...", jika itu adalah fungsi di inti Euler.

```
>function overwrite sin (x) := _sin(x°) // redefine sine in degrees  
>sin(45)
```

0.707106781187

Lebih baik kita menghapus redefinisi dosa ini.

```
>forget sin; sin(pi/4)
```

0.707106781187

Parameter Default

Fungsi numerik dapat memiliki parameter default.

```
>function f(x,a=1) := a*x^2
```

Menghilangkan parameter ini menggunakan nilai default.

```
>f(4)
```

16

Menyetelnya akan menimpa nilai default.

```
>f(4,5)
```

80

Parameter yang ditetapkan menimpanya juga. Ini digunakan oleh banyak fungsi Euler seperti plot2d, plot3d.

```
>f(4,a=1)
```

16

Jika suatu variabel bukan parameter, itu harus global. Fungsi satu baris dapat melihat variabel global.

```
>function f(x) := a*x^2  
>a=6; f(2)
```

24

Tetapi parameter yang ditetapkan menimpa nilai global.

Jika argumen tidak ada dalam daftar parameter yang telah ditentukan sebelumnya, argumen tersebut harus dideklarasikan dengan ":="!

```
>f(2,a:=5)
```

20

Fungsi simbolis didefinisikan dengan "&=". Mereka didefinisikan dalam Euler dan Maxima, dan bekerja di kedua dunia. Ekspresi yang mendefinisikan dijalankan melalui Maxima sebelum definisi.

```
>function g(x) &= x^3-x*exp(-x); $&g(x)
```

$$x^3 - x e^{-x}$$

Fungsi simbolik dapat digunakan dalam ekspresi simbolik.

```
> $&diff(g(x), x), $&% with x=4/3
```

$$\frac{e^{-\frac{4}{3}}}{3} + \frac{16}{3}$$

$$\frac{e^{-\frac{4}{3}}}{3} + \frac{16}{3}$$

Mereka juga dapat digunakan dalam ekspresi numerik. Tentu saja, ini hanya akan berfungsi jika EMT dapat menginterpretasikan semua yang ada di dalam fungsi tersebut.

```
> g(5+g(1))
```

$$178.635099908$$

Mereka dapat digunakan untuk mendefinisikan fungsi atau ekspresi simbolik lainnya.

```
> function G(x) &= factor(integrate(g(x), x)); $&G(c) // integrate: mengintegralkan
```

$$\frac{e^{-c} (c^4 e^c + 4 c + 4)}{4}$$

```
> solve(&g(x), 0.5)
```

$$0.703467422498$$

Berikut ini juga berfungsi, karena Euler menggunakan ekspresi simbolis dalam fungsi g , jika tidak menemukan variabel simbolik g , dan jika ada fungsi simbolis g .

```
> solve(&g, 0.5)
```

$$0.703467422498$$

```
> function P(x, n) &= (2*x-1)^n; $&P(x, n)
```

$$(2x - 1)^n$$

```
> function Q(x, n) &= (x+2)^n; $&Q(x, n)
```

$$(x + 2)^n$$

```
>${&P(x, 4)}, ${&expand(%)}
```

$$16x^4 - 32x^3 + 24x^2 - 8x + 1$$

```
>P(3, 4)
```

625

```
>${&P(x, 4) + Q(x, 3)}, ${&expand(%)}
```

$$16x^4 - 31x^3 + 30x^2 + 4x + 9$$

```
>${&P(x, 4) - Q(x, 3)}, ${&expand(%)}, ${&factor(%)}
```

$$16x^4 - 33x^3 + 18x^2 - 20x - 7$$

```
>${&P(x, 4) * Q(x, 3)}, ${&expand(%)}, ${&factor(%)}
```

$$(x + 2)^3 (2x - 1)^4$$

```
>${&P(x, 4) / Q(x, 1)}, ${&expand(%)}, ${&factor(%)}
```

$$\frac{(2x - 1)^4}{x + 2}$$

$$\frac{16x^4 - 32x^3 + 24x^2 - 8x + 1}{x + 2}$$

$$\frac{(2x - 1)^4}{x + 2}$$

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

$$x^3 - x$$

Dengan `&=` fungsinya simbolis, dan dapat digunakan dalam ekspresi simbolik lainnya.

```
>$&integrate(f(x),x)
```

$$\frac{x^4}{4} - \frac{x^2}{2}$$

Dengan `:=` fungsinya numerik. Contoh yang baik adalah integral tak tentu seperti

$$f(x) = \int_1^x t^t dt,$$

yang tidak dapat dinilai secara simbolis.

Jika kita mendefinisikan kembali fungsi dengan kata kunci "peta" dapat digunakan untuk vektor `x`. Secara internal, fungsi dipanggil untuk semua nilai `x` satu kali, dan hasilnya disimpan dalam vektor.

```
>function map f(x) := integrate("x^x",1,x)
>f(0:0.5:2)
```

```
[-0.783431, -0.410816, 0, 0.676863, 2.05045]
```

Fungsi dapat memiliki nilai default untuk parameter.

```
>function mylog (x,base=10) := ln(x)/ln(base);
```

Sekarang fungsi dapat dipanggil dengan atau tanpa parameter "basis".

```
>mylog(100), mylog(2^6.7,2)
```

```
2
6.7
```

Selain itu, dimungkinkan untuk menggunakan parameter yang ditetapkan.

```
>mylog(E^2,base=E)
```

```
2
```

Seringkali, kita ingin menggunakan fungsi untuk vektor di satu tempat, dan untuk elemen individual di tempat lain. Ini dimungkinkan dengan parameter vektor.

```
>function f([a,b]) &= a^2+b^2-a*b+b; $&f(a,b), $&f(x,y)
```

$$y^2 - x y + y + x^2$$

Fungsi simbolik seperti itu dapat digunakan untuk variabel simbolik.

Tetapi fungsinya juga dapat digunakan untuk vektor numerik.

```
>v=[3,4]; f(v)
```

17

Ada juga fungsi simbolis murni, yang tidak dapat digunakan secara numerik.

```
>function lapl(expr,x,y) &&= diff(expr,x,2)+diff(expr,y,2)//turunan parsial kedua
```

$$\text{diff(expr, y, 2)} + \text{diff(expr, x, 2)}$$

```
>$&realpart((x+I*y)^4), $&lapl(% ,x,y)
```

0

Tetapi tentu saja, mereka dapat digunakan dalam ekspresi simbolik atau dalam definisi fungsi simbolik.

```
>function f(x,y) &= factor(lapl((x+y^2)^5,x,y)); $&f(x,y)
```

$$10 (y^2 + x)^3 (9 y^2 + x + 2)$$

Untuk meringkas

- &= mendefinisikan fungsi simbolis,
- := mendefinisikan fungsi numerik,
- &&= mendefinisikan fungsi simbolis murni.

Memecahkan Ekspresi

Ekspresi dapat diselesaikan secara numerik dan simbolis.

Untuk menyelesaikan ekspresi sederhana dari satu variabel, kita dapat menggunakan fungsi solve(). Perlu nilai awal untuk memulai pencarian. Secara internal, solve() menggunakan metode secant.

```
>solve("x^2-2",1)
```

1.41421356237

Ini juga berfungsi untuk ekspresi simbolis. Ambil fungsi berikut.

```
>$&solve(x^2=2,x)
```

$$[x = -\sqrt{2}, x = \sqrt{2}]$$

```
>$&solve(x^2-2,x)
```

$$[x = -\sqrt{2}, x = \sqrt{2}]$$

```
>$&solve(a*x^2+b*x+c=0,x)
```

$$\left[x = \frac{-\sqrt{b^2 - 4ac} - b}{2a}, x = \frac{\sqrt{b^2 - 4ac} - b}{2a}\right]$$

```
>$&solve([a*x+b*y=c, d*x+e*y=f], [x,y])
```

$$\left[\left[x = -\frac{ce}{b(d-4) - ae}, y = \frac{c(d-4)}{b(d-4) - ae}\right]\right]$$

```
>px &= 4*x^8+x^7-x^4-x; $&px
```

$$4x^8 + x^7 - x^4 - x$$

Sekarang kita mencari titik, di mana polinomialnya adalah 2. Dalam solve(), nilai target default y=0 dapat diubah dengan variabel yang ditetapkan.

Kami menggunakan y=2 dan memeriksa dengan mengevaluasi polinomial pada hasil sebelumnya.

```
>solve(px,1,y=2), px(%)
```

0.966715594851
2

Memecahkan ekspresi simbolis dalam bentuk simbolis mengembalikan daftar solusi. Kami menggunakan pemecah simbolik solve() yang disediakan oleh Maxima.

```
>sol &= solve(x^2-x-1,x); $&sol
```

$$\left[x = \frac{1 - \sqrt{5}}{2}, x = \frac{\sqrt{5} + 1}{2} \right]$$

Cara termudah untuk mendapatkan nilai numerik adalah dengan mengevaluasi solusi secara numerik seperti ekspresi.

```
>longest sol()
```

```
-0.6180339887498949 1.618033988749895
```

Untuk menggunakan solusi secara simbolis dalam ekspresi lain, cara termudah adalah "dengan".

```
>$&x^2 with sol[1], $&expand(x^2-x-1 with sol[2])
```

```
0
```

Memecahkan sistem persamaan secara simbolis dapat dilakukan dengan vektor persamaan dan solver simbolis solve(). Jawabannya adalah daftar daftar persamaan.

```
>$&solve([x+y=2, x^3+2*y+x=4], [x, y])
```

```
[[x = -1, y = 3], [x = 1, y = 1], [x = 0, y = 2]]
```

Fungsi f() dapat melihat variabel global. Namun seringkali kita ingin menggunakan parameter lokal. lateks: $a^x - x^a = 0.1$ dengan $a=3$.

```
>function f(x,a) := x^a-a^x;
```

Salah satu cara untuk meneruskan parameter tambahan ke f() adalah dengan menggunakan daftar dengan nama fungsi dan parameter (sebaliknya adalah parameter titik koma).

```
>solve({{"f", 3}}, 2, y=0.1)
```

```
2.54116291558
```

Ini juga bekerja dengan ekspresi. Tapi kemudian, elemen daftar bernama harus digunakan. (Lebih lanjut tentang daftar di tutorial tentang sintaks EMT).

```
>solve({{"x^a-a^x"},a=3},2,y=0.1)
```

2.54116291558

Menyelesaikan Pertidaksamaan

Untuk menyelesaikan pertidaksamaan, EMT tidak akan dapat melakukannya, melainkan dengan bantuan Maxima, artinya secara eksak (simbolik). Perintah Maxima yang digunakan adalah `fourier_elim()`, yang harus dipanggil dengan perintah "`load(fourier_elim)`" terlebih dahulu.

```
>&load(fourier_elim)
```

C:/Program Files/Euler x64/maxima/share/maxima/5.35.1/share/fourier_elim

```
>$&fourier_elim([x^2 - 1>0],[x]) // x^2-1 > 0
```

$$[1 < x] \vee [x < -1]$$

```
>$&fourier_elim([x^2 - 1<0],[x]) // x^2-1 < 0
```

$$[-1 < x, x < 1]$$

```
>$&fourier_elim([x^2 - 1 # 0],[x]) // x^-1 <> 0
```

$$[-1 < x, x < 1] \vee [1 < x] \vee [x < -1]$$

```
>$&fourier_elim([x # 6],[x])
```

$$[x < 6] \vee [6 < x]$$

```
>$&fourier_elim([x < 1, x > 1],[x]) // tidak memiliki penyelesaian
```

emptyset

```
>$&fourier_elim([minf < x, x < inf],[x]) // solusinya R
```

universalset

```
>$&fourier_elim([x^3 - 1 > 0],[x])
```

$$[1 < x, x^2 + x + 1 > 0] \vee [x < 1, -x^2 - x - 1 > 0]$$

```
>$&fourier_elim([cos(x) < 1/2],[x]) // ??? gagal
```

$$[1 - 2 \cos x > 0]$$

```
>$&fourier_elim([y-x < 5, x - y < 7, 10 < y],[x,y]) // sistem pertidaksamaan
```

$$[y - 5 < x, x < y + 7, 10 < y]$$

```
>$&fourier_elim([y-x < 5, x - y < 7, 10 < y],[y,x])
```

$$[\max(10, x - 7) < y, y < x + 5, 5 < x]$$

```
>$&fourier_elim((x + y < 5) and (x - y > 8), [x,y])
```

$$\left[y + 8 < x, x < 5 - y, y < -\frac{3}{2} \right]$$

```
>$&fourier_elim((x + y < 5) and x < 1) or (x - y > 8), [x,y])
```

$$[y + 8 < x] \vee [x < \min(1, 5 - y)]$$

```
>&fourier_elim([max(x,y) > 6, x # 8, abs(y-1) > 12], [x,y])
```

$$[6 < x, x < 8, y < -11] \text{ or } [8 < x, y < -11] \text{ or } [x < 8, 13 < y] \text{ or } [x = y, 13 <$$

```
>$&fourier_elim([ (x+6) / (x-9) <= 6], [x])
```

$$[x = 12] \vee [12 < x] \vee [x < 9]$$

Bahasa Matriks

Dokumentasi inti EMT berisi diskusi terperinci tentang bahasa matriks Euler.

Vektor dan matriks dimasukkan dengan tanda kurung siku, elemen dipisahkan dengan koma, baris dipisahkan dengan titik koma.

```
>A=[ 1, 2; 3, 4 ]
```

$$\begin{matrix} 1 & 2 \\ 3 & 4 \end{matrix}$$

Produk matriks dilambangkan dengan titik.

```
>b=[ 3; 4 ]
```

$$\begin{matrix} 3 \\ 4 \end{matrix}$$

```
>b' // transpose b
```

$$[3, 4]$$

```
>inv(A) //inverse A
```

$$\begin{matrix} -2 & 1 \\ 1.5 & -0.5 \end{matrix}$$

```
>A.b //perkalian matriks
```

$$\begin{matrix} 11 \\ 25 \end{matrix}$$

```
>A.inv(A)
```

$$\begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}$$

Poin utama dari bahasa matriks adalah bahwa semua fungsi dan operator bekerja elemen untuk elemen.

```
>A.A
```

7	10
15	22

```
>A^2 //perpangkatan elemen2 A
```

1	4
9	16

```
>A.A.A
```

37	54
81	118

```
>power(A, 3) //perpangkatan matriks
```

37	54
81	118

```
>A/A //pembagian elemen-elemen matriks yang seletak
```

1	1
1	1

```
>A\b // pembagian elemen2 A oleh elemen2 b kolom demi kolom (karena b vektor kolom)
```

0.333333	0.666667
0.75	1

```
>A\b // hasil kali invers A dan b, A^(-1)b
```

-2
2.5

```
>inv(A).b
```

-2
2.5

```
>A\A //A^(-1)A
```

$$\begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}$$

```
>inv(A).A
```

$$\begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}$$

```
>A*A //perkalian elemen-elemen matriks seletak
```

$$\begin{matrix} 1 & 4 \\ 9 & 16 \end{matrix}$$

Ini bukan produk matriks, tetapi perkalian elemen demi elemen. Hal yang sama berlaku untuk vektor.

```
>b^2 // perpangkatan elemen-elemen matriks/vektor
```

$$\begin{matrix} 9 \\ 16 \end{matrix}$$

Jika salah satu operan adalah vektor atau skalar, itu diperluas secara alami.

```
>2*A
```

$$\begin{matrix} 2 & 4 \\ 6 & 8 \end{matrix}$$

Misalnya, jika operan adalah vektor kolom, elemennya diterapkan ke semua baris A.

```
>[1,2]*A
```

$$\begin{matrix} 1 & 4 \\ 3 & 8 \end{matrix}$$

Jika itu adalah vektor baris, itu diterapkan ke semua kolom A.

```
>A*[2,3]
```

$$\begin{matrix} 2 & 6 \\ 6 & 12 \end{matrix}$$

Seseorang dapat membayangkan perkalian ini seolah-olah vektor baris v telah digandakan untuk membentuk matriks dengan ukuran yang sama dengan A.

```
>dup([1,2],2) // dup: menduplikasi/menggandakan vektor [1,2] sebanyak 2 kali (baris)
```

```
1          2  
1          2
```

```
>A*dup([1,2],2)
```

```
1          4  
3          8
```

Ini juga berlaku untuk dua vektor di mana satu adalah vektor baris dan yang lainnya adalah vektor kolom. Kami menghitung i^*j untuk i,j dari 1 hingga 5. Caranya adalah dengan mengalikan 1:5 dengan transposnya. Bahasa matriks Euler secara otomatis menghasilkan tabel nilai.

```
>(1:5)*(1:5)' // hasil kali elemen-elemen vektor baris dan vektor kolom
```

1	2	3	4	5
2	4	6	8	10
3	6	9	12	15
4	8	12	16	20
5	10	15	20	25

Sekali lagi, ingat bahwa ini bukan produk matriks!

```
>(1:5).(1:5)' // hasil kali vektor baris dan vektor kolom
```

```
55
```

```
>sum((1:5)*(1:5)) // sama hasilnya
```

```
55
```

Bahkan operator seperti `<` atau `==` bekerja dengan cara yang sama.

```
>(1:10)<6 // menguji elemen-elemen yang kurang dari 6
```

```
[1, 1, 1, 1, 1, 0, 0, 0, 0]
```

Misalnya, kita dapat menghitung jumlah elemen yang memenuhi kondisi tertentu dengan fungsi `sum()`.

```
>sum((1:10)<6) // banyak elemen yang kurang dari 6
```

```
5
```

Euler memiliki operator perbandingan, seperti `"=="`, yang memeriksa kesetaraan.

Kami mendapatkan vektor 0 dan 1, di mana 1 berarti benar.

```
>t=(1:10)^2; t==25 //menguji elemen2 t yang sama dengan 25 (hanya ada 1)
```

```
[0, 0, 0, 0, 1, 0, 0, 0, 0]
```

Dari vektor seperti itu, "bukan nol" memilih elemen bukan nol.

Dalam hal ini, kami mendapatkan indeks semua elemen lebih besar dari 50.

```
>nonzeros(t>50) //indeks elemen2 t yang lebih besar daripada 50
```

```
[8, 9, 10]
```

Tentu saja, kita dapat menggunakan vektor indeks ini untuk mendapatkan nilai yang sesuai dalam t.

```
>t[nonzeros(t>50)] //elemen2 t yang lebih besar daripada 50
```

```
[64, 81, 100]
```

Sebagai contoh, mari kita cari semua kuadrat dari angka 1 hingga 1000, yaitu 5 modulo 11 dan 3 modulo 13.

```
>t=1:1000; nonzeros(mod(t^2,11)==5 && mod(t^2,13)==3)
```

```
[4, 48, 95, 139, 147, 191, 238, 282, 290, 334, 381, 425, 433, 477, 524, 552, 563, 610, 854, 862, 906, 953, 997]
```

EMT tidak sepenuhnya efektif untuk perhitungan bilangan bulat. Ini menggunakan titik mengambang presisi ganda secara internal. Namun, seringkali sangat berguna.

Kita dapat memeriksa keutamaan. Mari kita cari tahu, berapa banyak kuadrat ditambah 1 adalah bilangan prima.

```
>t=1:1000; length(nonzeros(isprime(t^2+1)))
```

```
112
```

Fungsi bukan nol() hanya berfungsi untuk vektor. Untuk matriks, ada mnonzeros().

```
>seed(2); A=random(3,4)
```

```
0.765761      0.401188      0.406347      0.267829  
0.13673       0.390567      0.495975      0.952814  
0.548138      0.006085      0.444255      0.539246
```

Ini mengembalikan indeks elemen, yang bukan nol.

```
>k=mnonzeros(A<0.4) //indeks elemen2 A yang kurang dari 0,4
```

```
1          4  
2          1  
2          2  
3          2
```

Indeks ini dapat digunakan untuk mengatur elemen ke beberapa nilai.

```
>mset(A,k,0) //mengganti elemen2 suatu matriks pada indeks tertentu
```

0.765761	0.401188	0.406347	0
0	0	0.495975	0.952814
0.548138	0	0.444255	0.539246

Fungsi mset() juga dapat mengatur elemen pada indeks ke entri dari beberapa matriks lainnya.

```
>mset(A,k,-random(size(A)) )
```

0.765761	0.401188	0.406347	-0.126917
-0.122404	-0.691673	0.495975	0.952814
0.548138	-0.483902	0.444255	0.539246

Dan dimungkinkan untuk mendapatkan elemen dalam vektor.

```
>mget(A,k)
```

```
[0.267829, 0.13673, 0.390567, 0.006085]
```

Fungsi lain yang berguna adalah ekstrem, yang mengembalikan nilai minimal dan maksimal di setiap baris matriks dan posisinya.

```
>ex=extrema(A)
```

0.267829	4	0.765761	1
0.13673	1	0.952814	4
0.006085	2	0.548138	1

Kita dapat menggunakan ini untuk mengekstrak nilai maksimal di setiap baris.

```
>ex[,3]'
```

```
[0.765761, 0.952814, 0.548138]
```

Ini, tentu saja, sama dengan fungsi max().

```
>max(A)'
```

```
[0.765761, 0.952814, 0.548138]
```

Tetapi dengan mget(), kita dapat mengekstrak indeks dan menggunakan informasi ini untuk mengekstrak elemen pada posisi yang sama dari matriks lain.

```
>j=(1:rows(A))' | ex[,4], mget(-A,j)
```

1	1
2	4
3	1
[-0.765761, -0.952814, -0.548138]	

Fungsi Matriks Lainnya (Membangun Matriks)

Untuk membangun matriks, kita dapat menumpuk satu matriks di atas yang lain. Jika keduanya tidak memiliki jumlah kolom yang sama, kolom yang lebih pendek akan diisi dengan 0.

```
>v=1:3; v_v
```

1	2	3
1	2	3

Demikian juga, kita dapat melampirkan matriks ke yang lain secara berdampingan, jika keduanya memiliki jumlah baris yang sama.

```
>A=random(3,4); A|v'
```

0.032444	0.0534171	0.595713	0.564454	1
0.83916	0.175552	0.396988	0.83514	2
0.0257573	0.658585	0.629832	0.770895	3

Jika mereka tidak memiliki jumlah baris yang sama, matriks yang lebih pendek diisi dengan 0.

Ada pengecualian untuk aturan ini. Bilangan real yang dilampirkan pada matriks akan digunakan sebagai kolom yang diisi dengan bilangan real tersebut.

```
>A|1
```

0.032444	0.0534171	0.595713	0.564454	1
0.83916	0.175552	0.396988	0.83514	1
0.0257573	0.658585	0.629832	0.770895	1

Diumungkinkan untuk membuat matriks vektor baris dan kolom.

```
>[v;v]
```

1	2	3
1	2	3

```
>[v',v']
```

1	1
2	2
3	3

Tujuan utama dari ini adalah untuk menafsirkan vektor ekspresi untuk vektor kolom.

```
>" [x, x^2] " (v' )
```

1	1
2	4
3	9

Untuk mendapatkan ukuran A, kita dapat menggunakan fungsi berikut.

```
>C=zeros(2,4); rows(C), cols(C), size(C), length(C)
```

2	
4	
[2,	4]
4	

Untuk vektor, ada panjang().

```
>length(2:10)
```

9

Ada banyak fungsi lain, yang menghasilkan matriks.

```
>ones(2,2)
```

1	1
1	1

Ini juga dapat digunakan dengan satu parameter. Untuk mendapatkan vektor dengan angka selain 1, gunakan yang berikut ini.

```
>ones(5)*6
```

[6, 6, 6, 6, 6]

Juga matriks bilangan acak dapat dihasilkan dengan acak (distribusi seragam) atau normal (distribusi Gau).

```
>random(2,2)
```

0.66566	0.831835
0.977	0.544258

Berikut adalah fungsi lain yang berguna, yang merestrukturisasi elemen matriks menjadi matriks lain.

```
>redim(1:9,3,3) // menyusun elemen 1, 2, 3, ..., 9 ke bentuk matriks 3x3
```

1	2	3
4	5	6
7	8	9

Dengan fungsi berikut, kita dapat menggunakan ini dan fungsi dup untuk menulis fungsi rep(), yang mengulang vektor n kali.

```
>function rep(v,n) := redim(dup(v,n),1,n*cols(v))
```

Mari kita uji.

```
>rep(1:3,5)
```

```
[1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3]
```

Fungsi multdup() menduplikasi elemen vektor.

```
>multdup(1:3,5), multdup(1:3,[2,3,2])
```

```
[1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3]  
[1, 1, 2, 2, 2, 3, 3]
```

Fungsi flipx() dan flipy() mengembalikan urutan baris atau kolom matriks. Yaitu, fungsi flipx() membalik secara horizontal.

```
>flipx(1:5) //membalik elemen2 vektor baris
```

```
[5, 4, 3, 2, 1]
```

Untuk rotasi, Euler memiliki rotleft() dan rotright().

```
>rotleft(1:5) // memutar elemen2 vektor baris
```

```
[2, 3, 4, 5, 1]
```

Sebuah fungsi khusus adalah drop(v,i), yang menghilangkan elemen dengan indeks di i dari vektor v.

```
>drop(10:20,3)
```

```
[10, 11, 13, 14, 15, 16, 17, 18, 19, 20]
```

Perhatikan bahwa vektor i di drop(v,i) mengacu pada indeks elemen di v, bukan nilai elemen. Jika Anda ingin menghapus elemen, Anda harus menemukan elemennya terlebih dahulu. Fungsi indexof(v,x) dapat digunakan untuk mencari elemen x dalam vektor terurut v.

```
>v=primes(50), i=indexof(v,10:20), drop(v,i)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]  
[0, 5, 0, 6, 0, 0, 0, 7, 0, 8, 0]  
[2, 3, 5, 7, 23, 29, 31, 37, 41, 43, 47]
```

Seperti yang Anda lihat, tidak ada salahnya untuk memasukkan indeks di luar rentang (seperti 0), indeks ganda, atau indeks yang tidak diurutkan.

```
>drop(1:10,shuffle([0,0,5,5,7,12,12]))
```

```
[1, 2, 3, 4, 6, 8, 9, 10]
```

Ada beberapa fungsi khusus untuk mengatur diagonal atau untuk menghasilkan matriks diagonal. Kita mulai dengan matriks identitas.

```
>A=id(5) // matriks identitas 5x5
```

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

Kemudian kita atur diagonal bawah (-1) menjadi 1:4.

```
>setdiag(A,-1,1:4) //mengganti diagonal di bawah diagonal utama
```

1	0	0	0	0
1	1	0	0	0
0	2	1	0	0
0	0	3	1	0
0	0	0	4	1

Perhatikan bahwa kami tidak mengubah matriks A. Kami mendapatkan matriks baru sebagai hasil dari setdiag().

Berikut adalah fungsi, yang mengembalikan matriks tri-diagonal.

```
>function tridiag (n,a,b,c) := setdiag(setdiag(b*id(n),1,c),-1,a); ...
>tridiag(5,1,2,3)
```

2	3	0	0	0
1	2	3	0	0
0	1	2	3	0
0	0	1	2	3
0	0	0	1	2

Diagonal suatu matriks juga dapat diekstraksi dari matriks tersebut. Untuk mendemonstrasikan ini, kami merestrukturisasi vektor 1:9 menjadi matriks 3x3.

```
>A=redim(1:9,3,3)
```

1	2	3
4	5	6
7	8	9

Sekarang kita dapat mengekstrak diagonal.

```
>d=getdiag(A, 0)
```

```
[1, 5, 9]
```

Misalnya. Kita dapat membagi matriks dengan diagonalnya. Bahasa matriks memperhatikan bahwa vektor kolom d diterapkan ke matriks baris demi baris.

```
>fraction A/d'
```

1	2	3
4/5	1	6/5
7/9	8/9	1

Vektorisasi

Hampir semua fungsi di Euler juga berfungsi untuk input matriks dan vektor, kapan pun ini masuk akal. Misalnya, fungsi sqrt() menghitung akar kuadrat dari semua elemen vektor atau matriks.

```
>sqrt(1:3)
```

```
[1, 1.41421, 1.73205]
```

Jadi Anda dapat dengan mudah membuat tabel nilai. Ini adalah salah satu cara untuk memplot suatu fungsi (alternatifnya menggunakan ekspresi).

```
>x=1:0.01:5; y=log(x)/x^2; // terlalu panjang untuk ditampilkan
```

Dengan ini dan operator titik dua a:delta:b, vektor nilai fungsi dapat dihasilkan dengan mudah.

Pada contoh berikut, kita membangkitkan vektor nilai t[i] dengan spasi 0,1 dari -1 hingga 1. Kemudian kita membangkitkan vektor nilai fungsi

lateks: $s = t^3 - t$

```
>t=-1:0.1:1; s=t^3-t
```

```
[0, 0.171, 0.288, 0.357, 0.384, 0.375, 0.336, 0.273, 0.192, 0.099, 0, -0.099,
-0.384, -0.357, -0.288, -0.171, 0]
```

EMT memperluas operator untuk skalar, vektor, dan matriks dengan cara yang jelas.

Misalnya, vektor kolom dikalikan vektor baris menjadi matriks, jika operator diterapkan. Berikut ini, v' adalah vektor yang ditransposisikan (vektor kolom).

```
>shortest (1:5)*(1:5)'
```

1	2	3	4	5
2	4	6	8	10
3	6	9	12	15
4	8	12	16	20
5	10	15	20	25

Perhatikan, bahwa ini sangat berbeda dari produk matriks. Produk matriks dilambangkan dengan titik "." di EMT.

```
>(1:5) . (1:5) '
```

55

Secara default, vektor baris dicetak dalam format yang ringkas.

```
>[1, 2, 3, 4]
```

[1, 2, 3, 4]

Untuk matriks operator khusus . menunjukkan perkalian matriks, dan A' menunjukkan transpos. Matriks 1x1 dapat digunakan seperti bilangan real.

```
>v:=[1,2]; v.v', %^2
```

5
25

Untuk mentranspos matriks kita menggunakan apostrof.

```
>v=1:4; v'
```

1
2
3
4

Jadi kita dapat menghitung matriks A kali vektor b.

```
>A=[1,2,3,4;5,6,7,8]; A.v'
```

30
70

Perhatikan bahwa v masih merupakan vektor baris. Jadi v'.v berbeda dari v.v'.

```
>v' . v
```

1	2	3	4
2	4	6	8
3	6	9	12
4	8	12	16

v.v' menghitung norma v kuadrat untuk vektor baris v. Hasilnya adalah vektor 1x1, yang bekerja seperti bilangan real.

```
>v.v'
```

30

Ada juga fungsi norm (bersama dengan banyak fungsi lain dari Aljabar Linier).

```
>norm(v)^2
```

30

Operator dan fungsi mematuhi bahasa matriks Euler.

Berikut ringkasan aturannya.

- Fungsi yang diterapkan ke vektor atau matriks diterapkan ke setiap elemen.
- Operator yang beroperasi pada dua matriks dengan ukuran yang sama diterapkan berpasangan ke elemen matriks.
- Jika kedua matriks memiliki dimensi yang berbeda, keduanya diperluas dengan cara yang masuk akal, sehingga memiliki ukuran yang sama.

Misalnya, nilai skalar kali vektor mengalikan nilai dengan setiap elemen vektor. Atau matriks kali vektor (dengan *, bukan .) memperluas vektor ke ukuran matriks dengan menduplikasinya.

Berikut ini adalah kasus sederhana dengan operator ^.

```
>[1,2,3]^2
```

[1, 4, 9]

Berikut adalah kasus yang lebih rumit. Vektor baris dikalikan dengan vektor kolom mengembang keduanya dengan menduplikasi.

```
>v:=[1,2,3]; v*v'
```

1	2	3
2	4	6
3	6	9

Perhatikan bahwa produk skalar menggunakan produk matriks, bukan *!

```
>v.v'
```

14

Ada banyak fungsi matriks. Kami memberikan daftar singkat. Anda harus berkonsultasi dengan dokumentasi untuk informasi lebih lanjut tentang perintah ini.

```
sum,prod menghitung jumlah dan produk dari baris  
cumsum,cumprod melakukan hal yang sama secara kumulatif  
menghitung nilai ekstrem dari setiap baris  
extrema mengembalikan vektor dengan informasi ekstrim  
diag(A,i) mengembalikan diagonal ke-i  
setdiag(A,i,v) mengatur diagonal ke-i  
id(n) matriks identitas  
det(A) penentu  
charpoly(A) polinomial karakteristik  
nilai eigen(A) nilai eigen
```

```
>v*v, sum(v*v), cumsum(v*v)
```

```
[1, 4, 9]  
14  
[1, 5, 14]
```

Operator : menghasilkan vektor baris spasi yang sama, opsional dengan ukuran langkah.

```
>1:4, 1:2:10
```

```
[1, 2, 3, 4]  
[1, 3, 5, 7, 9]
```

Untuk menggabungkan matriks dan vektor ada operator "|" dan "_" .

```
>[1,2,3] | [4,5], [1,2,3]_1
```

```
[1, 2, 3, 4, 5]  
1 2 3  
1 1 1
```

Unsur-unsur matriks disebut dengan "A[i,j]" .

```
>A:=[1,2,3;4,5,6;7,8,9]; A[2,3]
```

6

Untuk vektor baris atau kolom, v[i] adalah elemen ke-i dari vektor. Untuk matriks, ini mengembalikan baris ke-i lengkap dari matriks.

```
>v:=[2,4,6,8]; v[3], A[3]
```

```
6  
[7, 8, 9]
```

Indeks juga bisa menjadi vektor baris dari indeks. : menunjukkan semua indeks.

```
>v[1:2], A[:,2]
```

```
[2, 4]  
2  
5  
8
```

Bentuk singkat untuk : adalah menghilangkan indeks sepenuhnya.

```
>A[,2:3]
```

```
2      3  
5      6  
8      9
```

Untuk tujuan vektorisasi, elemen matriks dapat diakses seolah-olah mereka adalah vektor.

```
>A{4}
```

```
4
```

Matriks juga dapat diratakan, menggunakan fungsi redim(). Ini diimplementasikan dalam fungsi flatten().

```
>redim(A,1,prod(size(A))), flatten(A)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9]  
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Untuk menggunakan matriks untuk tabel, mari kita reset ke format default, dan menghitung tabel nilai sinus dan kosinus. Perhatikan bahwa sudut dalam radian secara default.

```
>defformat; w=0°:45°:360°; w=w'; deg(w)
```

```
0  
45  
90  
135  
180  
225  
270  
315  
360
```

Sekarang kita menambahkan kolom ke matriks.

```
>M = deg(w)|w|cos(w)|sin(w)
```

0	0	1	0
45	0.785398	0.707107	0.707107
90	1.5708	0	1
135	2.35619	-0.707107	0.707107
180	3.14159	-1	0
225	3.92699	-0.707107	-0.707107
270	4.71239	0	-1
315	5.49779	0.707107	-0.707107
360	6.28319	1	0

Dengan menggunakan bahasa matriks, kita dapat menghasilkan beberapa tabel dari beberapa fungsi sekaligus.

Dalam contoh berikut, kita menghitung $t[j]^i$ untuk i dari 1 hingga n . Kami mendapatkan matriks, di mana setiap baris adalah tabel t^i untuk satu i . Yaitu, matriks memiliki elemen lateks: $a_{i,j} = t_j^i$, \quad 1 \leq j \leq 101, \quad 1 \leq i \leq n

Fungsi yang tidak berfungsi untuk input vektor harus "divektorkan". Ini dapat dicapai dengan kata kunci "peta" dalam definisi fungsi. Kemudian fungsi tersebut akan dievaluasi untuk setiap elemen dari parameter vektor.

Integrasi numerik terintegrasi() hanya berfungsi untuk batas interval skalar. Jadi kita perlu membuat vektor.

```
>function map f(x) := integrate("x^x",1,x)
```

Kata kunci "peta" membuat vektor fungsi. Fungsinya sekarang akan bekerja untuk vektor bilangan.

```
>f([1:5])
```

```
[0, 2.05045, 13.7251, 113.336, 1241.03]
```

Sub-Matriks dan Matriks-Elemen

Untuk mengakses elemen matriks, gunakan notasi braket.

```
>A=[1,2,3;4,5,6;7,8,9], A[2,2]
```

1	2	3
4	5	6
7	8	9
5		

Kita dapat mengakses satu baris matriks yang lengkap.

```
>A[2]
```

```
[4, 5, 6]
```

Dalam kasus vektor baris atau kolom, ini mengembalikan elemen vektor.

```
>v=1:3; v[2]
```

2

Untuk memastikan, Anda mendapatkan baris pertama untuk matriks $1 \times n$ dan $m \times n$, tentukan semua kolom menggunakan indeks kedua kosong.

```
>A[2, ]
```

[4, 5, 6]

Jika indeks adalah vektor indeks, Euler akan mengembalikan baris matriks yang sesuai.

Di sini kita ingin baris pertama dan kedua dari A.

```
>A[[1,2]]
```

1	2	3
4	5	6

Kita bahkan dapat menyusun ulang A menggunakan vektor indeks. Tepatnya, kami tidak mengubah A di sini, tetapi menghitung versi A yang disusun ulang.

```
>A[[3,2,1]]
```

7	8	9
4	5	6
1	2	3

Trik indeks bekerja dengan kolom juga.

Contoh ini memilih semua baris A dan kolom kedua dan ketiga.

```
>A[1:3,2:3]
```

2	3
5	6
8	9

Untuk singkatan ":" menunjukkan semua indeks baris atau kolom.

```
>A[:,3]
```

3
6
9

Atau, biarkan indeks pertama kosong.

```
>A[, 2:3]
```

2	3
5	6
8	9

Kita juga bisa mendapatkan baris terakhir dari A.

```
>A[-1]
```

[7, 8, 9]

Sekarang mari kita ubah elemen A dengan menetapkan submatriks A ke beberapa nilai. Ini sebenarnya mengubah matriks A yang disimpan.

```
>A[1, 1]=4
```

4	2	3
4	5	6
7	8	9

Kami juga dapat menetapkan nilai ke baris A.

```
>A[1]=[-1, -1, -1]
```

-1	-1	-1
4	5	6
7	8	9

Kami bahkan dapat menetapkan sub-matriks jika memiliki ukuran yang tepat.

```
>A[1:2, 1:2]=[5, 6; 7, 8]
```

5	6	-1
7	8	6
7	8	9

Selain itu, beberapa jalan pintas diperbolehkan.

```
>A[1:2, 1:2]=0
```

0	0	-1
0	0	6
7	8	9

Peringatan: Indeks di luar batas mengembalikan matriks kosong, atau pesan kesalahan, tergantung pada pengaturan sistem. Standarnya adalah pesan kesalahan. Ingat, bagaimanapun, bahwa indeks negatif dapat digunakan untuk mengakses elemen matriks yang dihitung dari akhir.

```
>A[4]
```

```
Row index 4 out of bounds!
Error in:
A[4] ...
^
```

Menyortir dan Mengacak

Fungsi sort() mengurutkan vektor baris.

```
>sort([5, 6, 4, 8, 1, 9])
```

```
[1, 4, 5, 6, 8, 9]
```

Seringkali perlu untuk mengetahui indeks dari vektor yang diurutkan dalam vektor aslinya. Ini dapat digunakan untuk menyusun ulang vektor lain dengan cara yang sama.

Mari kita mengocok vektor.

```
>v=shuffle(1:10)
```

```
[4, 5, 10, 6, 8, 9, 1, 7, 2, 3]
```

Indeks berisi urutan yang tepat dari v.

```
>{vs,ind}=sort(v); v[ind]
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

Ini bekerja untuk vektor string juga.

```
>s=["a","d","e","a","aa","e"]
```

```
a  
d  
e  
a  
aa  
e
```

```
>{ss,ind}=sort(s); ss
```

```
a  
a  
aa  
d  
e  
e
```

Seperti yang Anda lihat, posisi entri ganda agak acak.

```
>ind
```

```
[4, 1, 5, 2, 6, 3]
```

Fungsi unik mengembalikan daftar elemen unik vektor yang diurutkan.

```
>intrandom(1,10,10), unique(%)
```

```
[4, 4, 9, 2, 6, 5, 10, 6, 5, 1]
[1, 2, 4, 5, 6, 9, 10]
```

Ini bekerja untuk vektor string juga.

```
>unique(s)
```

```
a
aa
d
e
```

Aljabar linier

EMT memiliki banyak fungsi untuk menyelesaikan sistem linier, sistem sparse, atau masalah regresi.

Untuk sistem linier $Ax=b$, Anda dapat menggunakan algoritma Gauss, matriks invers atau kecocokan linier. Operator $A\b$ menggunakan versi algoritma Gauss.

```
>A=[1,2;3,4]; b=[5;6]; A\b
```

```
-4
4.5
```

Untuk contoh lain, kami membuat matriks 200x200 dan jumlah barisnya. Kemudian kita selesaikan $Ax=b$ menggunakan matriks invers. Kami mengukur kesalahan sebagai deviasi maksimal semua elemen dari 1, yang tentu saja merupakan solusi yang benar.

```
>A=normal(200,200); b=sum(A); longest totalmax(abs(inv(A).b-1))
```

```
8.790745908981989e-13
```

Jika sistem tidak memiliki solusi, kecocokan linier meminimalkan norma kesalahan $Ax-b$.

```
>A=[1,2,3;4,5,6;7,8,9]
```

1	2	3
4	5	6
7	8	9

Determinan matriks ini adalah 0.

```
>det (A)
```

0

Matriks Simbolik

Maxima memiliki matriks simbolis. Tentu saja, Maxima dapat digunakan untuk masalah aljabar linier sederhana seperti itu. Kita dapat mendefinisikan matriks untuk Euler dan Maxima dengan &:=, dan kemudian menggunakan dalam ekspresi simbolis. Bentuk [...] biasa untuk mendefinisikan matriks dapat digunakan di Euler untuk mendefinisikan matriks simbolik.

```
>A &= [a,1,1;1,a,1;1,1,a]; $A
```

$$\begin{pmatrix} a & 1 & 1 \\ 1 & a & 1 \\ 1 & 1 & a \end{pmatrix}$$

```
>$&det (A), $&factor (%)
```

$$(a - 1)^2 (a + 2)$$

```
>$&invert (A) with a=0
```

$$\begin{pmatrix} -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \end{pmatrix}$$

```
>A &= [1,a;b,2]; $A
```

$$\begin{pmatrix} 1 & a \\ b & 2 \end{pmatrix}$$

Seperti semua variabel simbolik, matriks ini dapat digunakan dalam ekspresi simbolik lainnya.

```
>$&det (A-x*ident (2)), $&solve (% ,x)
```

$$\left[x = \frac{3 - \sqrt{4ab + 1}}{2}, x = \frac{\sqrt{4ab + 1} + 3}{2} \right]$$

$$\left[x = \frac{3 - \sqrt{4ab + 1}}{2}, x = \frac{\sqrt{4ab + 1} + 3}{2} \right]$$

Nilai eigen juga dapat dihitung secara otomatis. Hasilnya adalah vektor dengan dua vektor nilai eigen dan multiplisitas.

```
> $&eigenvalues([a, 1; 1, a])
```

$$[[a - 1, a + 1], [1, 1]]$$

Untuk mengekstrak vektor eigen tertentu perlu pengindeksan yang cermat.

```
> $&eigenvectors([a, 1; 1, a]), &%[2] [1] [1]
```

$$[[[a - 1, a + 1], [1, 1]], [[[1, -1]], [[1, 1]]]]$$

$$[1, \quad - 1]$$

Matriks simbolik dapat dievaluasi dalam Euler secara numerik seperti ekspresi simbolik lainnya.

```
> A(a=4, b=5)
```

$$\begin{matrix} 1 & 4 \\ 5 & 2 \end{matrix}$$

Dalam ekspresi simbolik, gunakan dengan.

```
> $&A with [a=4, b=5]
```

$$\begin{pmatrix} 1 & 4 \\ 5 & 2 \end{pmatrix}$$

Akses ke baris matriks simbolik bekerja seperti halnya dengan matriks numerik.

```
> $&A[1]
```

$$[1, a]$$

Ekspresi simbolis dapat berisi tugas. Dan itu mengubah matriks A.

```
> &A[1, 1]:=t+1; $&A
```

$$\begin{pmatrix} t + 1 & a \\ b & 2 \end{pmatrix}$$

Ada fungsi simbolik di Maxima untuk membuat vektor dan matriks. Untuk ini, lihat dokumentasi Maxima atau tutorial tentang Maxima di EMT.

```
>v &= makelist(1/(i+j), i, 1, 3); $v
```

$$\left[\frac{1}{j+1}, \frac{1}{j+2}, \frac{1}{j+3} \right]$$

```
>B &:= [1, 2; 3, 4]; $B, $&invert(B)
```

$$\begin{pmatrix} -2 & 1 \\ \frac{3}{2} & -\frac{1}{2} \end{pmatrix}$$

$$\begin{pmatrix} -2 & 1 \\ \frac{3}{2} & -\frac{1}{2} \end{pmatrix}$$

Hasilnya dapat dievaluasi secara numerik dalam Euler. Untuk informasi lebih lanjut tentang Maxima, lihat pengantar Maxima.

```
>$&invert(B)()
```

$$\begin{array}{cc} -2 & 1 \\ 1.5 & -0.5 \end{array}$$

Euler juga memiliki fungsi xinv() yang kuat, yang membuat upaya lebih besar dan mendapatkan hasil yang lebih tepat.

Perhatikan, bahwa dengan &:= matriks B telah didefinisikan sebagai simbolik dalam ekspresi simbolik dan sebagai numerik dalam ekspresi numerik. Jadi kita bisa menggunakanya di sini.

```
>longest B.xinv(B)
```

$$\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}$$

Misalnya. nilai eigen dari A dapat dihitung secara numerik.

```
>A=[1, 2, 3; 4, 5, 6; 7, 8, 9]; real(eigenvalues(A))
```

$$[16.1168, -1.11684, 0]$$

Atau secara simbolis. Lihat tutorial tentang Maxima untuk detailnya.

```
>$&eigenvalues(@A)
```

$$\left[\left[\frac{15 - 3\sqrt{33}}{2}, \frac{3\sqrt{33} + 15}{2}, 0 \right], [1, 1, 1] \right]$$

Nilai Numerik dalam Ekspresi simbolis

Ekspresi simbolis hanyalah string yang berisi ekspresi. Jika kita ingin mendefinisikan nilai baik untuk ekspresi simbolik maupun ekspresi numerik, kita harus menggunakan "&:=".

```
>A &:= [1,pi;4,5]
```

```
1      3.14159  
4          5
```

Masih ada perbedaan antara bentuk numerik dan simbolik. Saat mentransfer matriks ke bentuk simbolis, pendekatan fraksional untuk real akan digunakan.

```
>$&A
```

$$\begin{pmatrix} 1 & \frac{1146408}{364913} \\ 4 & 5 \end{pmatrix}$$

Untuk menghindarinya, ada fungsi "mxmset(variable)".

```
>m xmset (A); $&A
```

$$\begin{pmatrix} 1 & 3.141592653589793 \\ 4 & 5 \end{pmatrix}$$

Maxima juga dapat menghitung dengan angka floating point, dan bahkan dengan angka floating besar dengan 32 digit. Namun, evaluasinya jauh lebih lambat.

```
>$&bfloat(sqrt(2)), $&float(sqrt(2))
```

```
1.414213562373095
```

Ketepatan angka floating point besar dapat diubah.

```
>&fpprec:=100; &bfloat(pi)
```

```
3.141592653589793238462643383279502884197169399375105820974944592307816406286
```

Variabel numerik dapat digunakan dalam ekspresi simbolis apa pun menggunakan "@var".

Perhatikan bahwa ini hanya diperlukan, jika variabel telah didefinisikan dengan ":=" atau "=" sebagai variabel numerik.

```
>B:=[1,pi;3,4]; $&det (@B)
```

-5.424777960769379

Demo - Suku Bunga

Di bawah ini, kami menggunakan Euler Math Toolbox (EMT) untuk perhitungan suku bunga. Kami melakukannya secara numerik dan simbolis untuk menunjukkan kepada Anda bagaimana Euler dapat digunakan untuk memecahkan masalah kehidupan nyata.

Asumsikan Anda memiliki modal awal 5000 (katakanlah dalam dolar).

```
>K=5000
```

5000

Sekarang kita asumsikan tingkat bunga 3% per tahun. Mari kita tambahkan satu tarif sederhana dan hitung hasilnya.

```
>K*1.03
```

5150

Euler akan memahami sintaks berikut juga.

```
>K+K*3%
```

5150

Tetapi lebih mudah menggunakan faktornya

```
>q=1+3%, K*q
```

1.03
5150

Selama 10 tahun, kita cukup mengalikan faktornya dan mendapatkan nilai akhir dengan suku bunga majemuk.

```
>K*q^10
```

6719.58189672

Untuk tujuan kita, kita dapat mengatur format menjadi 2 digit setelah titik desimal.

```
>format(12,2); K*q^10
```

6719.58

Mari kita cetak yang dibulatkan menjadi 2 digit dalam kalimat lengkap.

```
>"Starting from " + K + "$ you get " + round(K*q^10,2) + "$."
```

Starting from 5000\$ you get 6719.58\$.

Bagaimana jika kita ingin mengetahui hasil antara dari tahun 1 sampai tahun 9? Untuk ini, bahasa matriks Euler sangat membantu. Anda tidak harus menulis loop, tetapi cukup masukkan

```
>K*q^(0:10)
```

Real 1 x 11 matrix

5000.00 5150.00 5304.50 5463.64 5627.54 5796.37 5970.26

Bagaimana keajaiban ini bekerja? Pertama ekspresi 0:10 mengembalikan vektor bilangan bulat.

```
>short 0:10
```

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

Kemudian semua operator dan fungsi dalam Euler dapat diterapkan pada elemen vektor untuk elemen. Jadi

```
>short q^(0:10)
```

[1, 1.03, 1.0609, 1.0927, 1.1255, 1.1593, 1.1941, 1.2299, 1.2668, 1.3048, 1.34

adalah vektor faktor q^0 sampai q^{10} . Ini dikalikan dengan K , dan kami mendapatkan vektor nilai.

```
>VK=K*q^(0:10);
```

Tentu saja, cara realistik untuk menghitung suku bunga ini adalah dengan membulatkan ke sen terdekat setelah setiap tahun. Mari kita tambahkan fungsi untuk ini.

```
>function oneyear (K) := round(K*q,2)
```

Mari kita bandingkan dua hasil, dengan dan tanpa pembulatan.

```
>longest oneyear(1234.57), longest 1234.57*q
```

1271.61
1271.6071

Sekarang tidak ada rumus sederhana untuk tahun ke- n , dan kita harus mengulang selama bertahun-tahun. Euler memberikan banyak solusi untuk ini.

Cara termudah adalah iterasi fungsi, yang mengulangi fungsi tertentu beberapa kali.

```
>VKr=iterate("oneyear",5000,10)
```

Real 1 x 11 matrix

5000.00	5150.00	5304.50	5463.64	5627.55	5796.38	5970.27
---------	---------	---------	---------	---------	---------	---------

Kami dapat mencetaknya dengan cara yang ramah, menggunakan format kami dengan tempat desimal tetap.

```
>VKr'
```

5000.00
5150.00
5304.50
5463.64
5627.55
5796.38
5970.27
6149.38
6333.86
6523.88
6719.60

Untuk mendapatkan elemen tertentu dari vektor, kami menggunakan indeks dalam tanda kurung siku.

```
>VKr[2], VKr[1:3]
```

5150.00		
5000.00	5150.00	5304.50

Anehnya, kita juga bisa menggunakan vektor indeks. Ingat bahwa 1:3 menghasilkan vektor [1,2,3].

Mari kita bandingkan elemen terakhir dari nilai yang dibulatkan dengan nilai penuh.

```
>VKr[-1], VK[-1]
```

6719.60
6719.58

Perbedaannya sangat kecil.

Memecahkan Persamaan

Sekarang kita mengambil fungsi yang lebih maju, yang menambahkan tingkat uang tertentu setiap tahun.

```
>function onepay (K) := K*q+R
```

Kita tidak perlu menentukan q atau R untuk definisi fungsi. Hanya jika kita menjalankan perintah, kita harus mendefinisikan nilai-nilai ini. Kami memilih R=200.

```
>R=200; iterate("onepay",5000,10)
```

```
Real 1 x 11 matrix  
5000.00 5350.00 5710.50 6081.82 6464.27 6858.20 7263.94
```

Bagaimana jika kita menghapus jumlah yang sama setiap tahun?

```
>R=-200; iterate("onepay",5000,10)
```

```
Real 1 x 11 matrix  
5000.00 4950.00 4898.50 4845.45 4790.82 4734.54 4676.58
```

Kami melihat bahwa uang berkurang. Jelas, jika kita hanya mendapatkan 150 bunga di tahun pertama, tetapi menghapus 200, kita kehilangan uang setiap tahun.

Bagaimana kita bisa menentukan berapa tahun uang itu akan bertahan? Kita harus menulis loop untuk ini. Cara termudah adalah dengan iterasi cukup lama.

```
>VKR=iterate("onepay",5000,50)
```

```
Real 1 x 51 matrix  
5000.00 4950.00 4898.50 4845.45 4790.82 4734.54 4676.58
```

Dengan menggunakan bahasa matriks, kita dapat menentukan nilai negatif pertama dengan cara berikut.

```
>min(nonzeros(VKR<0))
```

48.00

Alasan untuk ini adalah bahwa bukan nol(VKR<0) mengembalikan vektor indeks i, di mana VKR[i]<0, dan min menghitung indeks minimal.

Karena vektor selalu dimulai dengan indeks 1, jawabannya adalah 47 tahun.

Fungsi iterate() memiliki satu trik lagi. Itu bisa mengambil kondisi akhir sebagai argumen. Kemudian akan mengembalikan nilai dan jumlah iterasi.

```
>{x,n}=iterate("onepay",5000,till="x<0"); x, n,
```

-19.83
47.00

Mari kita coba menjawab pertanyaan yang lebih ambigu. Asumsikan kita tahu bahwa nilainya adalah 0 setelah 50 tahun. Apa yang akan menjadi tingkat bunga?

Ini adalah pertanyaan yang hanya bisa dijawab dengan angka. Di bawah ini, kita akan mendapatkan formula yang diperlukan. Kemudian Anda akan melihat bahwa tidak ada formula yang mudah untuk tingkat bunga. Tapi untuk saat ini, kami bertujuan untuk solusi numerik.

Langkah pertama adalah mendefinisikan fungsi yang melakukan iterasi sebanyak n kali. Kami menambahkan semua parameter ke fungsi ini.

```
>function f(K,R,P,n) := iterate("x*(1+P/100)+R",K,n;P,R)[-1]
```

Iterasinya sama seperti di atas

$$x_{n+1} = x_n \cdot \left(1 + \frac{P}{100}\right) + R$$

Tapi kami tidak lagi menggunakan nilai global R dalam ekspresi kami. Fungsi seperti iterate() memiliki trik khusus di Euler. Anda dapat meneruskan nilai variabel dalam ekspresi sebagai parameter titik koma. Dalam hal ini P dan R.

Selain itu, kami hanya tertarik pada nilai terakhir. Jadi kita ambil indeks [-1].

Mari kita coba tes.

```
>f(5000,-200,3,47)
```

-19.83

Sekarang kita bisa menyelesaikan masalah kita.

```
>solve("f(5000,-200,x,50)",3)
```

3.15

Rutin memecahkan memecahkan ekspresi=0 untuk variabel x. Jawabannya adalah 3,15% per tahun. Kami mengambil nilai awal 3% untuk algoritma. Fungsi solve() selalu membutuhkan nilai awal.

Kita dapat menggunakan fungsi yang sama untuk menyelesaikan pertanyaan berikut: Berapa banyak yang dapat kita keluarkan per tahun sehingga modal awal habis setelah 20 tahun dengan asumsi tingkat bunga 3% per tahun.

```
>solve("f(5000,x,3,20)",-200)
```

-336.08

Perhatikan bahwa Anda tidak dapat memecahkan jumlah tahun, karena fungsi kami mengasumsikan n sebagai nilai integer.

Solusi Simbolik untuk Masalah Suku Bunga

Kita dapat menggunakan bagian simbolik dari Euler untuk mempelajari masalah tersebut. Pertama kita mendefinisikan fungsi onepay() kita secara simbolis.

```
>function op(K) &= K*q+R; $&op(K)
```

$$R + q K$$

Kita sekarang dapat mengulangi ini.

```
>$&op(op(op(op(K)))) , $&expand(%)
```

$$q^3 R + q^2 R + q R + R + q^4 K$$

Kami melihat sebuah pola. Setelah n periode yang kita miliki
lateks: $K_n = q^n K + R (1+q+\dots+q^{n-1}) = q^n K + \frac{q^n - 1}{q - 1} R$
Rumusnya adalah rumus untuk jumlah geometri, yang diketahui Maxima.

```
>&sum(q^k, k, 0, n-1); $& % = ev(%, simpsum)
```

$$\sum_{k=0}^{n-1} q^k = \frac{q^n - 1}{q - 1}$$

Ini agak rumit. Jumlahnya dievaluasi dengan bantuan "simpsum" untuk menguranginya menjadi hasil bagi.
Mari kita membuat fungsi untuk ini.

```
>function fs(K, R, P, n) &= (1+P/100)^n*K + ((1+P/100)^n-1)/(P/100)*R; $&fs(K, R, P, n)
```

$$\frac{100 \left(\left(\frac{P}{100} + 1\right)^n - 1\right) R}{P} + K \left(\frac{P}{100} + 1\right)^n$$

Fungsi tersebut melakukan hal yang sama seperti fungsi f kita sebelumnya. Tapi itu lebih efektif.

```
>longest f(5000,-200,3,47), longest fs(5000,-200,3,47)
```

```
-19.82504734650985  
-19.82504734652684
```

Kita sekarang dapat menggunakannya untuk menanyakan waktu n. Kapan modal kita habis? Dugaan awal kami adalah 30 tahun.

```
>solve("fs(5000,-330,3,x)",30)
```

20.51

Jawaban ini mengatakan bahwa itu akan menjadi negatif setelah 21 tahun.

Kita juga dapat menggunakan sisi simbolis Euler untuk menghitung formula pembayaran.

Asumsikan kita mendapatkan pinjaman sebesar K, dan membayar n pembayaran sebesar R (dimulai setelah tahun pertama) meninggalkan sisa hutang sebesar Kn (pada saat pembayaran terakhir). Rumus untuk ini jelas

```
>equ &= fs(K,R,P,n)=Kn; $&equ
```

$$\frac{100 \left(\left(\frac{P}{100} + 1 \right)^n - 1 \right) R}{P} + K \left(\frac{P}{100} + 1 \right)^n = Kn$$

Biasanya rumus ini diberikan dalam bentuk

$$i = \frac{P}{100}$$

```
>equ &= (equ with P=100*i); $&equ
```

$$\frac{(i+1)^n - 1}{i} R + (i+1)^n K = Kn$$

Kita dapat memecahkan tingkat R secara simbolis.

```
>$&solve(equ,R)
```

$$\left[R = \frac{i Kn - i (i+1)^n K}{(i+1)^n - 1} \right]$$

Seperti yang Anda lihat dari rumus, fungsi ini mengembalikan kesalahan titik mengambang untuk $i=0$. Euler tetap merencanakannya.

Tentu saja, kami memiliki batas berikut.

```
>$&limit(R(5000,0,x,10),x,0)
```

$$\lim_{x \rightarrow 0} R(5000, 0, x, 10)$$

Jelas, tanpa bunga kita harus membayar kembali 10 tarif 500.

Persamaan juga dapat diselesaikan untuk n. Kelihatannya lebih bagus, jika kita menerapkan beberapa penye-derhanaan untuk itu.

```
>fn &= solve(equ,n) | ratsimp; $&fn
```

$$\left[n = \frac{\log\left(\frac{R+iKn}{R+iK}\right)}{\log(i+1)} \right]$$

BAB 3

LATIHAN SOAL DARI BUKU ALJABAR DENGAN EMT

LATIHAN SOAL

Pilih minimal 5 soal dari setiap Latihan atau tipe soal (misalnya diantara soal-soal yang sudah saya blok). Jangan lupa tuliskan soalnya di teks komentar (dengan format LaTeX) dan beri penjelasan hasil output-nya. Ubah file notebook pekerjaan Anda menjadi file PDF menggunakan salah satu metode di atas.

R.2 Exercise set

Soal No 49 Menyederhanakan:

$$\left(\frac{24a^{10}b^{-8}c^7}{12a^6b^{-3}c^5} \right)^{-5}$$

```
> $& ((24*a^(10)*b^(-8)*c^7)/(12*a^6*b^(-3)*c^5)) ^ (-5)
```

$$\frac{b^{25}}{32 a^{20} c^{10}}$$

Soal No 50

Menyederhanakan:

$$\left(\frac{125p^{12}q^{-14}r^{22}}{25p^8q^6r^{-15}} \right)^{-4}$$

$$> \$ & ((125*p^{12}*q^{-14}*r^{22}) / 25*p^8*q^6*r^{-15})^{-4}$$

$$\frac{q^{32}}{625 p^{80} r^{28}}$$

Soal No 90

calculate

$$2^6 * 2^{-3} / 2^{10} / 2^{-8}$$

$$> 2^6 * 2^{-3} / 2^{10} / 2^{-8}$$

2

Soal No 91

Calculate

$$\left(\frac{4(8-6)^2 - 4*3 + 2*8}{3^1 + 9^0} \right)$$

$$> (4 * (8-6)^2 - 4 * 3 + 2 * 8) / (3^1 + 9^0)$$

5

Soal No 92

Calculate

$$\left(\frac{[4(8-6)^2 - 4](3 + 2 * 8)}{2^2(2^5 + 5)} \right)$$

$$> ((4 * (8-6)^2 - 4) * 3 + 2 * 8) / (3^1 + 9^0)$$

13

R.3 Exercise Set

Perform the indicated operations.

no 27

$$(x + 3)^2$$

```
> $&showev (' expand( (x+3)^2) )
```

$$\text{expand} \left((x + 3)^2 \right) = x^2 + 6x + 9$$

no 29

$$(y - 5)^2$$

```
> $&showev (' expand( (y-5)^2) )
```

$$\text{expand} \left((y - 5)^2 \right) = y^2 - 10y + 25$$

no 33

$$(2x + 3y)^2$$

```
> $&showev (' expand( (2*x+3*y)^2) )
```

$$\text{expand} \left((3y + 2x)^2 \right) = 9y^2 + 12xy + 4x^2$$

no 39

$$(3y + 4)(3y - 4)$$

```
> $&showev (' expand( (3*y+4) * (3*y-4) ) )
```

$$\text{expand} ((3y - 4)(3y + 4)) = 9y^2 - 16$$

no 42

$$(3x + 5y)(3x - 5y)$$

```
>$&showev('expand ((3*x + 5*y)*(3*x - 5*y)))
```

$$\text{expand } ((3x - 5y)(5y + 3x)) = 9x^2 - 25y^2$$

R.4 Exercise Set

Faktor Trinomial
Nomor 24

$$y^2 + 12y + 27$$

```
>$&solve(y^2+12*y+27)
```

$$[y = -9, y = -3]$$

Nomor 23

$$t^2 + 8t + 15$$

```
>$&solve(t^2+8*t+15)
```

$$[t = -3, t = -5]$$

Factor the difference of squares
Nomor 47

$$z^2 - 81$$

```
>$&solve(z^2-81)
```

$$[z = -9, z = 9]$$

Nomor 48

$$m^2 - 4$$

```
>$&solve(m^2-4)
```

$$[m = -2, m = 2]$$

Nomor 49

$$16x^2 - 9$$

```
> $&solve(16*x^2-9)
```

$$\left[x = -\frac{3}{4}, x = \frac{3}{4} \right]$$

R.5 Exercise Set

soal no 36

tentukan nilai y

$$y^2 - 4y - 45 = 0$$

```
> $&solve(y^2-4*y-45, y)
```

$$[y = 9, y = -5]$$

soal no 38

tentukan nilai y

$$t^2 + 6t = 0$$

```
> $&solve(t^2+ 6*t, t)
```

$$[t = -6, t = 0]$$

soal no 41

tentukan nilai x

$$x^2 + 100 = 20x$$

```
> $&solve(x^2+100=20*x, x)
```

$$[x = 10]$$

soal no 42
tentukan nilai y

$$y^2 + 25 = 10y$$

```
>$&solve(y^2+25=10*y,y)
```

$$[y = 5]$$

soal no 45
tentukan nilai y

$$3y^2 + 8y + 4 = 0$$

```
>$&solve(3*y^2 + 8*y + 4,y)
```

$$\left[y = -\frac{2}{3}, y = -2 \right]$$

soal no 47
tentukan nilai z

$$12z^2 + z = 6$$

```
>$&solve(12*z^2+z=6,z)
```

$$\left[z = -\frac{3}{4}, z = \frac{2}{3} \right]$$

soal no 60
tentukan nilai x

$$5x^2 - 75 = 0$$

```
>$&solve(5*x^2-75=0,x)
```

$$\left[x = -\sqrt{15}, x = \sqrt{15} \right]$$

R.6 Exercise Set

Nomor 9
Menyederhanakan

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

```
>\$&((x^2) / (x^2 - 4*x + 4)), \$&factor(%)
```

$$\frac{x^2}{(x - 2)^2}$$

$$\frac{x^2}{(x - 2)^2}$$

Nomor 11
Menyederhanakan

$$\frac{x^3 - 6x^2 + 9x}{x^3 - 3x^2}$$

```
>\$&((x^3 - 6*x^2 + 9*x) / (x^3 - 3*x^2)), \$&factor(%)
```

$$\frac{x - 3}{x}$$

Nomor 14
Menyederhanakan

$$\frac{2x^2 - 20x + 50}{10x^2 - 30x - 100}$$

```
>\$&((2*x^2 - 20*x + 50) / (10*x^2 - 30*x - 100)), \$&factor(%)
```

$$\frac{x - 5}{5(x + 2)}$$

$$\frac{x - 5}{5(x + 2)}$$

Nomor 15
Menyederhanakan

$$\frac{4 - x}{x^2 + 4x - 32}$$

```
>${&((4-x)/(x^2+4*x-32)), \$&factor(%)}
```

$$-\frac{1}{x+8}$$

$$-\frac{1}{x+8}$$

Nomor 16
Menyederhanakan

$$\frac{6-x}{x^2-36}$$

```
>${&((6-x)/(x^2-36)), \$&factor(%)}
```

$$-\frac{1}{x+6}$$

$$-\frac{1}{x+6}$$

Nomor 23

```
>${&(((m^2-n^2)/(r+s))/((m-n)/r+s)), \$&factor(%)}
```

$$\frac{(m-n)(n+m)r}{(s+r)(rs-n+m)}$$

$$\frac{(m-n)(n+m)r}{(s+r)(rs-n+m)}$$

Nomor 25

```
>${&(((3*x+12)/(2*x-8))/(((x+4)^2)/(x-4)^2)), \$&factor(%)}
```

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

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

Nomor 28

```
>${&(((c^3+8)/(c^2-4))/((c^2-2*c+4)/(c^2-4*c+4))), \$&factor(%)}
```

$$c - 2$$

REVIEW

Multiply Nomor 73

$$(a^n - b^n)^x$$

```
>function P(a,b,n,x) &= (a^n - b^n)^x; $&P(a,b,n,x)
```

$$(a^n - b^n)^x$$

```
>$&P(a,b,n,3), $&expand(%)
```

$$-b^{3n} + 3a^n b^{2n} - 3a^{2n} b^n + a^{3n}$$

Nomor 71

```
>function P(a,n) &= (t^a + t^{-a})^n; $&P(a,n)
```

$$\left(t^a + \frac{1}{t^a}\right)^n$$

```
>$&P(a,2), $&expand(%)
```

$$t^{2a} + \frac{1}{t^{2a}} + 2$$

Soal 39

```
>$&solve(9*x^2 - 30*x + 25, x)
```

$$\left[x = \frac{5}{3}\right]$$

soal nomor 41

```
>$&solve(18*x^2 - 3*x + 6, x)
```

$$\left[x = \frac{1 - \sqrt{47}i}{12}, x = \frac{\sqrt{47}i + 1}{12}\right]$$

soal nomor 48

```
>$_& solve((8-3*x)=(-7+2*x),x)
```

$$[x = 3]$$

soal nomor 70

```
>$_& '((x^{n+10})*(x^{n-4})) = expand(((x^{n+10})*(x^{n-4})))
```

$$(x^n - 4) (x^n + 10) = x^{2n} + 6x^n - 40$$

soal nomor 71

```
>$_& '((t^a+t^{-a})^2) = expand(((t^a+t^{-a})^2))
```

$$\left(t^a + \frac{1}{t^a}\right)^2 = t^{2a} + \frac{1}{t^{2a}} + 2$$

soal nomor 72

```
>$_& '((y^b-z^c)*(y^b+z^c)) = expand((y^b-z^c)*(y^b+z^c))
```

$$(y^b - z^c) (z^c + y^b) = y^{2b} - z^{2c}$$

soal nomor 73

```
>$_& '((a^n-b^n)^3) = expand((a^n-b^n)^3)
```

$$(a^n - b^n)^3 = -b^{3n} + 3a^n b^{2n} - 3a^{2n} b^n + a^{3n}$$

soal chapter R test nomor 32

```
>$_& '(((x^2+x-6)/(x^2+8*x+15))*((x^2-25)/(x^2-4*x+4))) = simplify(((x^2+x-6)/(x^2+8*x+15)))
```

$$\frac{(x^2 - 25) (x^2 + x - 6)}{(x^2 - 4x + 4) (x^2 + 8x + 15)} = simplify\left(\frac{(x^2 - 25) (x^2 + x - 6)}{(x^2 - 4x + 4) (x^2 + 8x + 15)}\right)$$

```
>$_& solve(((x^2+x-6)/(x^2+8*x+15))*((x^2-25)/(x^2-4*x+4)),x)
```

$$[x = 5]$$

soal nomor 33

```
>\$& ' ((x) / (x^2-1)) - ((3) / (x^2+4*x-5)) =simplify ((x) / (x^2-1)) - ((3) / (x^2+4*x-5))
```

$$\frac{x}{x^2 - 1} - \frac{3}{x^2 + 4x - 5} = \text{simplify} \left(\frac{x}{x^2 - 1} - \frac{3}{x^2 + 4x - 5} \right)$$

```
>\$&solve ((x) / (x^2-1)) - ((3) / (x^2+4*x-5))
```

$$[x = -3]$$

2.3 Exercise Set

Cari

$$(f \circ g)(x) \text{ dan } (g \circ f)(x)$$

dan domain nya !

$$1. f(x) = x + 3, \quad g(x) = x - 3$$

$$(f \circ g)(x) =$$

```
>\$&gx:=x-3; \$&fx:=gx+3; \$&fx
```

$$x$$

dengan domainnya

$$D_{f \circ g} = \{x \in \mathbb{R}\}$$

$$(g \circ f)(x) =$$

```
>\$&fx:=x+3; \$&gx:=fx-3; \$&gx
```

$$x$$

dengan domainnya

$$D_{g \circ f} = \{x \in \mathbb{R}\}$$

$$2. f(x) = 4/(1 - 5x) , g(x) = 1/x$$

$$(f \circ g)(x) =$$

```
> $gx:=1/x; $fx:=4/(1-5*gx); $fx
```

$$\frac{4}{1 - \frac{5}{x}}$$

dengan domain

$$D_{f \circ g} = \{x \in \mathbb{R} | x \neq 0 \cup x \neq 5\}$$

$$(g \circ f)(x) =$$

```
> $fx:=4/(1-5*x); $gx:=1/fx; $gx
```

$$\frac{1 - 5x}{4}$$

dengan domainnya

$$D_{g \circ f} = \{x \in \mathbb{R}\}$$

Diberikan fungsi

$$f(x) = 3x + 1, g(x) = x^2 - 2x - 6, h(x) = x^3$$

cari

$$3. (f \circ g)(1/3)$$

```
> $x:=1/3; $gx:=x^2-2*x-6; $fx:=3*gx+1; $fx
```

$$-\frac{56}{3}$$

$$4. (g \circ h)(1/2)$$

```
> $x:=1/2; $hx:=x^3; $gx:=hx^2-2*hx-6; $gx
```

$$-\frac{399}{64}$$

$$5. (g \circ g)(-2)$$

```
>$x:=-2; $gx:=x^2-2*x-6; $gx:=gx^2-2*gx-6; $gx
```

$$-6$$

3.1 Exercise Set

Use the quadratic formula to find exact solutions

Nomor 37

$$x^2 - 2 = 15$$

```
>$&solve(x^2-2= 15,x)
```

Maxima said:

```
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);
```

Error in:

```
$&solve(x^2-2= 15,x) ...  
^
```

Nomor 39

$$5m^2 + 3m = 2$$

```
>$&solve(5*m^2+3*m=2,m)
```

$$\left[m = \frac{2}{5}, m = -1 \right]$$

Nomor 40

$$2y^2 - 3y - 2 = 0$$

```
>$&solve(2*y^2-3*y-2=0,y)
```

$$\left[y = -\frac{1}{2}, y = 2 \right]$$

Solve.

Nomor 83

$$y^4 + 4y^2 - 5 = 0$$

```
> $&solve(y^4+4*y^2-5=0, y)
```

$$\left[y = -1, y = 1, y = -\sqrt{5}i, y = \sqrt{5}i \right]$$

Nomor 84

$$y^4 - 15y^2 - 16 = 0$$

```
> $&solve(y^4-15*y^2-16=0, y)
```

$$\left[y = -i, y = i, y = -4, y = 4 \right]$$

3.4 Exercise Set Solve

Nomor 1

```
> $&solve(1/4+1/5=1/t, t)
```

$$\left[t = \frac{20}{9} \right]$$

Nomor 2

```
> $&solve(1/3-5/6=1/x, x)
```

Maxima said:
solve: all variables must not be numbers.
-- an error. To debug this try: debugmode(true);

Error in:
\$&solve(1/3-5/6=1/x, x) ...
^

Nomor 6

```
> $&solve(1/t+1/2*t+1/3*t=5, t)
```

$$\left[t = \frac{15 - \sqrt{195}}{5}, t = \frac{\sqrt{195} + 15}{5} \right]$$

Nomor 7

```
>${&}solve(5/3*x+2=3/2*x, x)
```

Maxima said:
solve: all variables must not be numbers.
-- an error. To debug this try: debugmode(true);

Error in:
 \${&}solve(5/3*x+2=3/2*x, x) ...
 ^

NOmor 10

```
>${&}solve(x-12/x=1, x)
```

Maxima said:
solve: all variables must not be numbers.
-- an error. To debug this try: debugmode(true);

Error in:
 \${&}solve(x-12/x=1, x) ...
 ^

Nomor 18

```
>${&}solve(3*y+5/y^2+5*y +y+4/y+5=y+1/y, y)
```

$$y = -\frac{47 \left(\frac{\sqrt{3} i}{2} - \frac{1}{2}\right)}{576 \left(\frac{\sqrt{35539}}{128 3^{\frac{3}{2}}} - \frac{3905}{13824}\right)^{\frac{1}{3}}} + \left(\frac{\sqrt{35539}}{128 3^{\frac{3}{2}}} - \frac{3905}{13824}\right)^{\frac{1}{3}} \left(-\frac{\sqrt{3} i}{2} - \frac{1}{2}\right) - \frac{5}{24}, y = \left(\frac{\sqrt{35539}}{128 3^{\frac{3}{2}}} - \frac{3905}{13824}\right)^{\frac{1}{3}} \left(\frac{\sqrt{3} i}{2} - \frac{1}{2}\right) - \frac{5}{24}$$

3.5 Exercise Set

```
>&load(fourier_elim)
```

C:/Program Files/Euler x64/maxima/share/maxima/5.35.1/share/fourier_elim

Nomor 44

```
>${&}fourier_elim([4*x]>20, [x]) // 4*x > 20
```

Maxima said:
Function "\$elim" expects a symbol, instead found -2
-- an error. To debug this try: debugmode(true);

Error in:
 \${&}fourier_elim([4*x]>20, [x]) // 4*x > 20 ...
 ^

Nomor 45

```
> $&fourier_elim([x+8]<9, [x]) // x+8<9
```

Maxima said:
Function "\$elim" expects a symbol, instead found -2
-- an error. To debug this try: debugmode(true);

Error in:
\$&fourier_elim([x+8]<9, [x]) // x+8<9 ...
^

Nomor 47

```
> $&fourier_elim([x+8]>= 9, [x]) // x+8 >=9
```

Maxima said:
Function "\$elim" expects a symbol, instead found -2
-- an error. To debug this try: debugmode(true);

Error in:
\$&fourier_elim([x+8]>= 9, [x]) // x+8 >=9 ...
^

Nomor 52

```
> $&fourier_elim([3*x+4]<13, [x]) // 3*x+4<13
```

Maxima said:
Function "\$elim" expects a symbol, instead found -2
-- an error. To debug this try: debugmode(true);

Error in:
\$&fourier_elim([3*x+4]<13, [x]) // 3*x+4<13 ...
^

Nomor 62

```
> $&fourier_elim([3*x+5]<0, [x]) // 3*x+5<0
```

Maxima said:
Function "\$elim" expects a symbol, instead found -2
-- an error. To debug this try: debugmode(true);

Error in:
\$&fourier_elim([3*x+5]<0, [x]) // 3*x+5<0 ...
^

Nomor 8

```
>${&}solve(3/3*x+4 + 2/x-1 =2,x)
```

Maxima said:
solve: all variables must not be numbers.
-- an error. To debug this try: debugmode(true);

Error in:
\${&}solve(3/3*x+4 + 2/x-1 =2,x) ...
^

```
>${&}load(fourier_elim)
```

Nomor 11

```
>${&}fourier_elim([x+4]=7,[x])//x+4=7
```

Maxima said:
Function "\$elim" expects a symbol, instead found -2
-- an error. To debug this try: debugmode(true);

Error in:
\${&}fourier_elim([x+4]=7,[x])//x+4=7 ...
^

Nomor 12

```
>${&}fourier_elim([4*y-3]=5,[x])//4*y-3=5
```

Maxima said:
Function "\$elim" expects a symbol, instead found -2
-- an error. To debug this try: debugmode(true);

Error in:
\${&}fourier_elim([4*y-3]=5,[x])//4*y-3=5 ...
^

Solve

Nomor 13

```
>${&}fourier_elim([x+3]<=4,[x])//x+3<=4
```

Maxima said:
Function "\$elim" expects a symbol, instead found -2
-- an error. To debug this try: debugmode(true);

Error in:
\${&}fourier_elim([x+3]<=4,[x])//x+3<=4 ...
^

Nomor 15

```
> $&fourier_elim([x+5]>2, [x]) //x+5>2
```

Maxima said:
Function "\$elim" expects a symbol, instead found -2
-- an error. To debug this try: debugmode(true);

Error in:
\$&fourier_elim([x+5]>2, [x]) //x+5>2 ...
^

Nomor 19

```
> $&solve(x^2+4*x =1, x)
```

Maxima said:
solve: all variables must not be numbers.
-- an error. To debug this try: debugmode(true);

Error in:
\$&solve(x^2+4*x =1, x) ...
^

4.1 Exercise Set

Use the substitution to determine whether 2,3 and -1 are zeros of

Nomor 23

```
> function P(x) &= (x^3-9*x^2+14*x+24); $&P(x)
```

-48

```
>P(4)
```

-48

```
>P(5)
```

-48

```
>P(-2)
```

-48

Jadi hasil substitusi yang menghasilkan persamaan mempunyai nilai nol adalah dengan mensubtsisusi angka 4

Nomor 24

```
>function P(x) &= (2*x^3-3*x^2+x+6); $&P(x)
```

-24

```
>P(2)
```

-24

```
>P(3)
```

-24

```
>P(-1)
```

-24

Jadi hasil substitusi yang menghasilkan persamaan mempunyai nilai nol adalah dengan mensubtsisusi angka -1

Nomor 25

```
>function P(x) &= (x^4-6*x^3+8*x^2+6*x-9); $&P(x)
```

75

```
>P(2)
```

75

```
>P(3)
```

75

```
>P(-1)
```

75

Jadi hasil substitusi yang menghasilkan persamaan mempunyai nilai nol adalah dengan mensubstisusi angka 3 dan -1

Nomor 37

```
> $&solve(x^4-4*x^2+3, x)
```

Maxima said:

```
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);
```

Error in:

```
$&solve(x^4-4*x^2+3, x) ...  
^
```

Nomor 39

```
> $&solve(x^3+3*x^2-x-3, x)
```

Maxima said:

```
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);
```

Error in:

```
$&solve(x^3+3*x^2-x-3, x) ...  
^
```

4.3 Exercise Set

Nomor 1

For the function

$$f(x) = x^4 - 6x^3 + x^2 + 24x - 20$$

use long division to determine whether each of the following is a factor of $f(x)$

- a) $x+1$
- b) $x-2$
- c) $x + 5$

```
> function f(x) &= (x^4-6*x^3+x^2+24*x-20); $&f(x)
```

0

```
> $&f(x+1), $&expand(%)
```

0

```
>${&f(x-2)}, ${&expand(%)}
```

0

```
>${&f(x+5)}, ${&expand(%)}
```

0

Nomor 23

Use synthetic division to find the function values.

$$f(x) = x^3 - 6x^2 + 11x - 6$$

find f(1), f(-2), dan f(3)

```
>function f(x) &= (x^3-6*x^2+11*x-6); ${&f(x)}
```

-60

```
>f(1)
```

-60

```
>f(-2)
```

-60

```
>f(3)
```

-60

Nomor 24

$$f(x) = x^3 + 7x^2 - 12x - 3$$

find $f(-3), f(-2)$, dan $f(1)$

```
>function f(x) &= (x^3+7*x^2-12*x-3); $&f(x)
```

41

```
>f(-3)
```

41

```
>f(-2)
```

41

```
>f(1)
```

41

Nomor 25

$$f(x) = x^4 - 3x^2 + 2x + 8$$

find $f(-1), f(4)$ dan $f(-5)$

```
>function f(x) &= (x^4-3*x^3+2*x+8); $&f(x)
```

44

```
>f(-1)
```

44

```
>f(4)
```

44

```
>f(-5)
```

44

Factor the polynomial function $f(x)$. Then Solve the equation $f(x)=0$
Nomor 39

$$f(x) = x^3 + 4x^2 + x - 6$$

```
>fx &= (x^3+4*x^2+x-6=0); $&fx
```

$$0 = 0$$

```
>$&factor((fx,x^3+4*x^2+x-6=0))
```

$$0 = 0$$

```
>$&solve(x^3+4*x^2+x-6=0,x)
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$&solve(x^3+4*x^2+x-6=0,x) ...  
^
```

Nomor 40

$$f(x) = x^3 + 5x^2 - 2x - 24$$

```
>fx &= (x^3+5*x^2-2*x-24=0); $&fx
```

$$-8 = 0$$

```
>$&factor((fx,x^3+5*x^2-2*x-24=0))
```

$$-8 = 0$$

```
>$&solve(x^3+5*x^2-2*x-24=0,x)
```

```

Maxima said:
solve: all variables must not be numbers.
-- an error. To debug this try: debugmode(true);

Error in:
$&solve(x^3+5*x^2-2*x-24=0,x) ...
^

```

Mid-Chapter Mixed Review

Use synthetic division to find the function values
Nomor 18

$$g(x) = x^3 - 9x^2 + 4x - 10$$

find $g(-5)$

```
>function g(x) &= (x^3-9*x^2+4*x-10); $&g(x)
```

-62

```
>g(-5)
```

-62

Nomor 19

$$f(x) = 20x^2 - 40x$$

find $f(1/2)$

```
>function f(x) &= (20*x^2-40*x); $&f(x)
```

160

```
>f(1/2)
```

160

Using synthetic division, determine whether the numbers are zeros of the polynomial function.

Nomor 22

-1,5;

$$f(x) = x^6 - 35x^4 + 259x^2 - 225$$

```
>function f(x) &= (x^6-35*x^4+259*x^2-225); $&f(x)
```

315

```
>f(-1.5)
```

315

Factor the polynomial function $f(x)$. Then solve the equation $f(x) = 0$.

Nomor 23

$$h(x) = x^3 - 2x^2 - 55x + 56$$

```
>hx &= (x^3-2*x^2-55*x+56=0); $&hx
```

150 = 0

```
>$&factor((hx,x^3-2*x^2-55*x+56=0))
```

150 = 0

```
>$&solve(x^3-2*x^2-55*x+56=0,x)
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$&solve(x^3-2*x^2-55*x+56=0,x) ...  
^
```

Nomor 24

$$g(x) = x^4 - 2x^3 - 13x^2 + 14x + 24$$

```
>gx &= (x^4-2*x^3-13*x^2+14*x+24=0); $&gx
```

$$-24 = 0$$

```
>$&factor((gx,x^4-2*x^3-13*x^2+14*x+24=0))
```

$$-24 = 0$$

```
>$&solve(x^4-2*x^3-13*x^2+14*x+24=0,x)
```

Maxima said:
solve: all variables must not be numbers.
-- an error. To debug this try: debugmode(true);

Error in:
\$&solve(x^4-2*x^3-13*x^2+14*x+24=0,x) ...
^

BAB 4

PENGGUNAAN SOFTWARE EMT UNTUK PLOT 2D

TUGAS INDIVIDU

Menggambar Grafik 2D dengan EMT

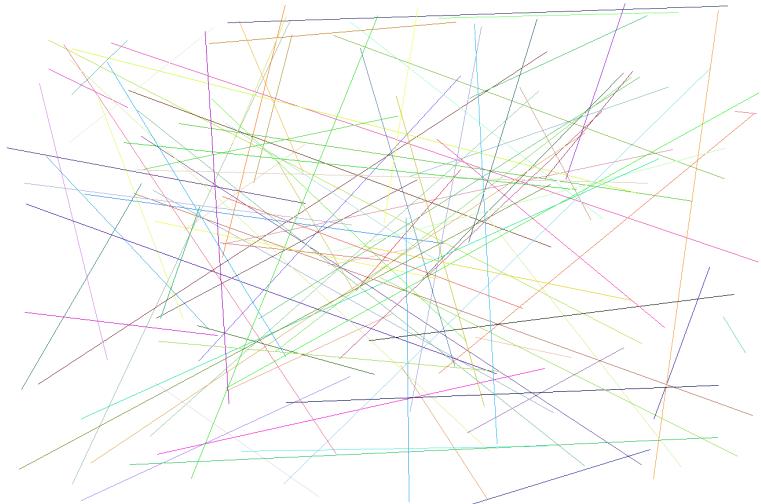
Notebook ini menjelaskan tentang cara menggambar berbagai kurva dan grafik 2D dengan software EMT. EMT menyediakan fungsi plot2d() untuk menggambar berbagai kurva dan grafik dua dimensi (2D).

Plot Dasar

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

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

```
>clg; // clear screen
>window(0,0,1024,1024); // use all of the window
>setplot(0,1,0,1); // set plot coordinates
>hold on; // start overwrite mode
>n=100; X=random(n,2); Y=random(n,2); // get random points
>colors=rgb(random(n),random(n),random(n)); // get random colors
>loop 1 to n; color(colors[#]); plot(X[#],Y[#]); end; // plot
>hold off; // end overwrite mode
>insimg; // insert to notebook
```



```
>reset;
```

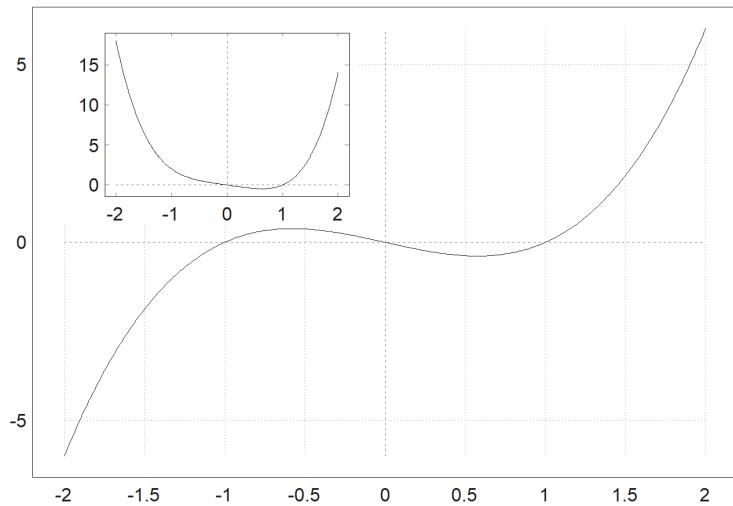
Grafik perlu ditahan, karena perintah plot() akan menghapus jendela plot.

Untuk menghapus semua yang kami lakukan, kami menggunakan reset().

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

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

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



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

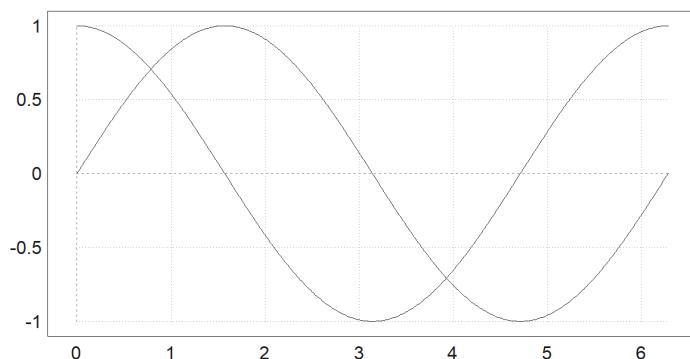
Plot dengan banyak angka dicapai dengan cara yang sama. Ada fungsi figure() utilitas untuk ini.

Aspek Plot

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

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

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



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

Fungsi reset() mengembalikan default plot termasuk rasio aspek.

Plot 2D di Euler

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

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

Euler dapat memplot plot 2D dari

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

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

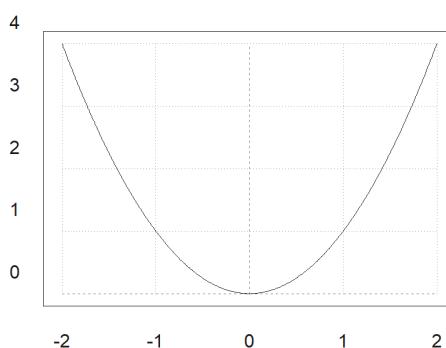
Plot Ekspresi atau Variabel

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

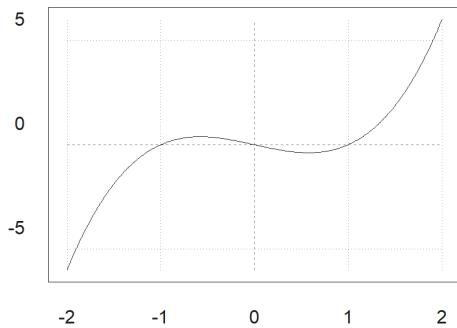
Berikut adalah contoh paling dasar, yang menggunakan rentang default dan menetapkan rentang y yang tepat agar sesuai dengan plot fungsi.

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

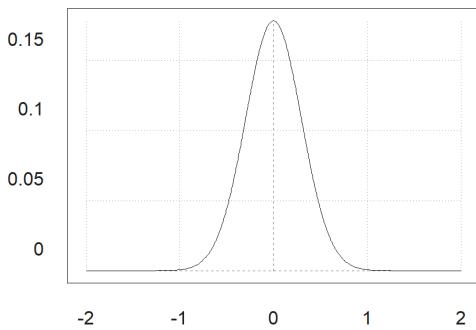
```
>plot2d("x^2");
```



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



```
>a:=5.6; plot2d("exp(-a*x^2)/a"); insimg(30); // menampilkan gambar hasil plot setinggi 250px
```

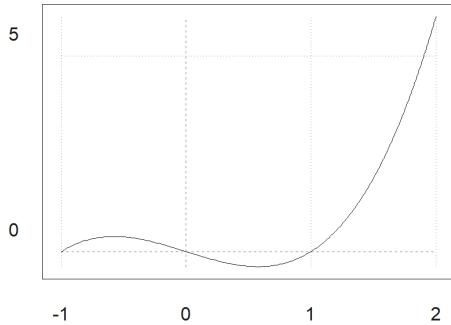


Dari beberapa contoh sebelumnya Anda dapat melihat bahwa Gambaran gambar plot menggunakan sumbu X dengan rentang nilai dari -2 sampai dengan 2. Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan nilai batas X (dan Y) di belakang ekspresi yang digambar.

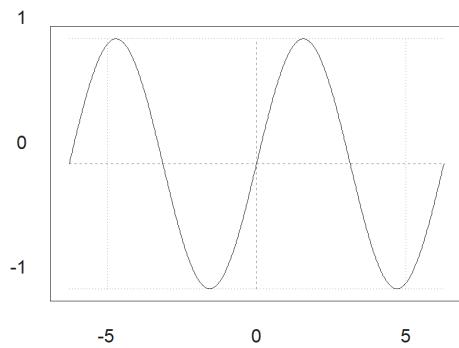
Rentang plot diatur dengan parameter yang ditetapkan berikut:

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

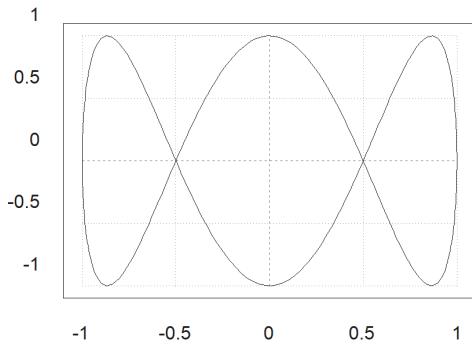
```
>plot2d("x^3-x",-1,2):
```



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



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



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

Dalam opsi, plot dapat diatur untuk muncul

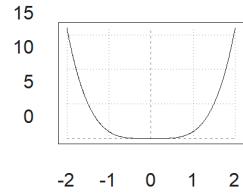
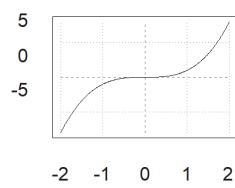
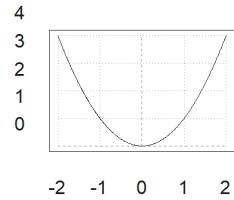
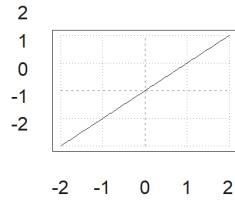
- di jendela terpisah yang dapat diubah ukurannya,
- di jendela buku catatan.

Lebih banyak gaya dapat dicapai dengan perintah plot tertentu.

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

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

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



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

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

-6
—
-2101

-6
—
-2101

-5
—
-2101

-6
—
-2101

-6
—
-2101

-6
—
-2101

-5
—
-21 1

-5
—
-21 1

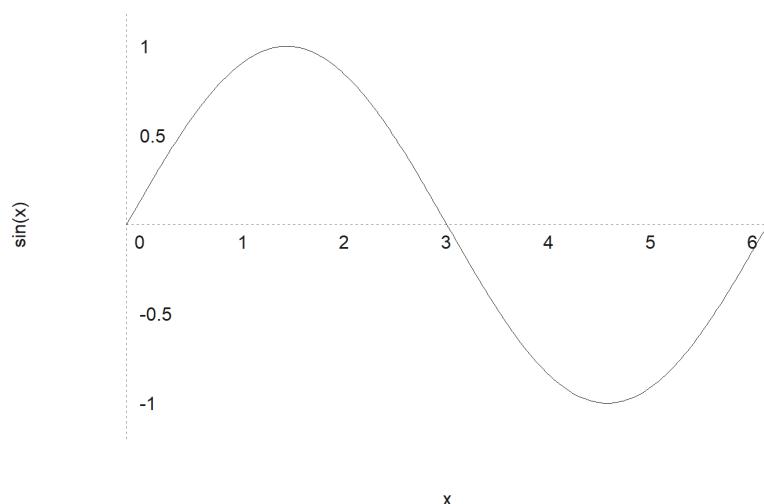
-6
—
-2101

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

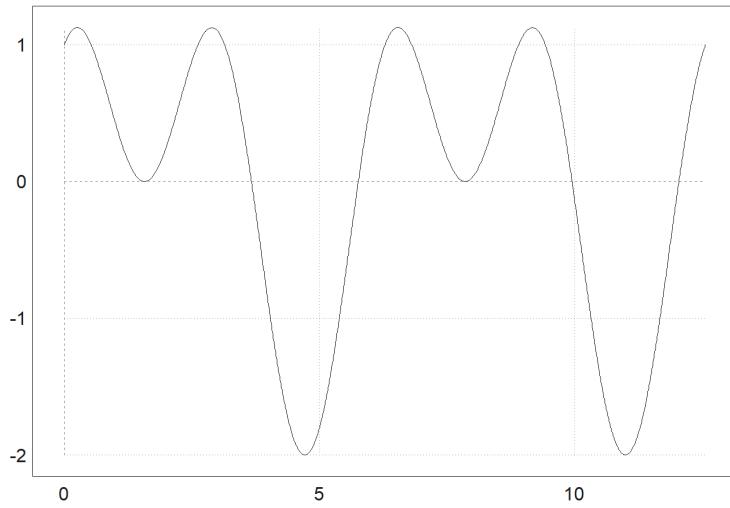
Atau, a, b, c, d dapat ditentukan sebagai parameter yang ditetapkan sebagai a=... dll.

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

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



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

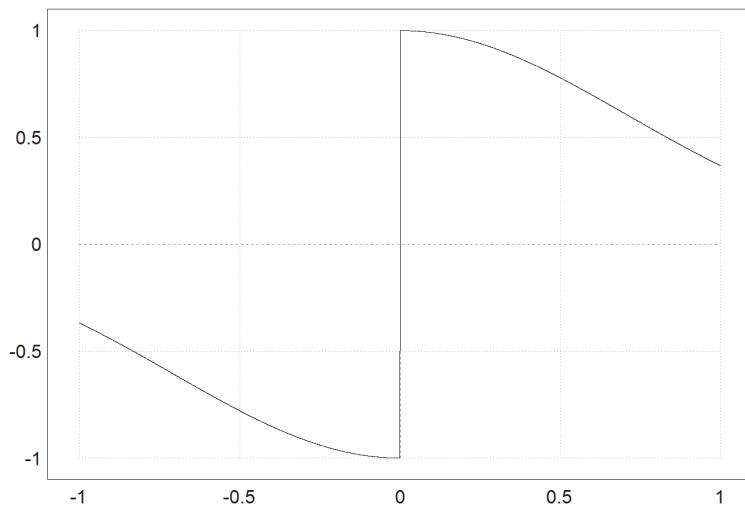


Gambar yang dihasilkan dengan memasukkan plot ke dalam jendela teks disimpan di direktori yang sama dengan buku catatan, secara default di subdirektori bernama "gambar". Mereka juga digunakan oleh ekspor HTML.

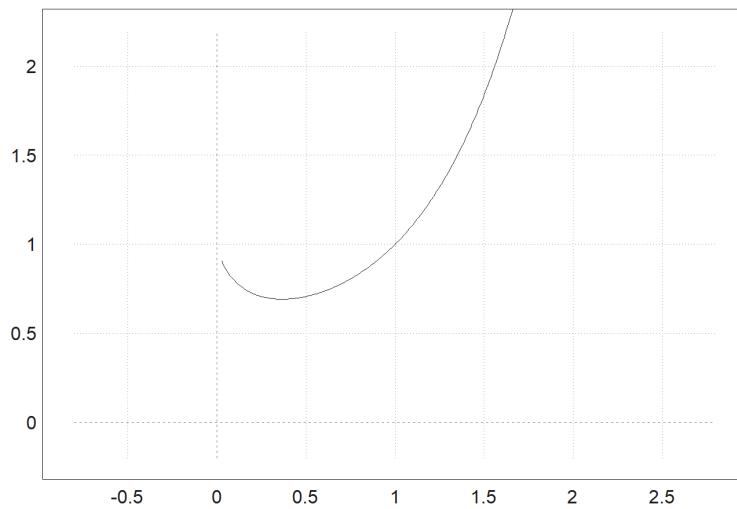
Anda cukup menandai gambar apa saja dan menyalinnya ke clipboard dengan Ctrl-C. Tentu saja, Anda juga dapat mengekspor grafik saat ini dengan fungsi di menu File.

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

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

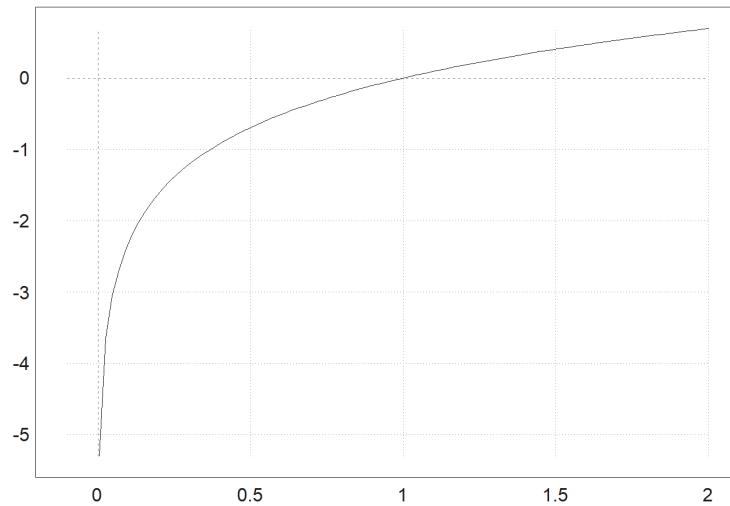


```
>plot2d("x^x", r=1.2, cx=1, cy=1):
```



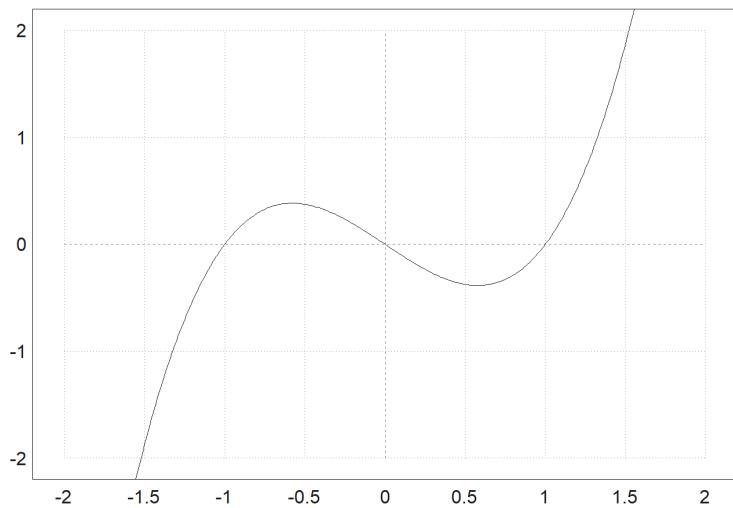
Perhatikan bahwa x^x tidak didefinisikan untuk $x \leq 0$. Fungsi plot2d menangkap kesalahan ini, dan mulai merencanakan segera setelah fungsi didefinisikan. Ini berfungsi untuk semua fungsi yang mengembalikan NAN keluar dari jangkauan definisinya.

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

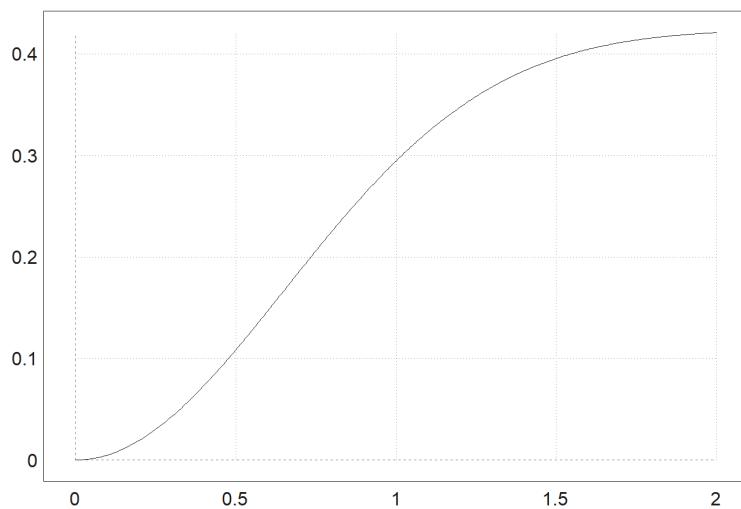


Parameter square=true (atau >square) memilih y-range secara otomatis sehingga hasilnya adalah jendela plot persegi. Perhatikan bahwa secara default, Euler menggunakan ruang persegi di dalam jendela plot.

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

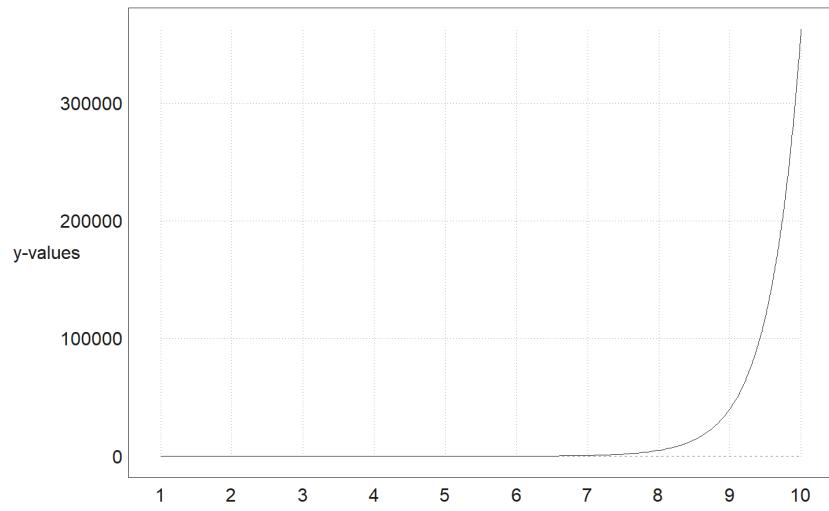


```
>plot2d(''integrate("sin(x)*exp(-x^2)", 0, x)'', 0, 2): // plot integral
```



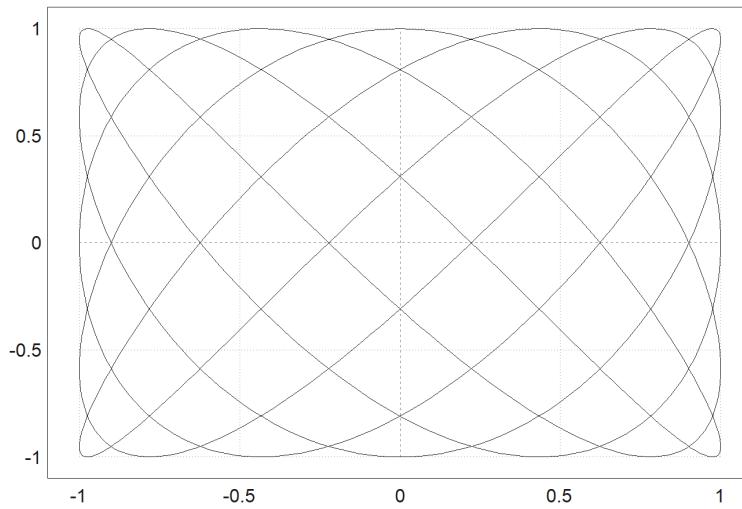
Jika Anda membutuhkan lebih banyak ruang untuk label-y, panggil shrinkwindow() dengan parameter yang lebih kecil, atau tetapkan nilai positif untuk "lebih kecil" di plot2d().

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

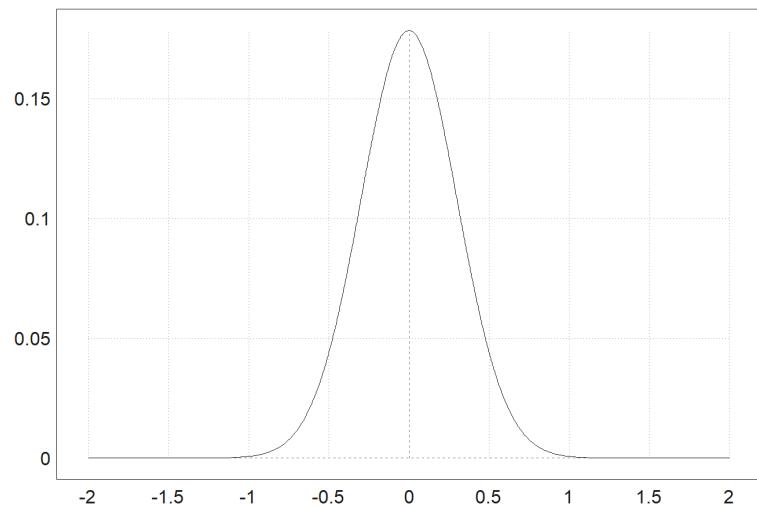


Ekspresi simbolik juga dapat digunakan, karena disimpan sebagai ekspresi string sederhana.

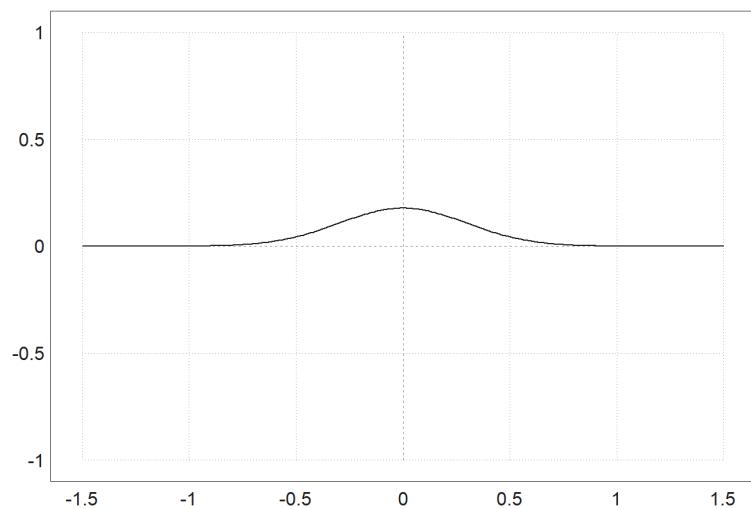
```
>x=linspace(0,2pi,1000); plot2d(sin(5x),cos(7x)):
```



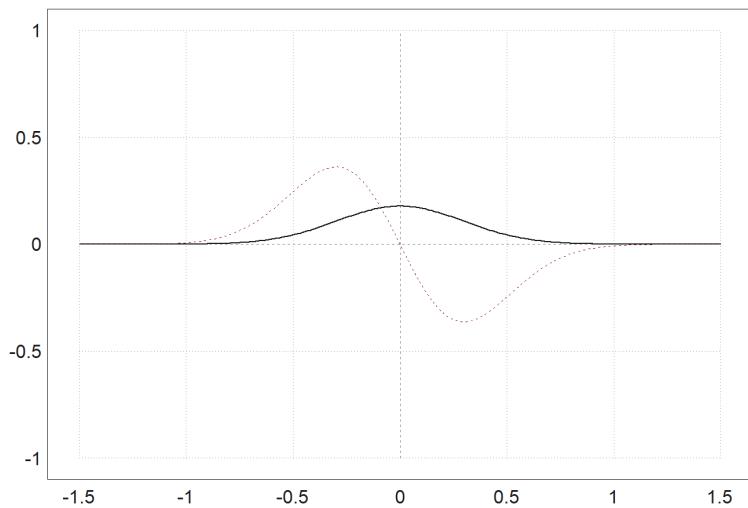
```
>a:=5.6; expr &= exp(-a*x^2)/a; // define expression
>plot2d(expr,-2,2); // plot from -2 to 2
```



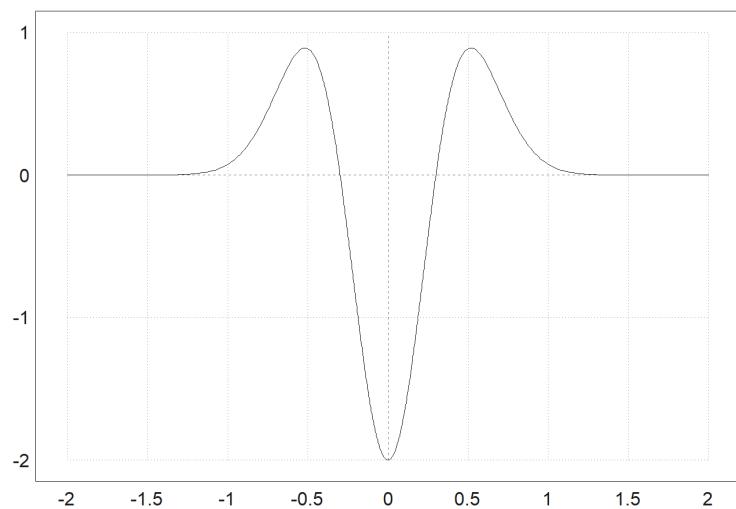
```
>plot2d(expr,r=1,thickness=2): // plot in a square around (0,0)
```



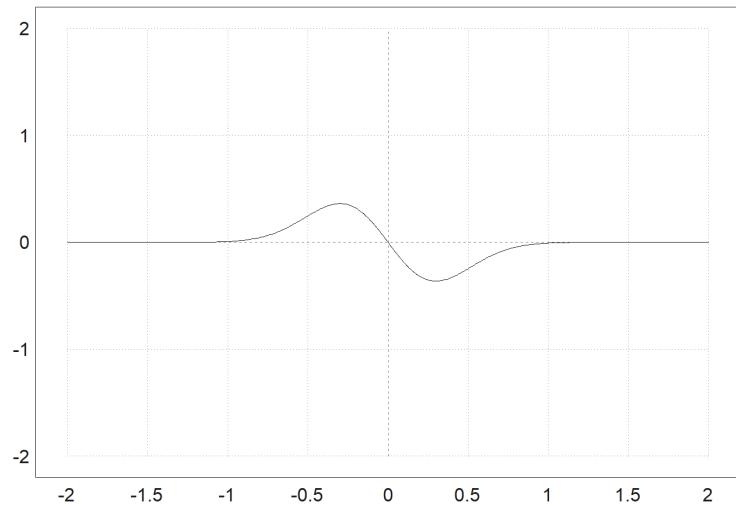
```
>plot2d(&diff(expr,x),>add,style="--",color=red): // add another plot
```



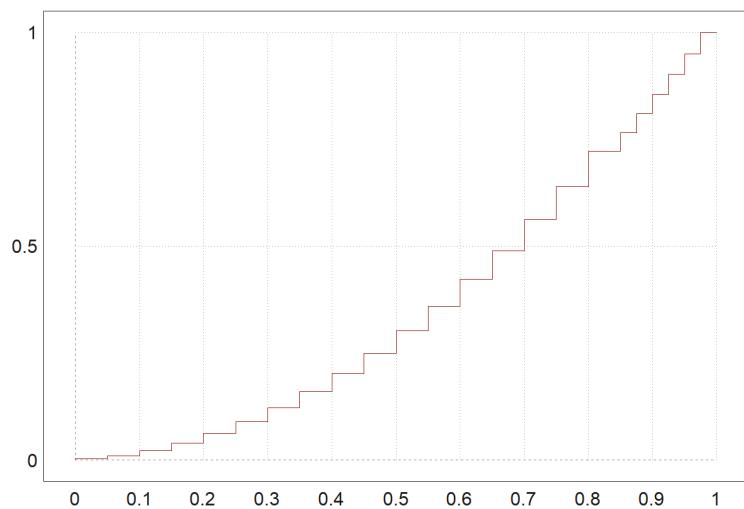
```
>plot2d(&diff(expr,x,2),a=-2,b=2,c=-2,d=1): // plot in rectangle
```



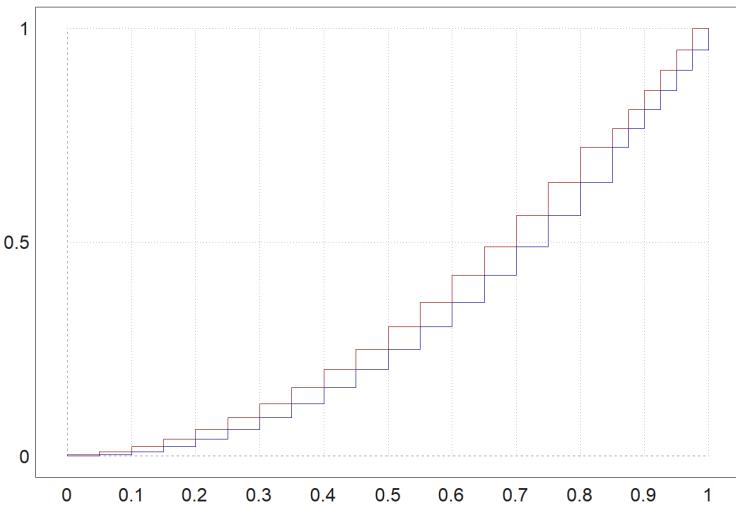
```
>plot2d(&diff(expr,x),a=-2,b=2,>square): // keep plot square
```



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



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

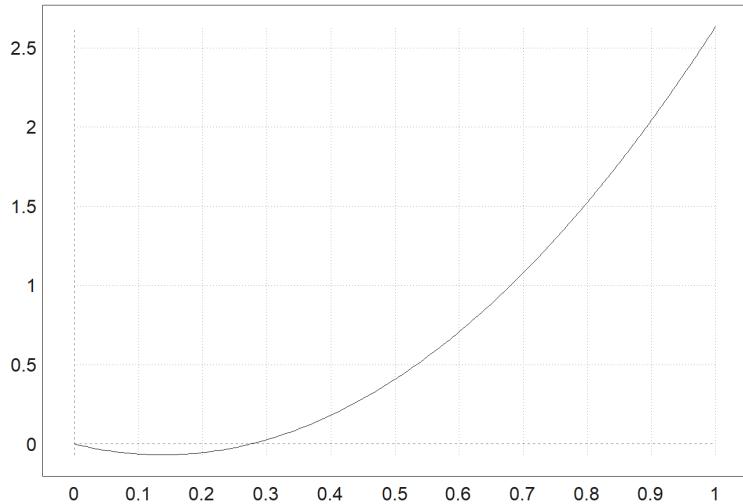


Fungsi dalam satu Parameter

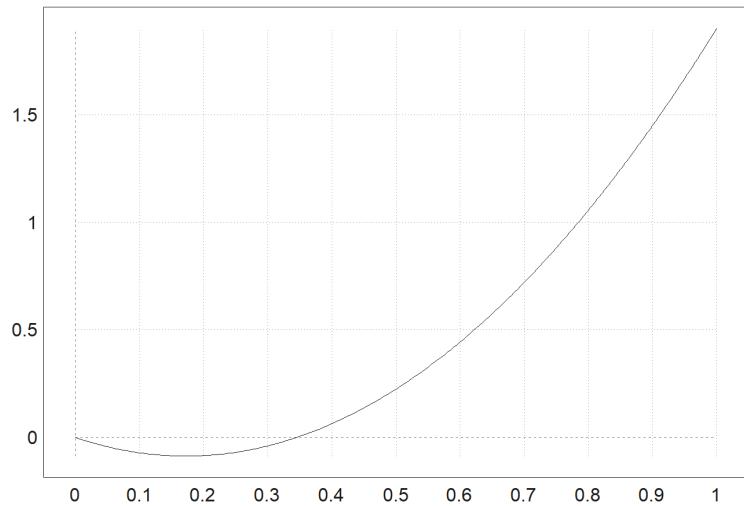
Fungsi plot yang paling penting untuk plot planar adalah `plot2d()`. Fungsi ini diimplementasikan dalam bahasa Euler dalam file "plot.e", yang dimuat di awal program.

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

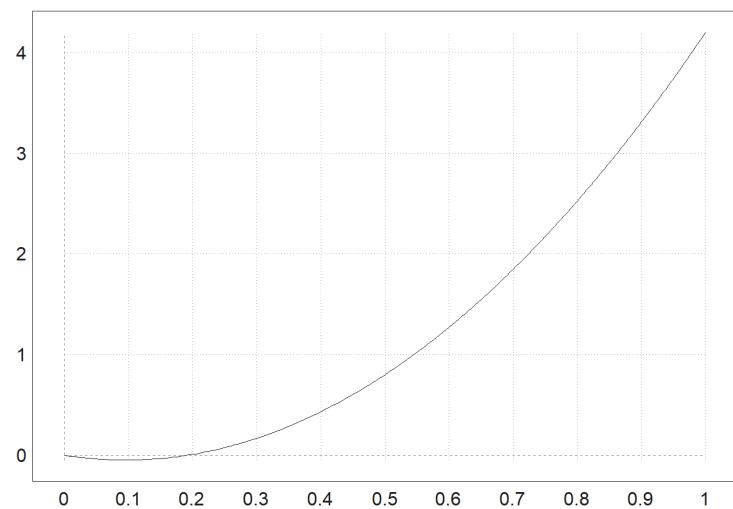
```
>function f(x,a) := x^2/a+a*x^2-x; // define a function
>a=0.3; plot2d("f",0,1;a); // plot with a=0.3
```



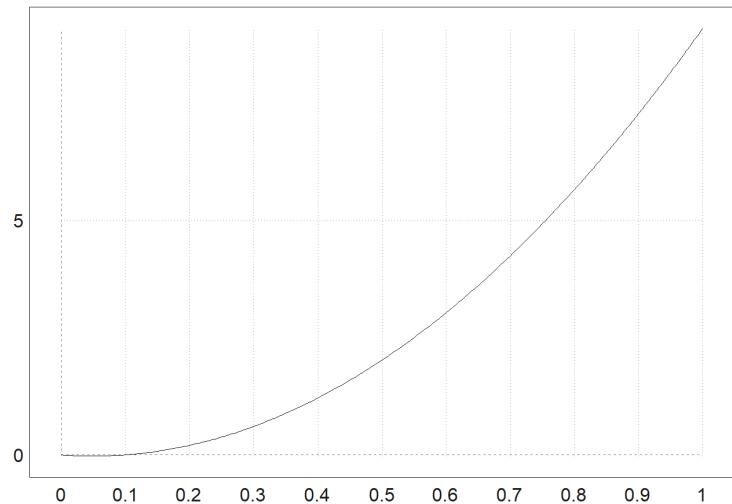
```
>plot2d("f",0,1;0.4); // plot with a=0.4
```



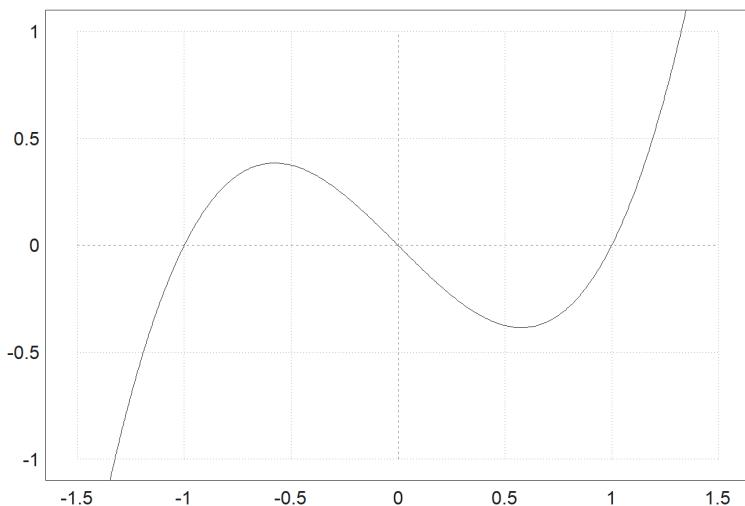
```
>plot2d({{"f",0.2}},0,1); // plot with a=0.2
```



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



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



Berikut adalah ringkasan dari fungsi yang diterima

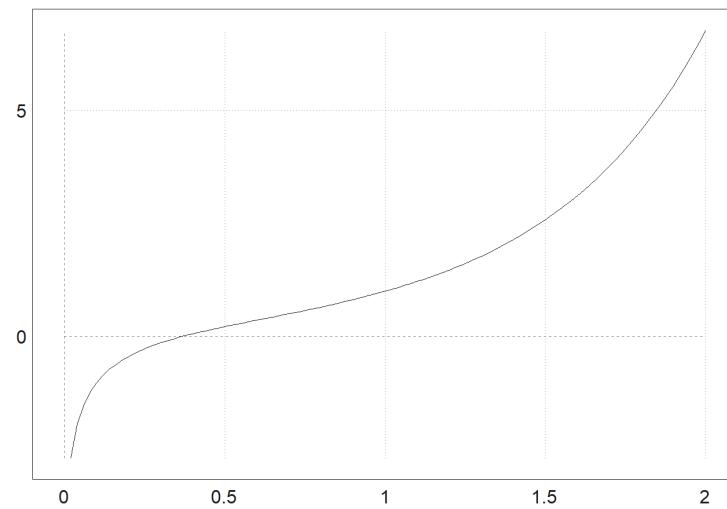
- ekspresi atau ekspresi simbolik dalam x
- fungsi atau fungsi simbolis dengan nama sebagai "f"
- fungsi simbolis hanya dengan nama f

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

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

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

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

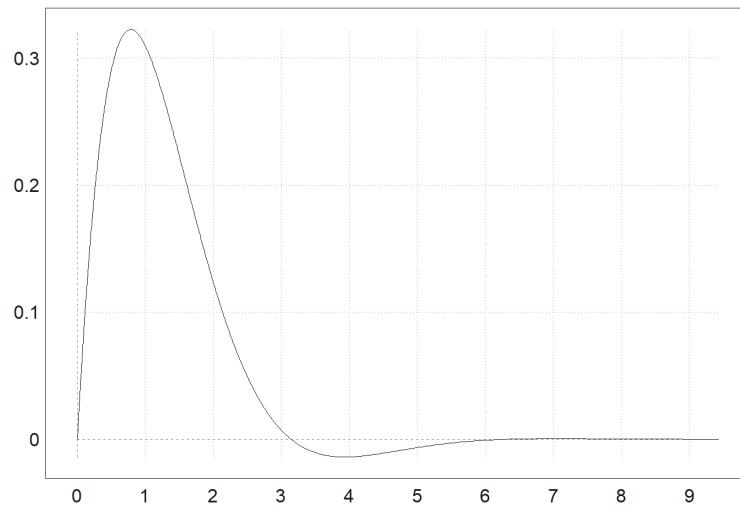


Tentu saja, untuk ekspresi atau ekspresi simbolik, nama variabel sudah cukup untuk memplotnya.

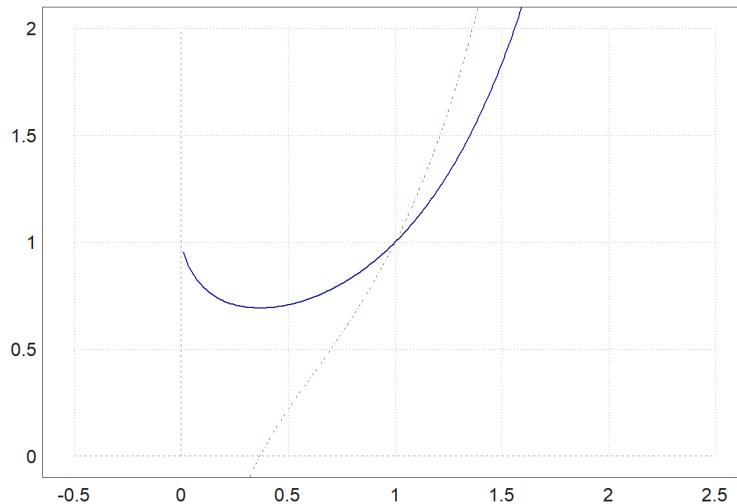
```
>expr &= sin(x)*exp(-x)
```

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

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



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



Untuk gaya garis ada berbagai pilihan.

- gaya="...". Pilih dari "-", "--", "-.", ".-", "-.-", "-.-".

- warna: Lihat di bawah untuk warna.

- ketebalan: Default adalah 1.

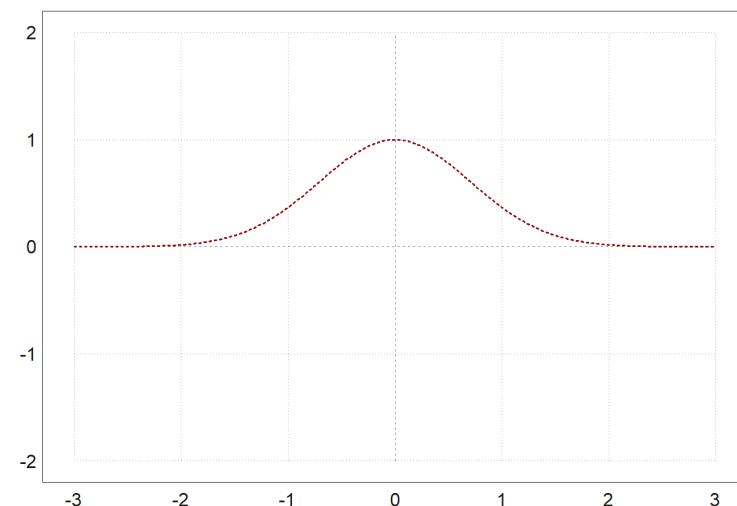
Warna dapat dipilih sebagai salah satu warna default, atau sebagai warna RGB.

- 0.15: indeks warna default.

- konstanta warna: putih, hitam, merah, hijau, biru, cyan, zaitun, abu-abu muda, abu-abu, abu-abu tua, oranye, hijau muda, pirus, biru muda, oranye terang, kuning

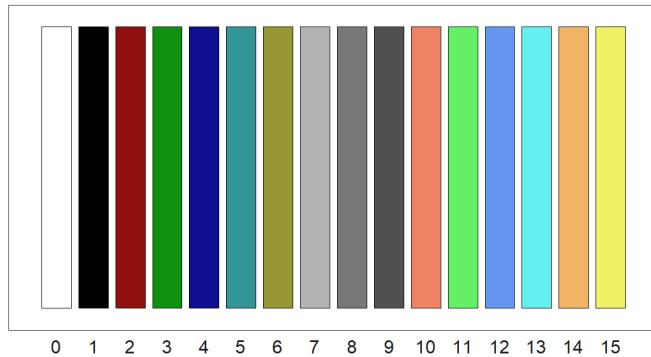
- rgb(merah, hijau, biru): parameter adalah real dalam [0,1].

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



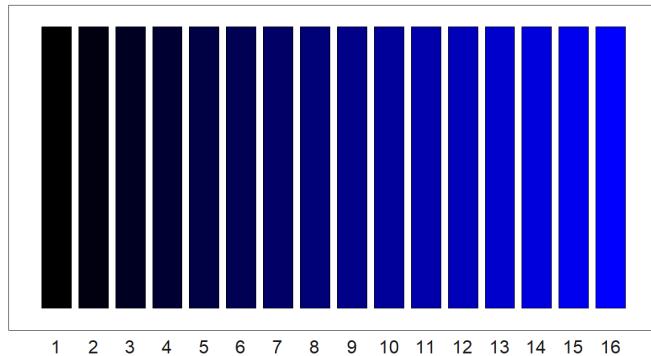
Berikut adalah tampilan warna EMT yang telah ditentukan sebelumnya.

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



But you can use any color.

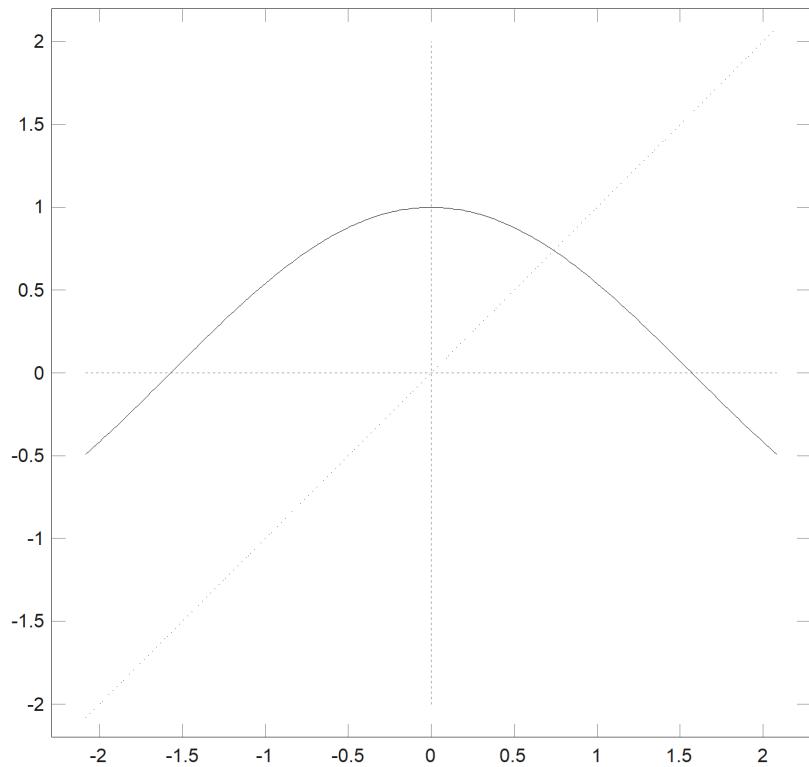
```
>columnsplot(ones(1,16),grid=0,color=rgb(0,0,linspace(0,1,15))):
```



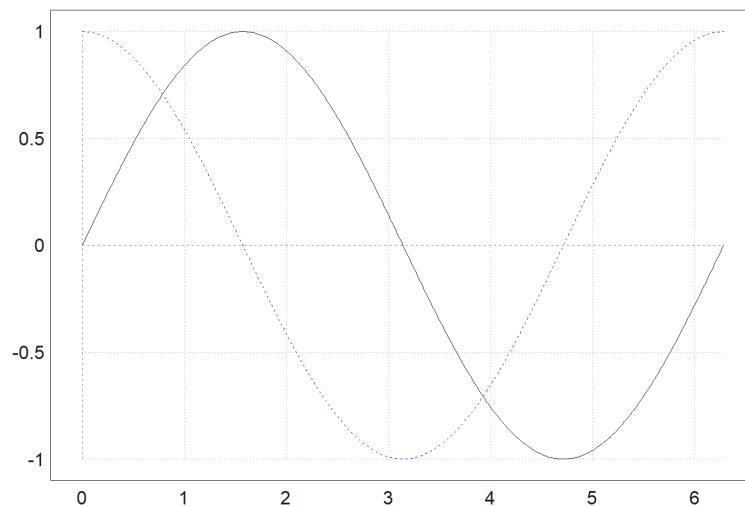
Menggambar Beberapa Kurva pada bidang koordinat yang sama

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

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

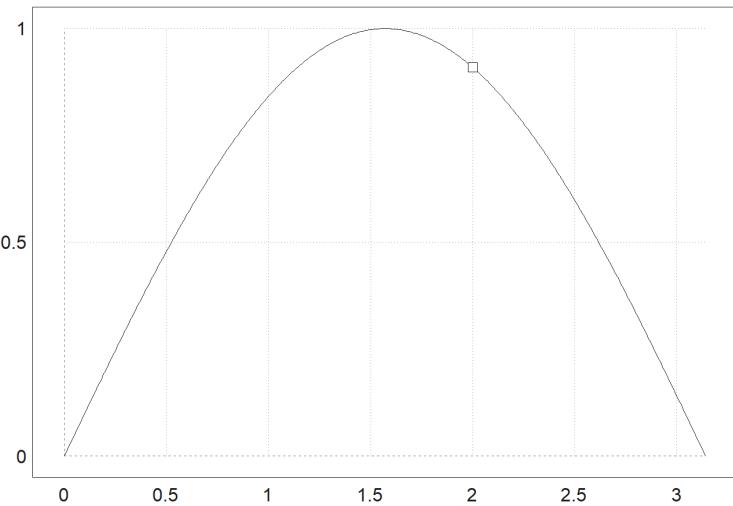


```
>aspect(1.5); plot2d("sin(x)",0,2pi); plot2d("cos(x)",color=blue,style="--",>add):
```



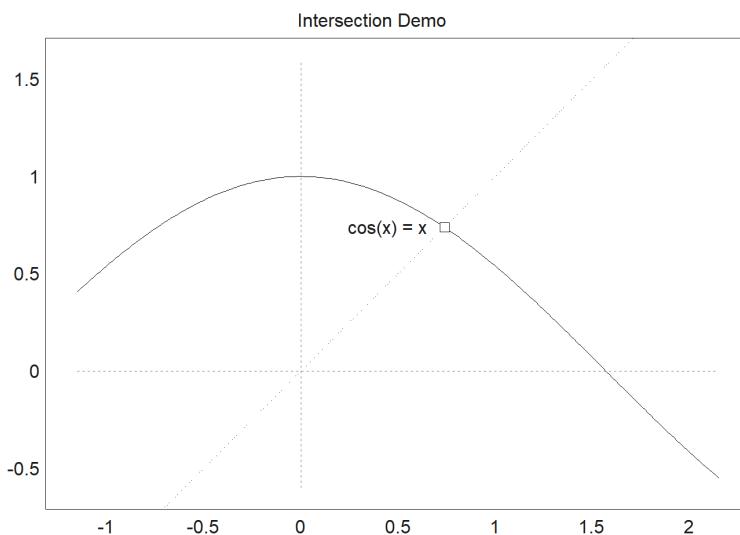
Salah satu kegunaan `>add` adalah untuk menambahkan titik pada kurva.

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



Kami menambahkan titik persimpangan dengan label (pada posisi "cl" untuk kiri tengah), dan memasukkan hasilnya ke dalam notebook. Kami juga menambahkan judul ke plot.

```
>plot2d(["cos(x)", "x"], r=1.1, cx=0.5, cy=0.5, ...
> color=[black,blue], style=["-", "."], ...
> grid=1);
>x0=solve("cos(x)-x",1); ...
> plot2d(x0,x0,>points,>add,title="Intersection Demo"); ...
> label("cos(x) = x",x0,x0, pos="cl", offset=20):
```



Dalam demo berikut, kami memplot fungsi $\text{sinc}(x)=\sin(x)/x$ dan ekspansi Taylor ke-8 dan ke-16. Kami menghitung ekspansi ini menggunakan Maxima melalui ekspresi simbolis.

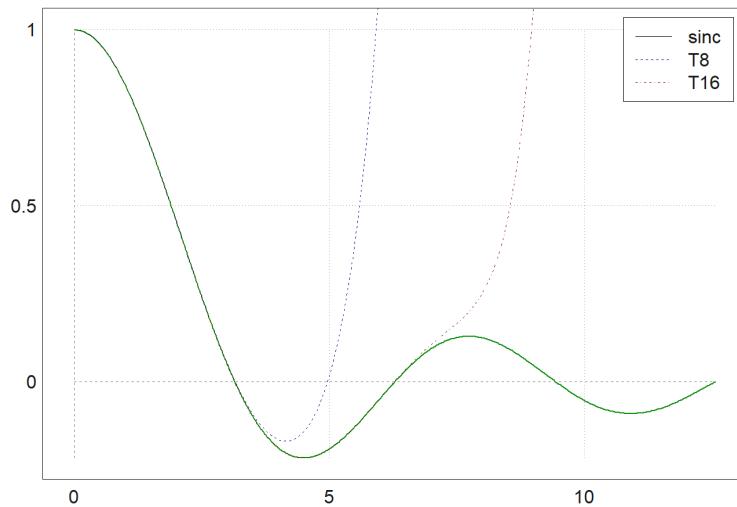
Plot ini dilakukan dalam perintah multi-baris berikut dengan tiga panggilan ke `plot2d()`. Yang kedua dan yang ketiga memiliki set flag `>add`, yang membuat plot menggunakan rentang sebelumnya.

Kami menambahkan kotak label yang menjelaskan fungsi.

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

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

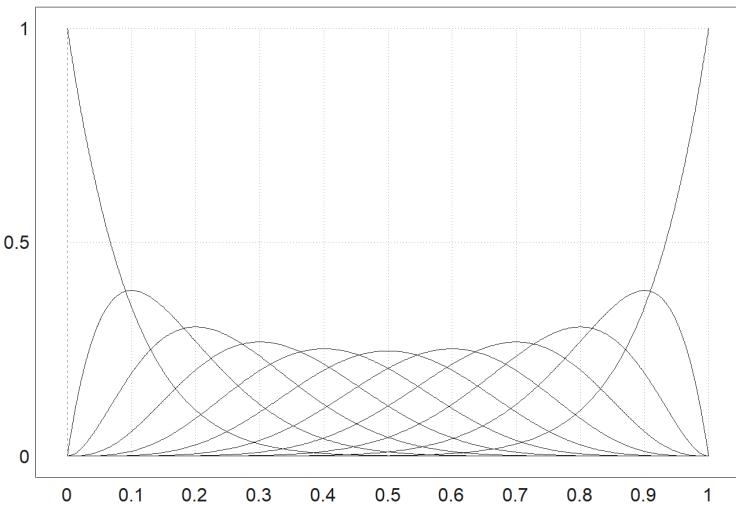
```
>plot2d("sinc(x)",0,4pi,color=green,thickness=2); ...
> plot2d(&taylor(sin(x)/x,x,0,8),>add,color=blue,style="--"); ...
> plot2d(&taylor(sin(x)/x,x,0,16),>add,color=red,style="-.-"); ...
> labelbox(["sinc","T8","T16"],styles=["-","--","-.-"], ...
> colors=[black,blue,red]):
```



Dalam contoh berikut, kami menghasilkan Bernstein-Polinomial.

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

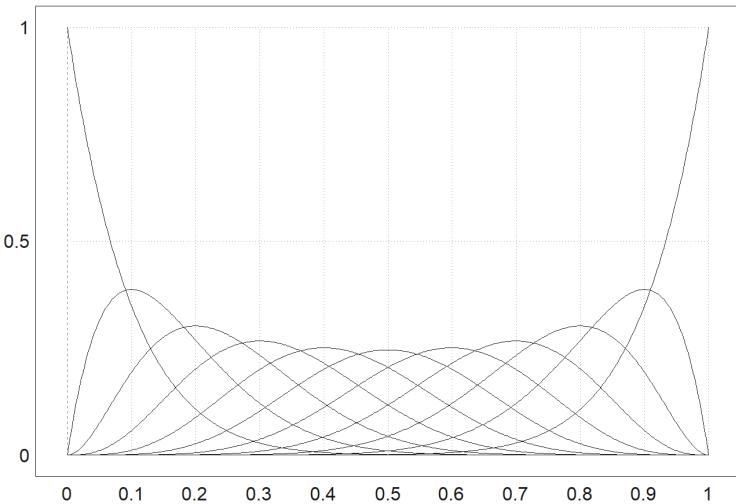
```
>plot2d("(1-x)^10",0,1); // plot first function
>for i=1 to 10; plot2d("bin(10,i)*x^i*(1-x)^(10-i)",>add); end;
>insimg;
```



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

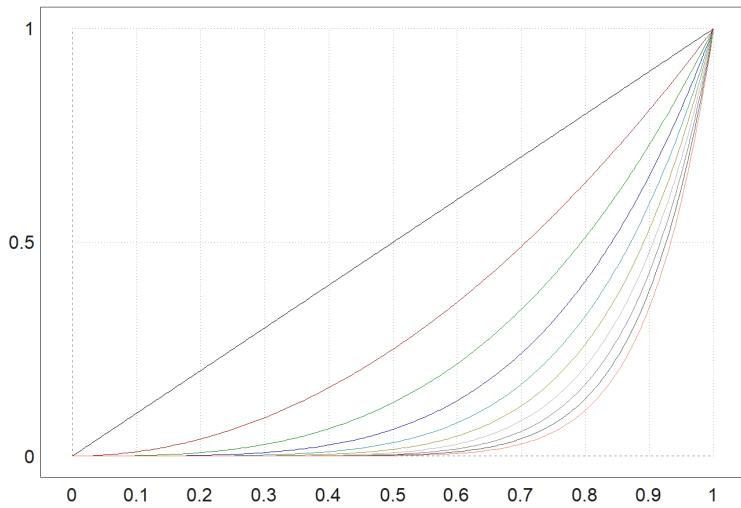
Kami menghasilkan matriks nilai dengan satu Polinomial Bernstein di setiap baris. Untuk ini, kita cukup menggunakan vektor kolom i. Lihat pengantar tentang bahasa matriks untuk mempelajari lebih detail.

```
>x=linspace(0,1,500);
>n=10; k=(0:n)'; // n 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):
```



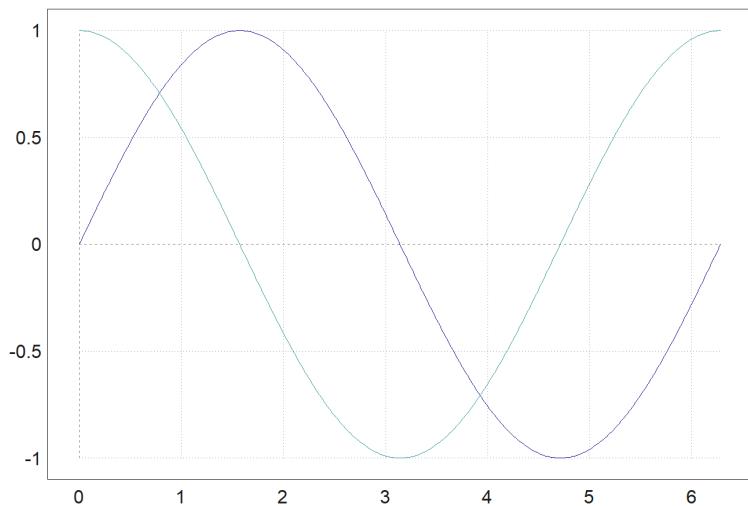
Perhatikan bahwa parameter warna dapat berupa vektor. Kemudian setiap warna digunakan untuk setiap baris matriks.

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

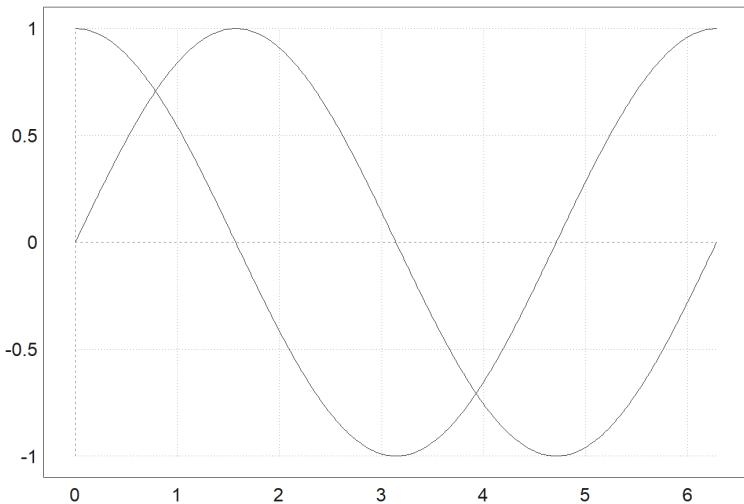


Metode lain adalah menggunakan vektor ekspresi (string). Anda kemudian dapat menggunakan larik warna, larik gaya, dan larik ketebalan dengan panjang yang sama.

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



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



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

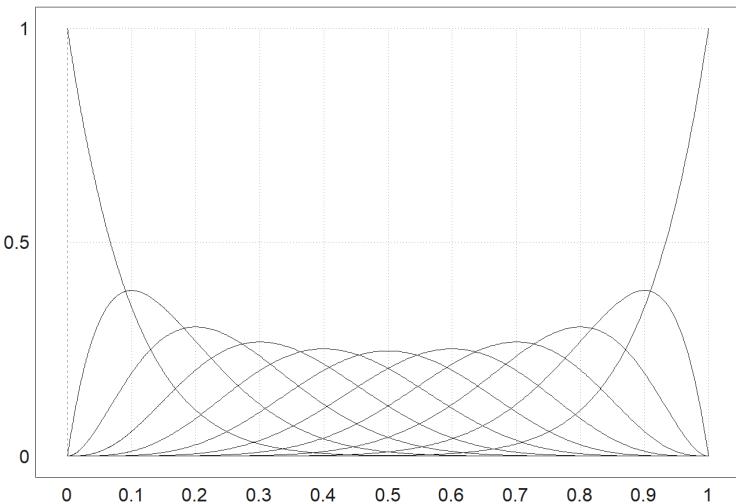
```
>v &= makelist(binomial(10,i)*x^i*(1-x)^(10-i),i,0,10) // make list
```

$$\begin{aligned} &[(1-x)^{10}, 10(1-x)^9x, 45(1-x)^8x^2, 120(1-x)^7x^3, \\ &\quad 210(1-x)^6x^4, 252(1-x)^5x^5, 210(1-x)^4x^6, 120(1-x)^3x^7, \\ &\quad 45(1-x)^2x^8, 10(1-x)x^9, x^{10}] \end{aligned}$$

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

$$\begin{aligned} &(1-x)^{10} \\ &10*(1-x)^9*x \\ &45*(1-x)^8*x^2 \\ &120*(1-x)^7*x^3 \\ &210*(1-x)^6*x^4 \\ &252*(1-x)^5*x^5 \\ &210*(1-x)^4*x^6 \\ &120*(1-x)^3*x^7 \\ &45*(1-x)^2*x^8 \\ &10*(1-x)*x^9 \\ &x^{10} \end{aligned}$$

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

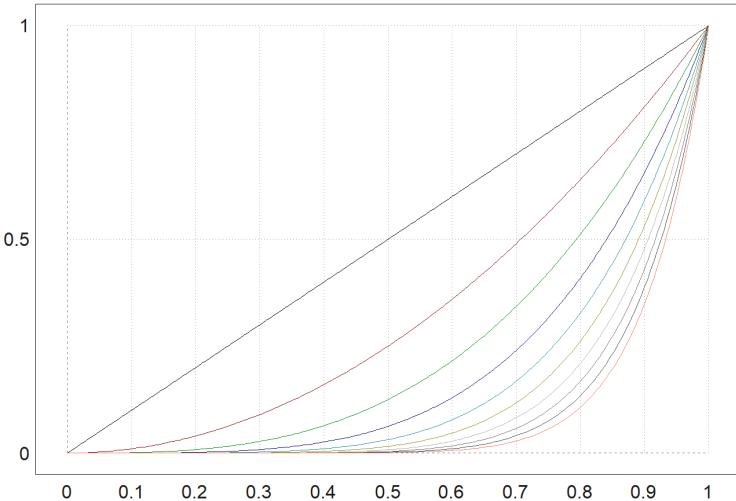


Alternatif lain adalah dengan menggunakan bahasa matriks Euler.

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

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

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

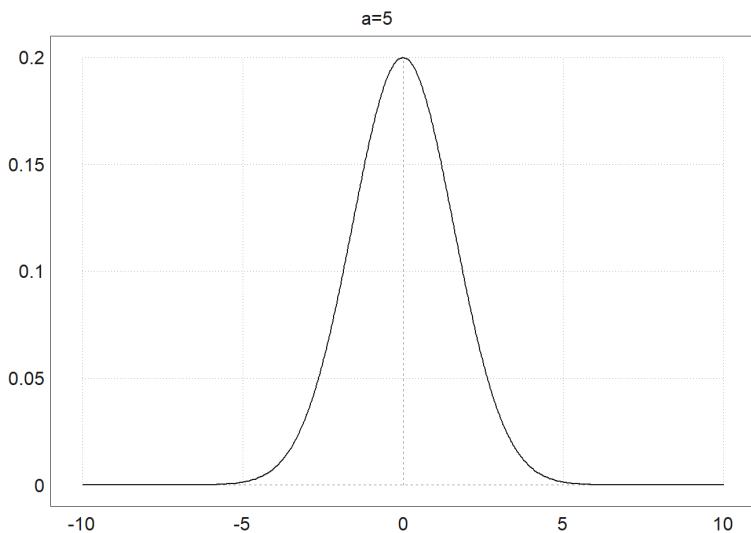


Ekspresi dan fungsi satu baris dapat melihat variabel global.

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

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

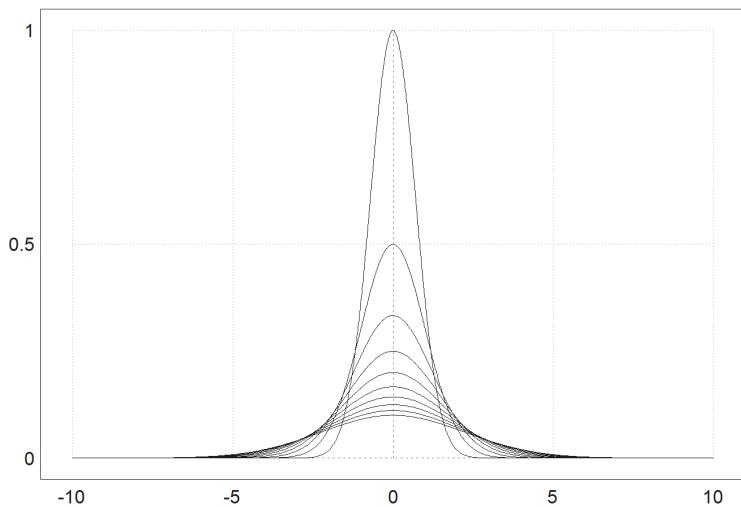
```
>function f(x,a) := 1/a*exp(-x^2/a); ...
>plot2d("f",-10,10;5,thickness=2,title="a=5");
```



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

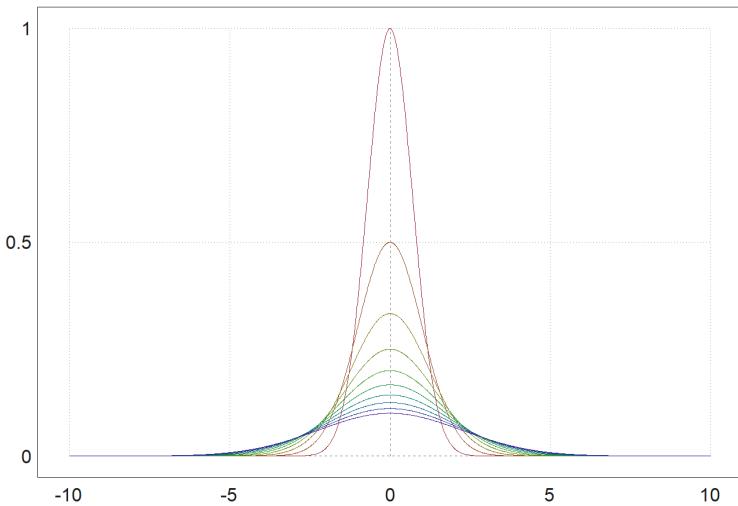
Dalam contoh berikut, kami menggunakan loop untuk memplot beberapa fungsi (lihat tutorial tentang pemrograman untuk loop).

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



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

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



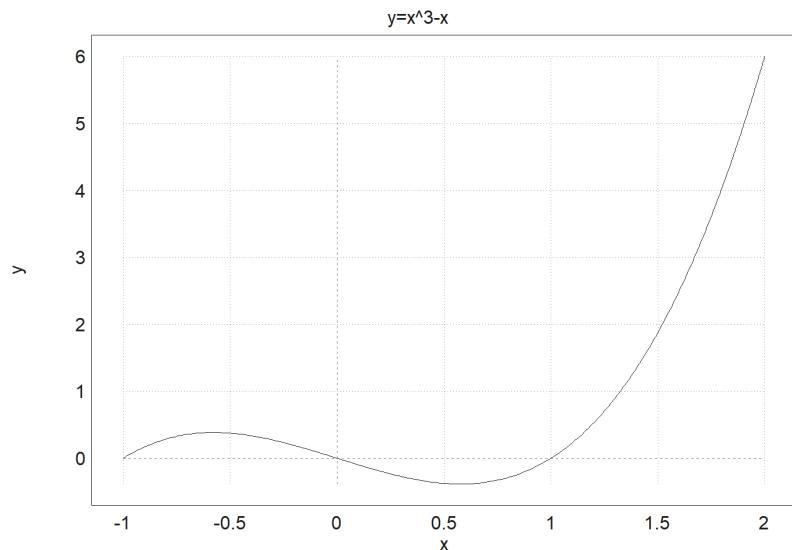
Label Teks

Dekorasi sederhana bisa

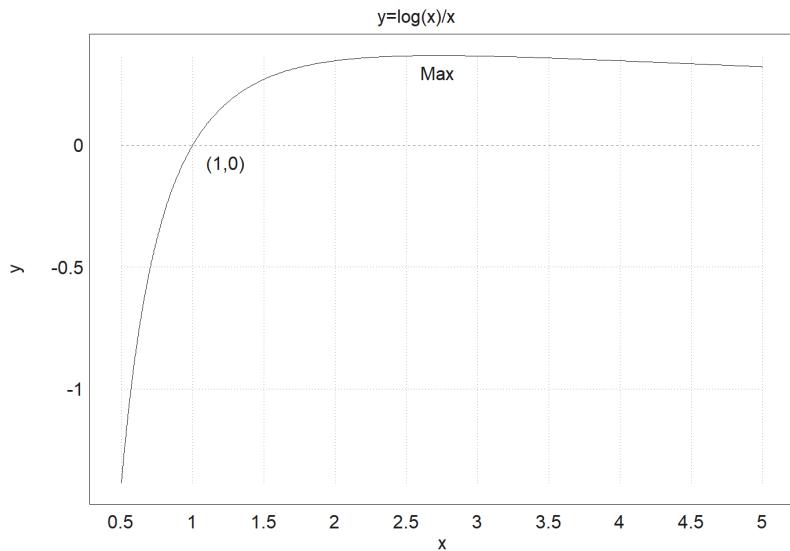
- judul dengan `judul="..."`
- x- dan y-label dengan `xl="...", yl="..."`
- label teks lain dengan `label("...",x,y)`

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

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

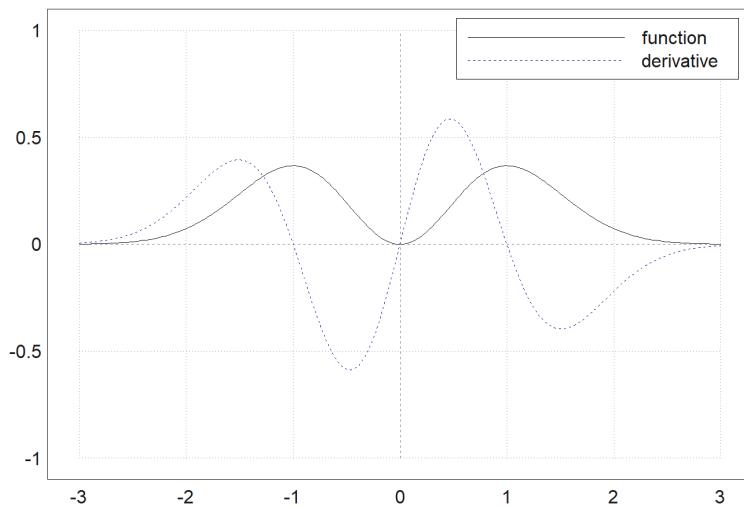


```
>expr := "log(x)/x"; ...
> plot2d(expr,0.5,5,title="y="+expr,xl="x",yl="y"); ...
> label("(1,0)",1,0); label("Max",E,expr(E),pos="lc") :
```



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

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

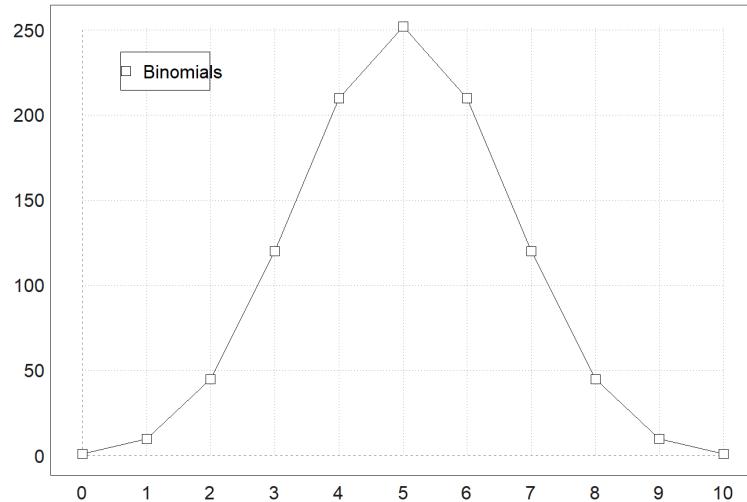


Kotak ditambatkan di kanan atas secara default, tetapi > kiri menambatkannya di kiri atas. Anda dapat memindahkannya ke tempat yang Anda suka. Posisi jangkar adalah sudut kanan atas kotak, dan angkanya adalah pecahan dari ukuran jendela grafik. Lebarnya otomatis.

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

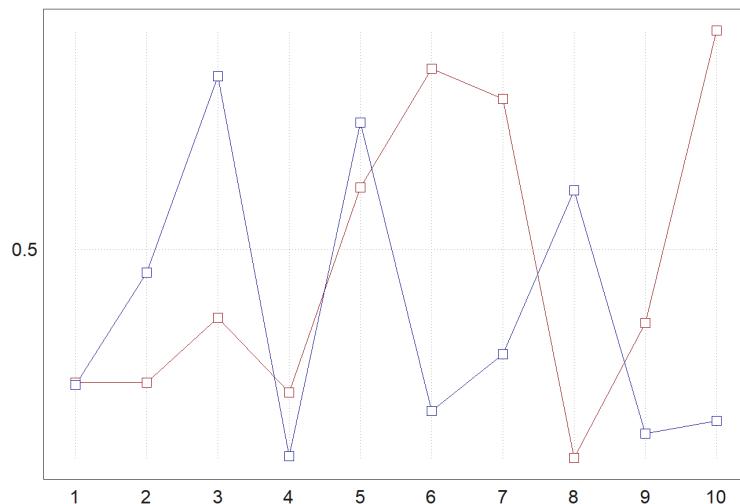
Dalam contoh berikut, hanya ada satu fungsi. Jadi kita bisa menggunakan string sebagai pengganti vektor string. Kami mengatur warna teks menjadi hitam untuk contoh ini.

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



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

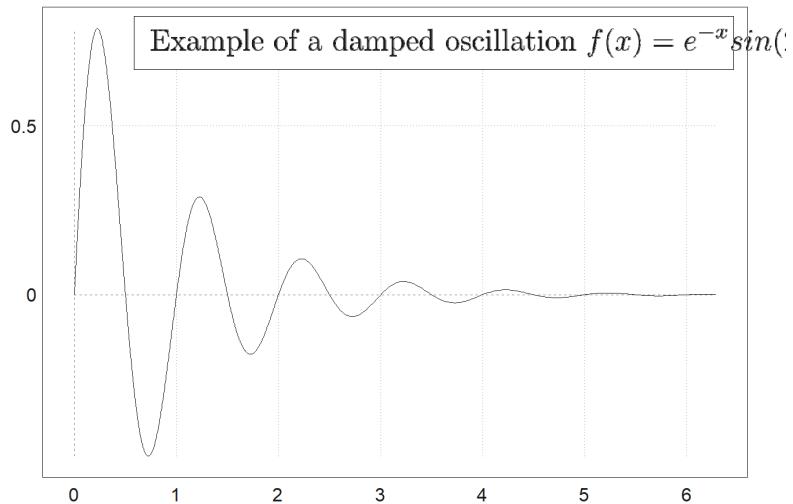
```
>statplot(1:10,random(2,10),color=[red,blue]):
```



Fitur serupa adalah fungsi textbox().

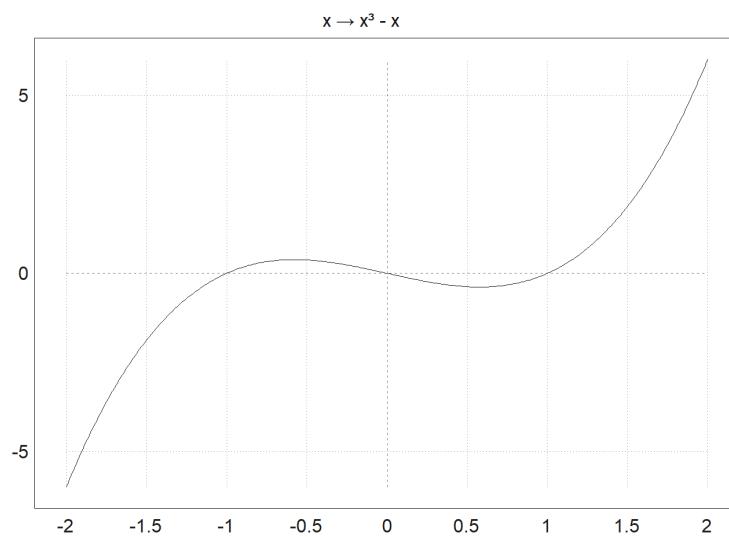
Lebar secara default adalah lebar maksimal dari baris teks. Tapi itu bisa diatur oleh pengguna juga.

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



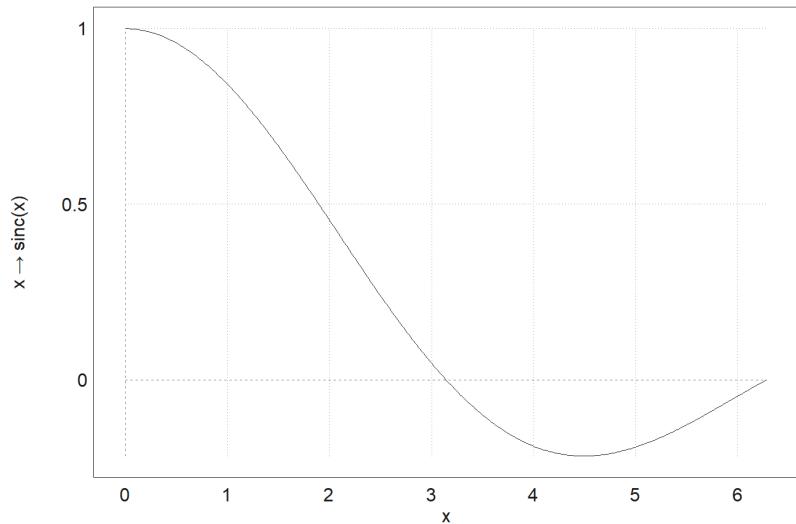
Label teks, judul, kotak label, dan teks lainnya dapat berisi string Unicode (lihat sintaks EMT untuk mengetahui lebih lanjut tentang string Unicode).

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



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

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



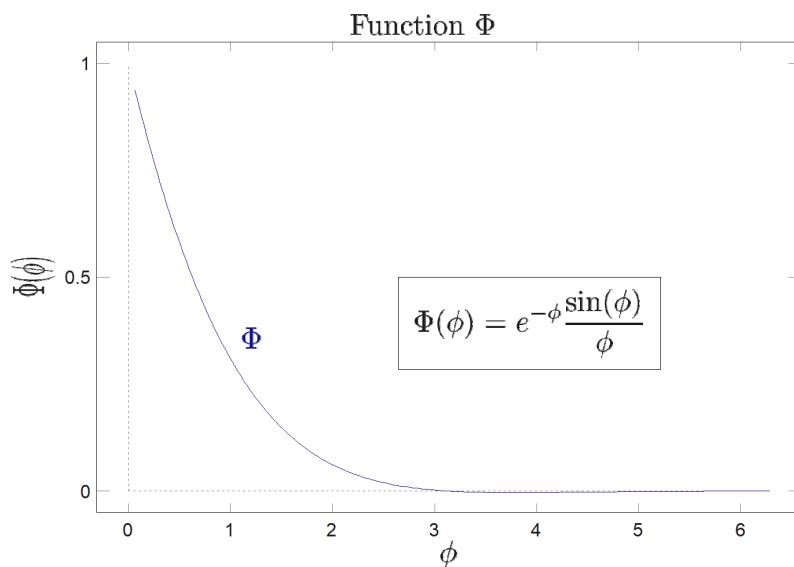
LaTeX

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

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

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

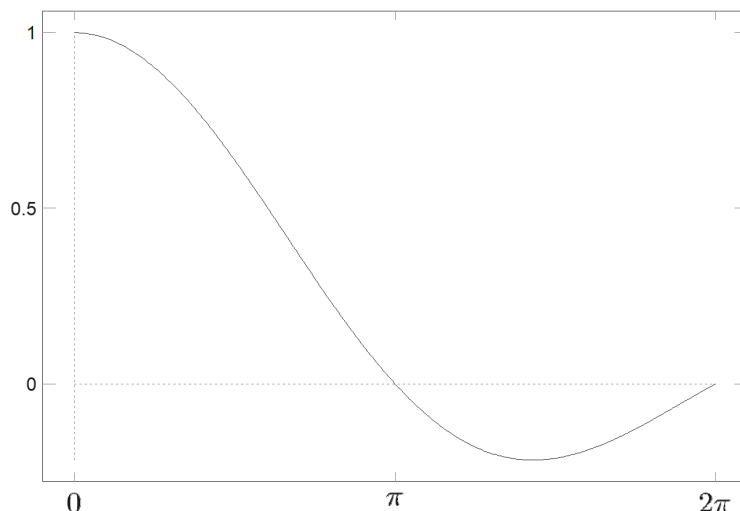
```
>plot2d("exp(-x)*sin(x)/x",a=0,b=2pi,c=0,d=1,grid=6,color=blue, ...
> title=latex("\text{Function } \Phi"), ...
> xl=latex("\phi"),yl=latex("\Phi(\phi)"); ...
>textbox( ...
> latex("\Phi(\phi) = e^{-\phi} \frac{\sin(\phi)}{\phi}"),x=0.8,y=0.5); ...
>label(latex("\Phi",color=blue),1,0.4):
```



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

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

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



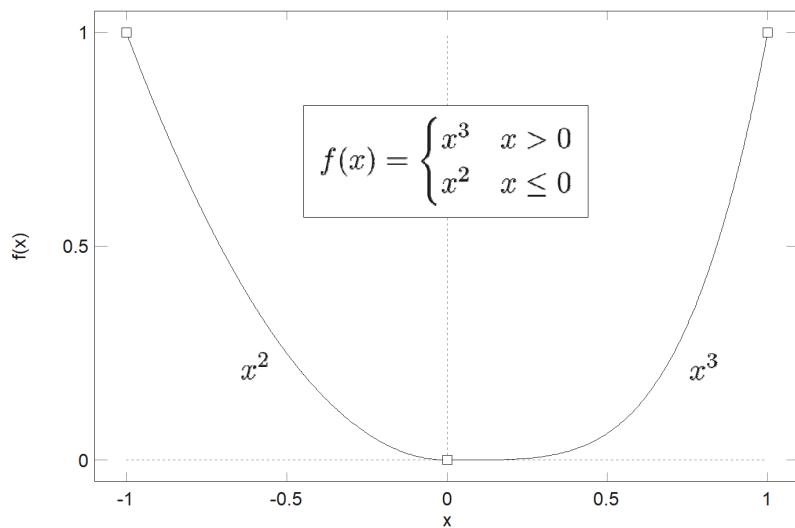
Tentu saja, fungsi juga dapat digunakan.

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

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

Pada plot berikut, kami juga memasukkan beberapa kode LaTeX. Kami menggunakanannya untuk dua label dan kotak teks. Tentu saja, Anda hanya akan dapat menggunakan LaTeX jika Anda telah menginstal LaTeX dengan benar.

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



Interaksi pengguna

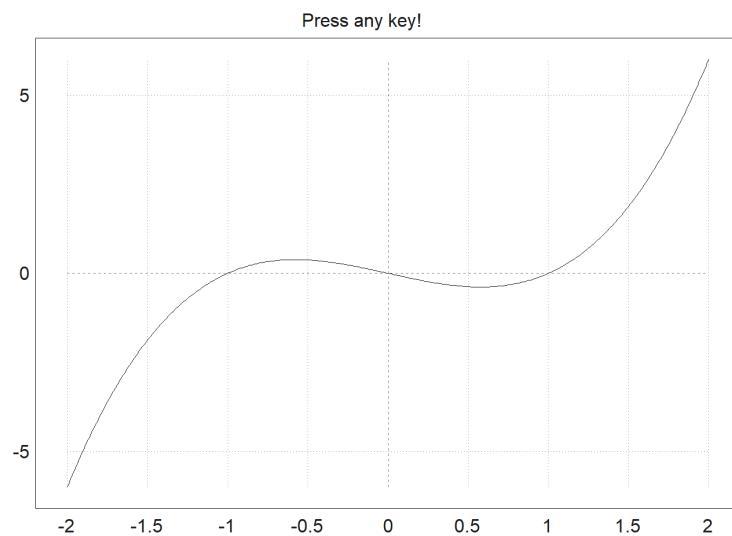
Saat memplot fungsi atau ekspresi, parameter `>user` memungkinkan pengguna untuk memperbesar dan menggeser plot dengan tombol kursor atau mouse. Pengguna dapat

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

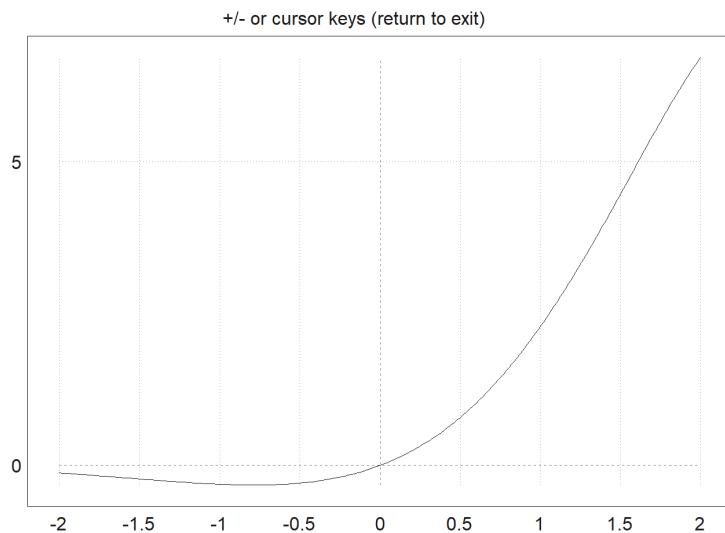
Tombol spasi akan mengatur ulang plot ke jendela plot asli.

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

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



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



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

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

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

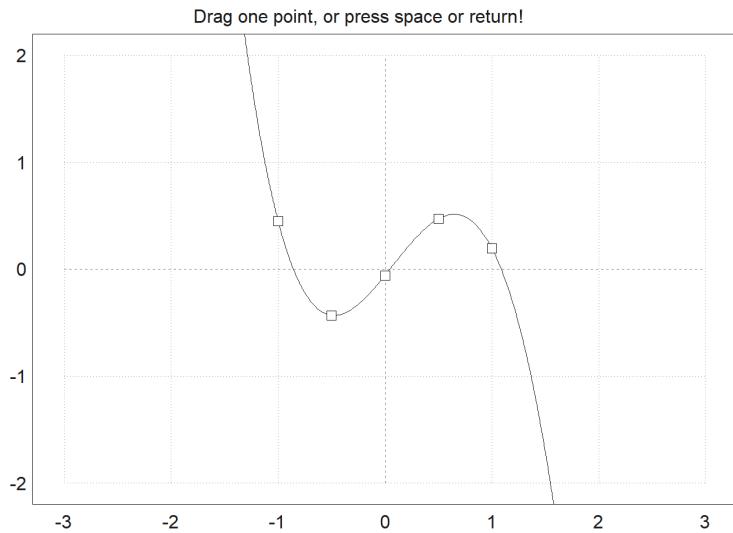
```
>function plotf(xp,yp,select) ...
```

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

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

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

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



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

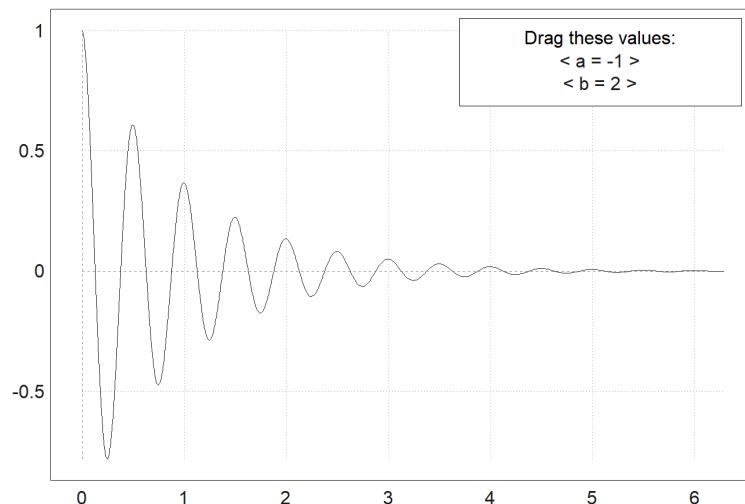
Pertama kita membutuhkan fungsi plot.

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

Kemudian kita membutuhkan nama untuk parameter, nilai awal dan matriks rentang nx2, opsional baris judul.

Ada slider interaktif, yang dapat mengatur nilai oleh pengguna. Fungsi dragvalues() menyediakan ini.

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

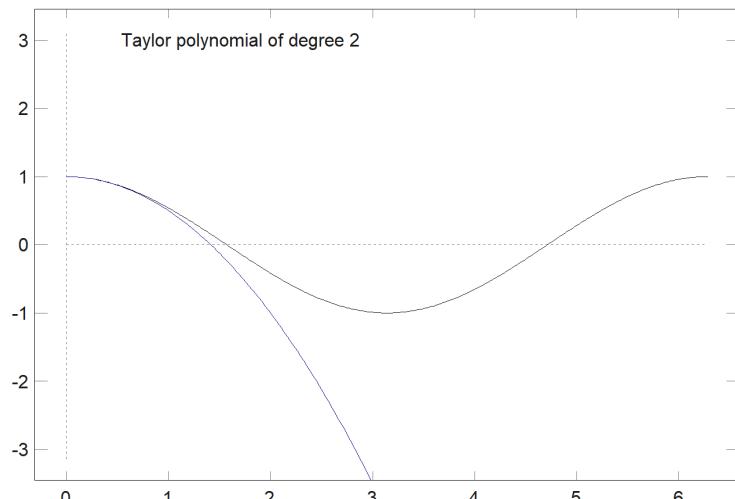


Dimungkinkan untuk membatasi nilai yang diseret ke bilangan bulat. Sebagai contoh, kita menulis fungsi plot, yang memplot polinomial Taylor derajat n ke fungsi kosinus.

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

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

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



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

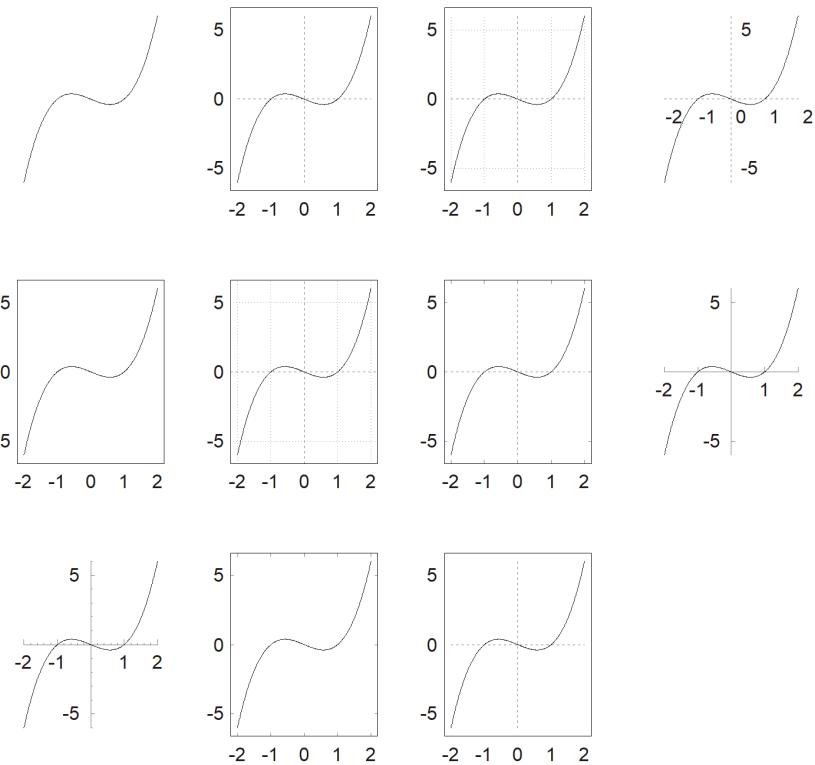
```
>function dragtest ...
    plot2d(none,r=1,title="Drag with the mouse, or press any key!");
    start=0;
repeat
    {flag,m,time}=mousedrag();
    if flag==0 then return; endif;
    if flag==2 then
        hold on; mark(m[1],m[2]); hold off;
    endif;
end
endfunction
```

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

Gaya Plot 2D

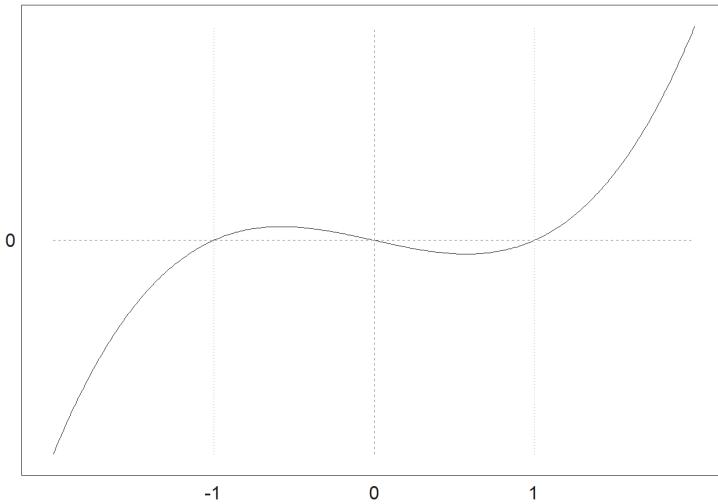
Secara default, EMT menghitung tick sumbu otomatis dan menambahkan label ke setiap tick. Ini dapat diubah dengan parameter grid. Gaya default sumbu dan label dapat dimodifikasi. Selain itu, label dan judul dapat ditambahkan secara manual. Untuk mengatur ulang ke gaya default, gunakan reset().

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



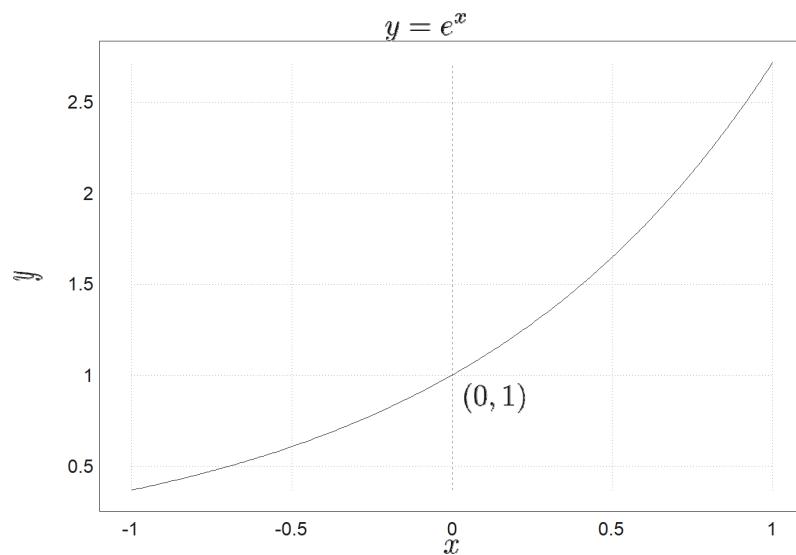
Parameter `<frame` mematikan frame, dan `framecolor=blue` mengatur frame ke warna biru. Jika Anda ingin centang sendiri, Anda dapat menggunakan `style=0`, dan menambahkan semuanya nanti.

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



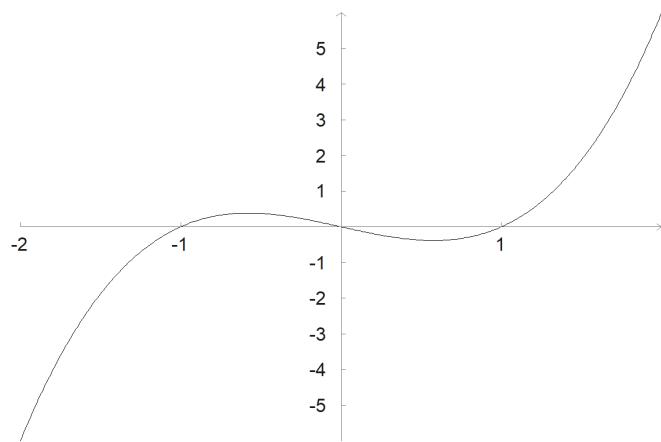
Untuk judul plot dan label sumbu, lihat contoh berikut.

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



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

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

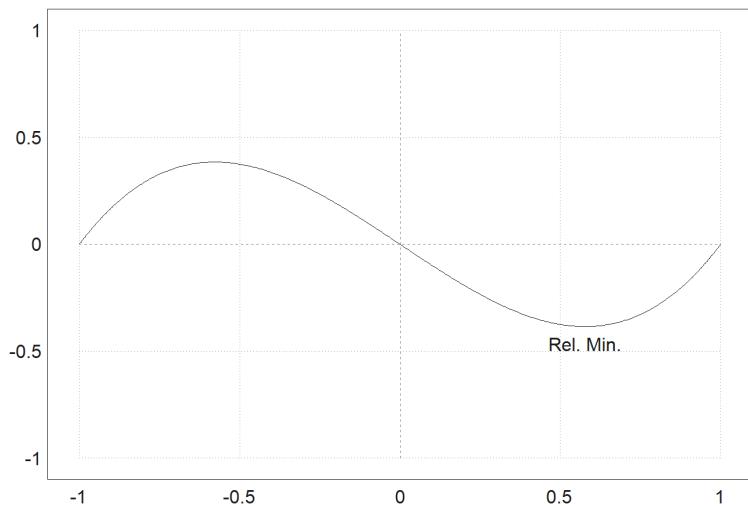


Teks pada plot dapat diatur dengan label(). Dalam contoh berikut, "lc" berarti tengah bawah. Ini mengatur posisi label relatif terhadap koordinat plot.

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

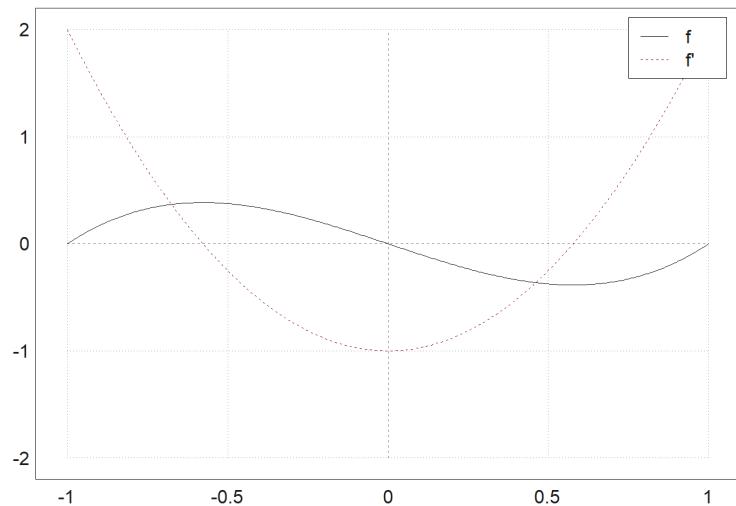
$$x^3 - x$$

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

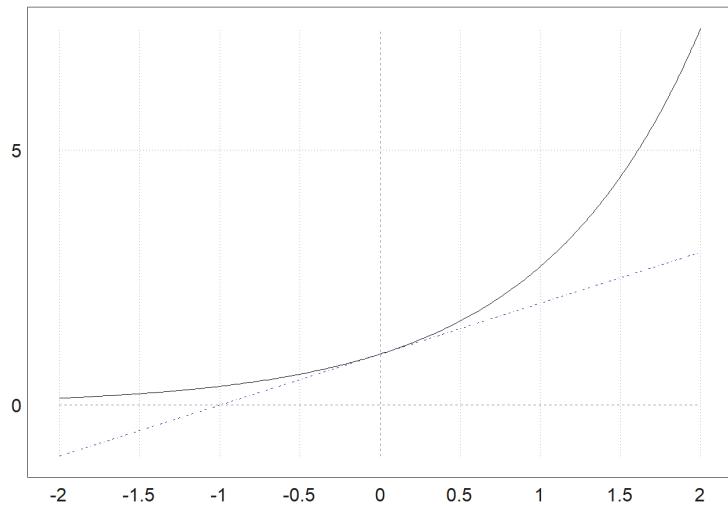


Ada juga kotak teks.

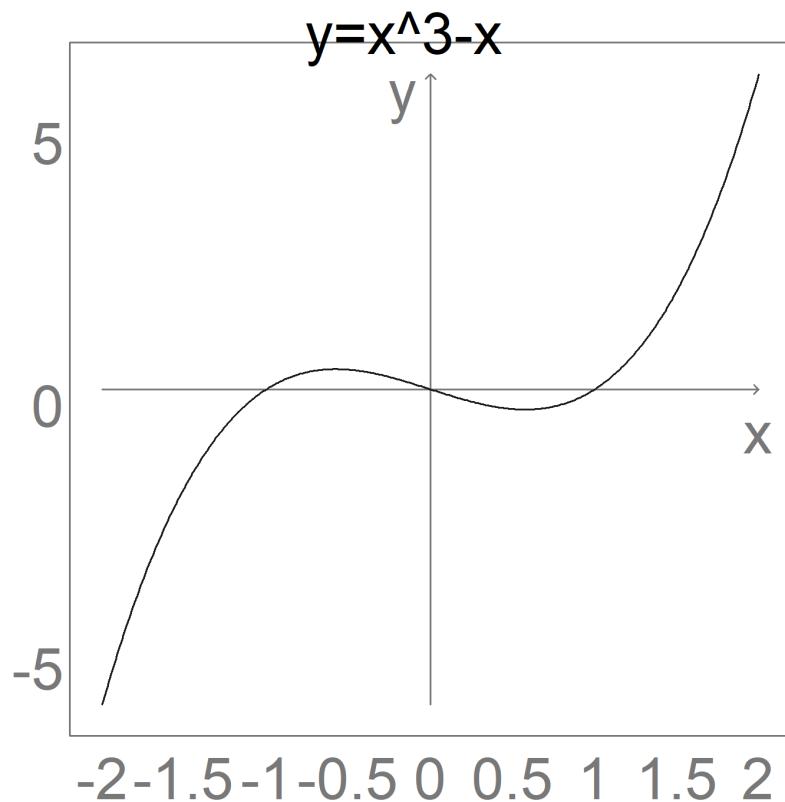
```
>plot2d(&f(x),-1,1,-2,2); // function
>plot2d(&diff(f(x),x),>add,style="--",color=red); // derivative
>labelbox(["f","f'"],["-", "--"],[black,red]): // label box
```



```
>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():
```



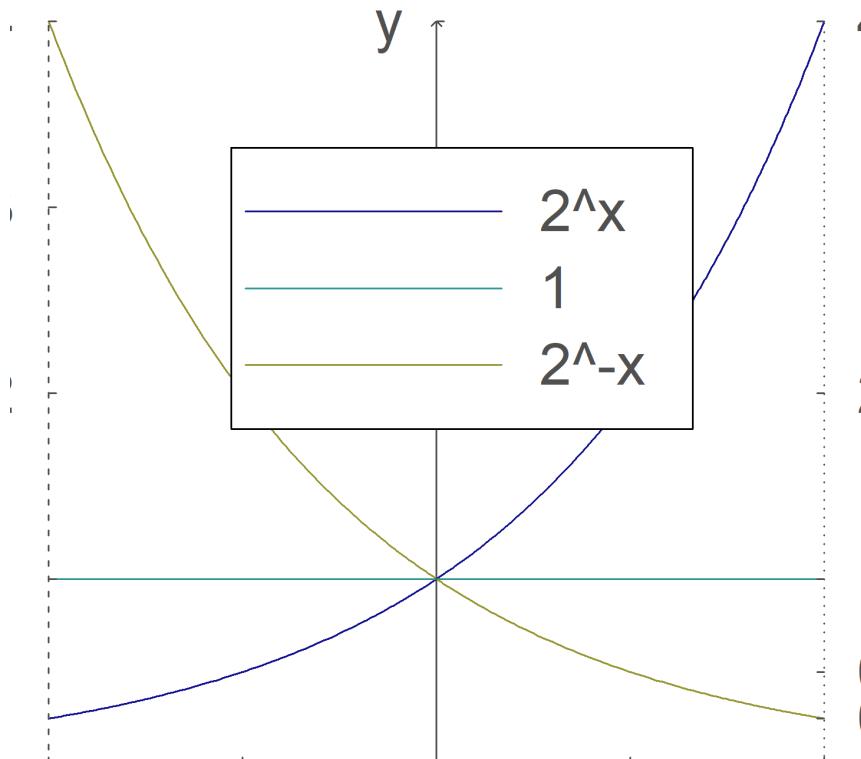
Untuk kontrol lebih, sumbu x dan sumbu y dapat dilakukan secara manual.

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

```

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

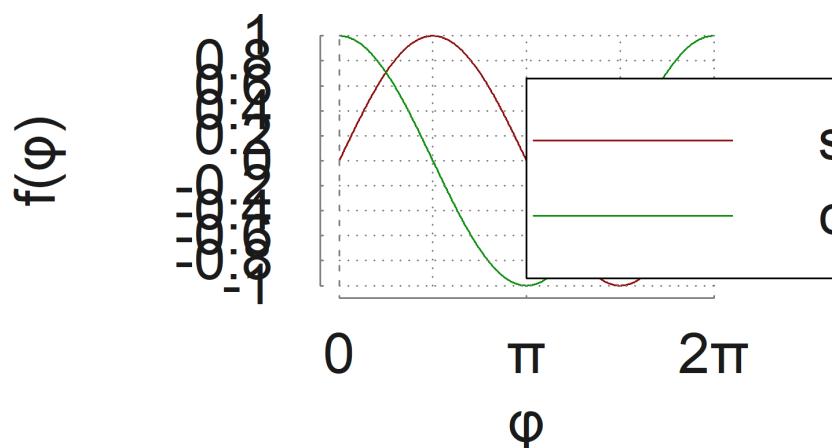
```



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

```

>aspect(1.5);
>plot2d(["sin(x)","cos(x)"],0,2pi,color=[red,green],<grid,<frame); ...
> xaxis(-1.1,(0:2)*pi,xt=["0",u"\u03c0;","u"2\u03c0;"],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"\u03c6"); ylabel(u"f(\u03c6)"):
```



Merencanakan Data 2D

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

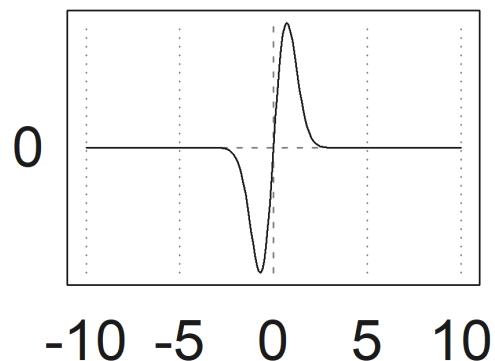
Memplot ekspresi hanyalah singkatan untuk plot data. Untuk plot data, Anda memerlukan satu atau beberapa baris nilai x , dan satu atau beberapa baris nilai y . Dari rentang dan nilai- x , fungsi plot2d akan menghitung data yang akan diplot, secara default dengan evaluasi fungsi yang adaptif. Untuk plot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

Tapi Anda bisa memasukkan data secara langsung.

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

Berikut adalah contoh dengan satu baris untuk x dan y .

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



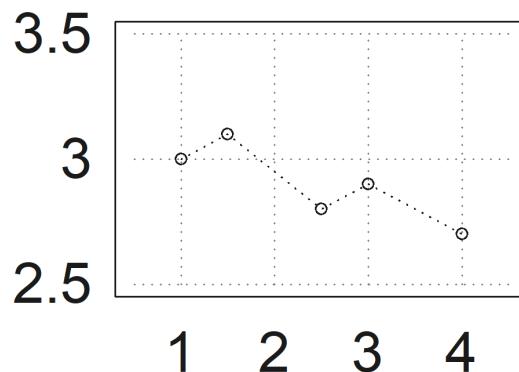
Data juga dapat diplot sebagai titik. Gunakan poin=true untuk ini. Plotnya bekerja seperti poligon, tetapi hanya menggambar sudut-sudutnya.

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

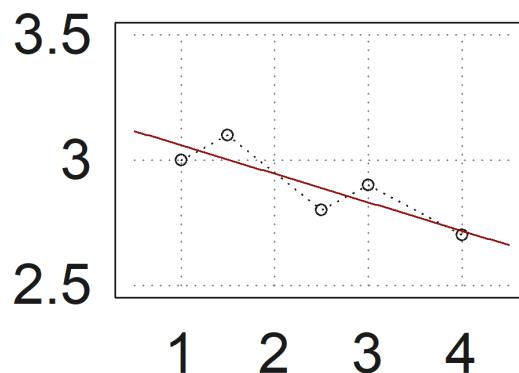
Untuk memplot set poin gunakan >points. Jika warna adalah vektor warna, setiap titik mendapat warna yang berbeda. Untuk matriks koordinat dan vektor kolom, warna berlaku untuk baris matriks.

Parameter >addpoints menambahkan titik ke segmen garis untuk plot data.

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



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



Menggambar Daerah Yang Dibatasi Kurva

Plot data benar-benar poligon. Kita juga dapat memplot kurva atau kurva terisi.

- `terisi=benar` mengisi plot.
- `style="...":` Pilih dari "", "/", "\", "\\"/".
- `fillcolor:` Lihat di atas untuk warna yang tersedia.

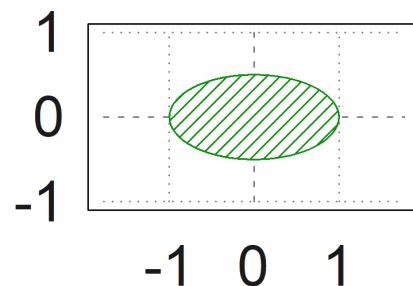
Warna isian ditentukan oleh argumen "fillcolor", dan pada <outline opsional mencegah menggambar batas untuk semua gaya kecuali yang default.

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

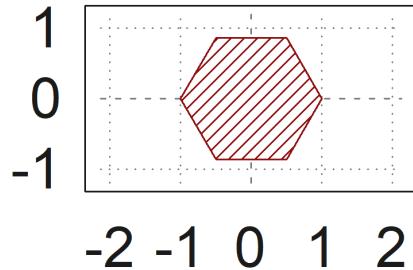


Dalam contoh berikut kami memplot elips terisi dan dua segi enam terisi menggunakan kurva tertutup dengan 6 titik dengan gaya isian berbeda.

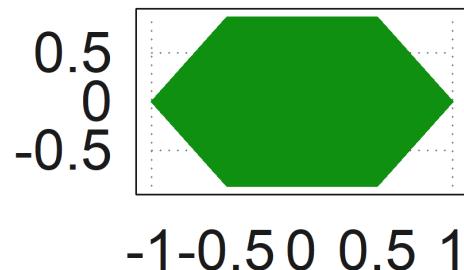
```
>x=linspace(0,2pi,1000); plot2d(sin(x),cos(x)*0.5,r=1,>filled,style="/"):
```



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

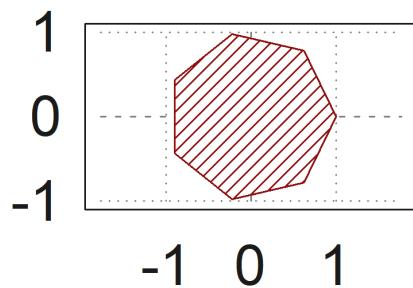


```
>t=linspace(0,2pi,6); plot2d(cos(t),sin(t),>filled,style="#" ):
```



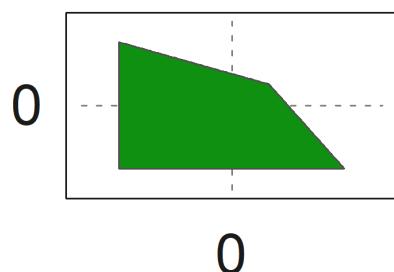
Contoh lainnya adalah segi empat, yang kita buat dengan 7 titik pada lingkaran satuan.

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



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

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

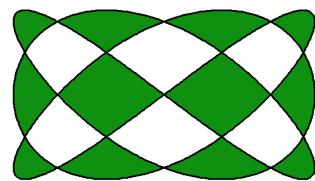


Poin utama dari bahasa matriks adalah memungkinkan untuk menghasilkan tabel fungsi dengan mudah.

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

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

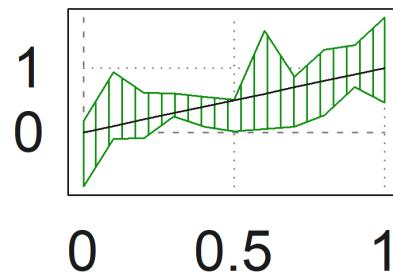
```
>plot2d(x,y,<grid,<frame,>filled):
```



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

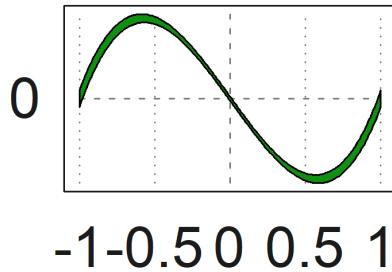
Hal ini dapat berguna untuk memplot kesalahan perhitungan. Tapi itu bisa juga digunakan untuk memplot kesalahan statistik.

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



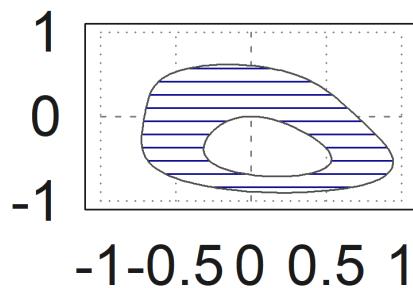
Jika x adalah vektor yang diurutkan, dan y adalah vektor interval, maka plot2d akan memplot rentang interval yang terisi dalam bidang. Gaya isian sama dengan gaya poligon.

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



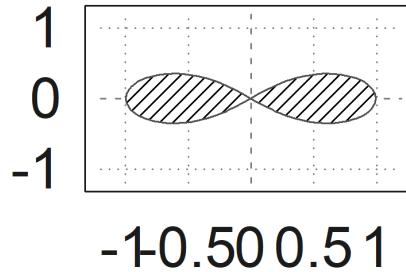
Jika x adalah vektor yang diurutkan, dan y adalah vektor interval, maka `plot2d` akan memplot rentang interval yang terisi dalam bidang. Gaya isian sama dengan gaya poligon.

```
>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
```

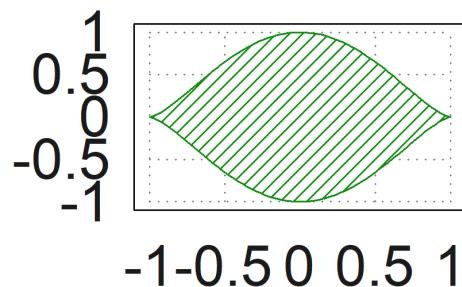


Kami juga dapat mengisi rentang nilai seperti
lateks: $-1 \leq (x^2+y^2)^2-x^2+y^2 \leq 0$.

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



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



Grafik Fungsi Parametrik

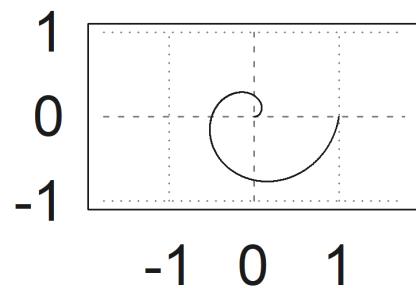
Nilai-x tidak perlu diurutkan. (x,y) hanya menggambarkan kurva. Jika x diurutkan, kurva tersebut merupakan grafik fungsi.

Dalam contoh berikut, kami memplot spiral

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

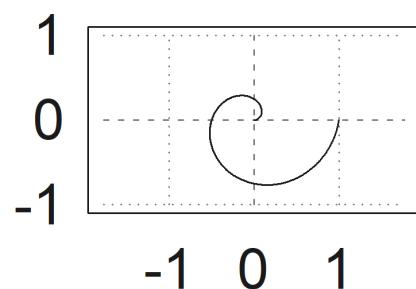
Kita perlu menggunakan banyak titik untuk tampilan yang halus atau fungsi adaptif() untuk mengevaluasi ekspresi (lihat fungsi adaptif() untuk lebih jelasnya).

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

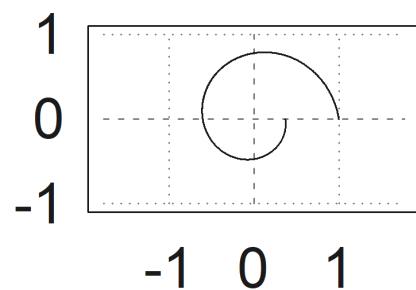


Atau, dimungkinkan untuk menggunakan dua ekspresi untuk kurva. Berikut ini plot kurva yang sama seperti di atas.

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

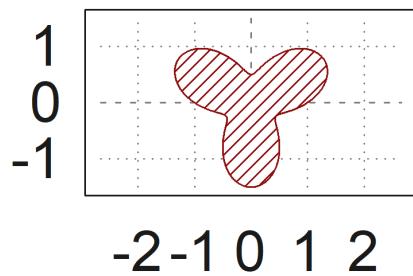


```
>t=linspace(0,1,1000); r=exp(-t); x=r*cos(2pi*t); y=r*sin(2pi*t);
>plot2d(x,y,r=1):
```



Dalam contoh berikutnya, kami memplot kurva
lateks: $\gamma(t) = (r(t) \cos(t), r(t) \sin(t))$
dengan
lateks: $r(t) = 1 + \frac{\sin(3t)}{2}$.

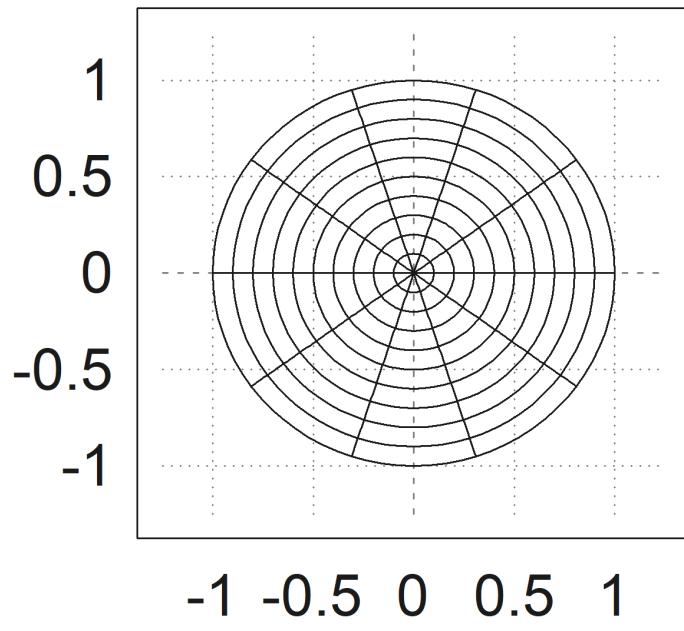
```
>t=linspace(0,2pi,1000); r=1+sin(3*t)/2; x=r*cos(t); y=r*sin(t); ...
>plot2d(x,y,>filled,fillcolor=red,style="/" ,r=1.5):
```



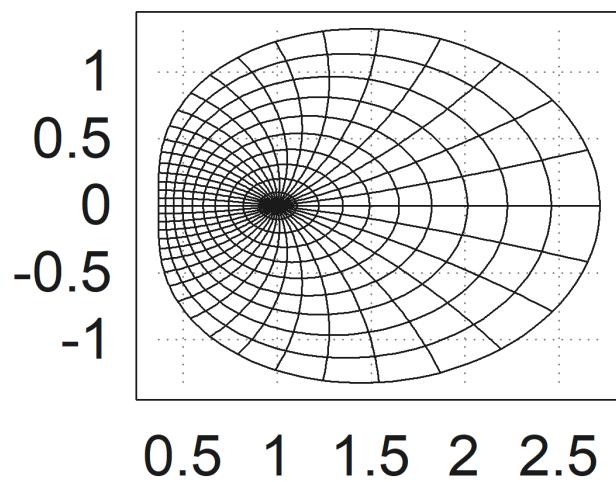
Menggambar Grafik Bilangan Kompleks

Array bilangan kompleks juga dapat diplot. Kemudian titik-titik grid akan terhubung. Jika sejumlah garis kisi ditentukan (atau vektor garis kisi 1×2) dalam argumen cgrid, hanya garis kisi tersebut yang terlihat.
Matriks bilangan kompleks akan secara otomatis diplot sebagai kisi di bidang kompleks.
Dalam contoh berikut, kami memplot gambar lingkaran satuan di bawah fungsi eksponensial. Parameter cgrid menyembunyikan beberapa kurva grid.

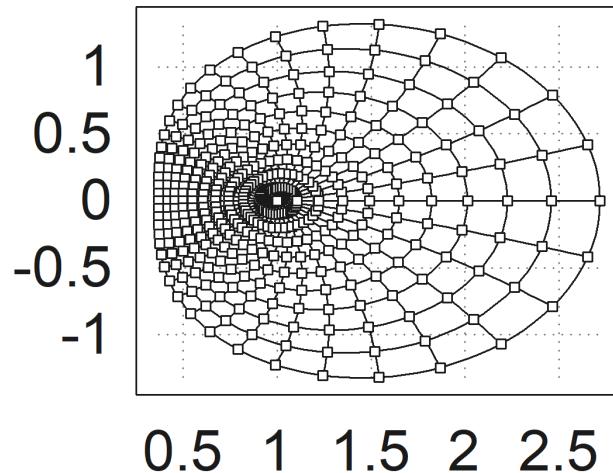
```
>aspect(); r=linspace(0,1,50); a=linspace(0,2pi,80)'; z=r*exp(I*a);...
>plot2d(z,a=-1.25,b=1.25,c=-1.25,d=1.25,cgrid=10):
```



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



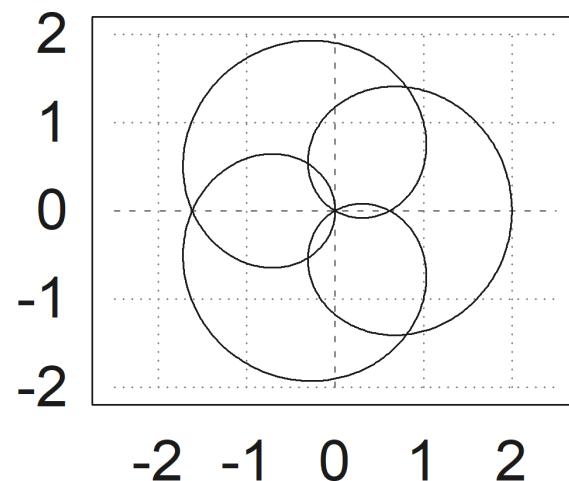
```
>r=linspace(0,1,10); a=linspace(0,2pi,40)'; z=r*exp(I*a);
>plot2d(exp(z),>points,>add):
```



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

Dalam contoh, kami memplot lingkaran satuan dengan
lateks: $\gamma(t) = e^{it}$

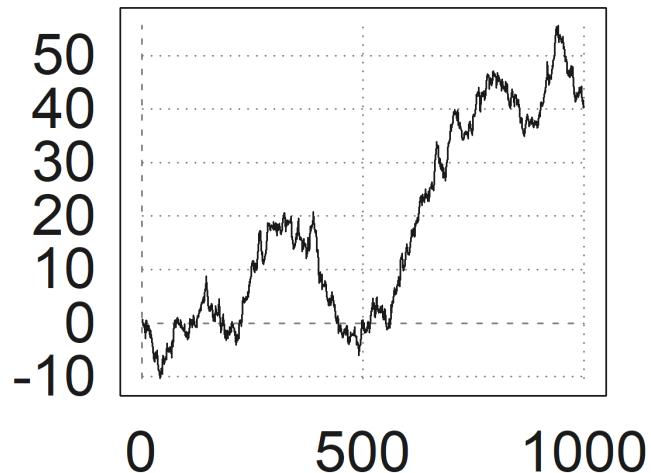
```
>t=linspace(0,2pi,1000); ...
>plot2d(exp(I*t)+exp(4*I*t),r=2):
```



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

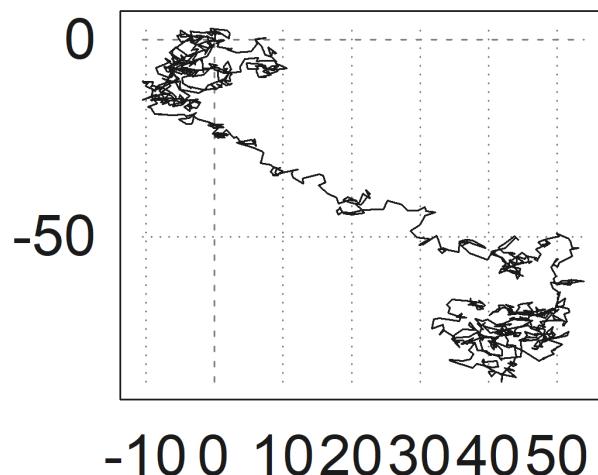
Jumlah kumulatif dari nilai terdistribusi 0-1-normal menghasilkan jalan acak.

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

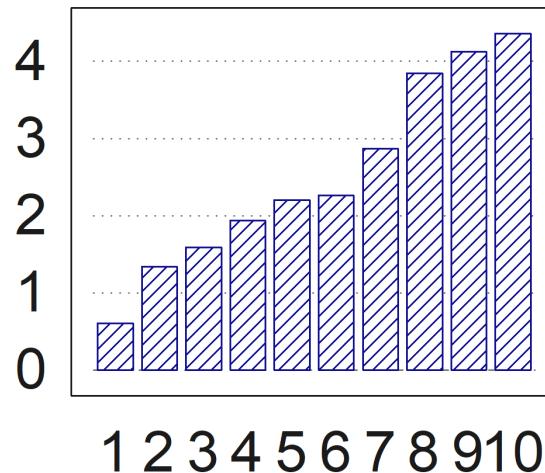


Menggunakan dua baris menunjukkan jalan dalam dua dimensi.

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

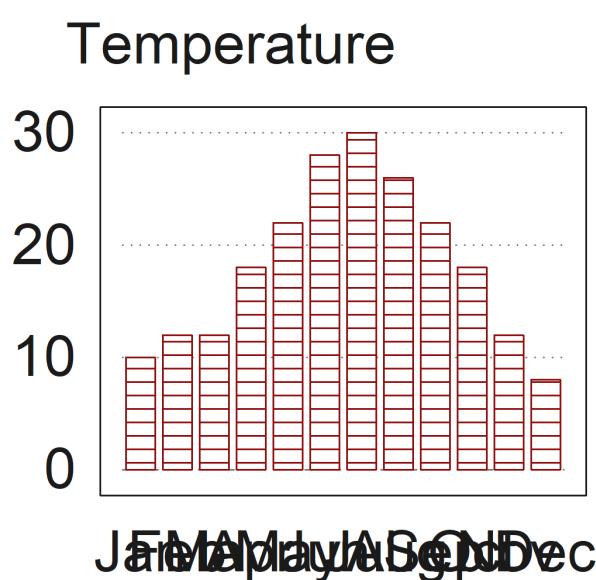


```
>columnsplot(cumsum(random(10)), style="/", color=blue) :
```

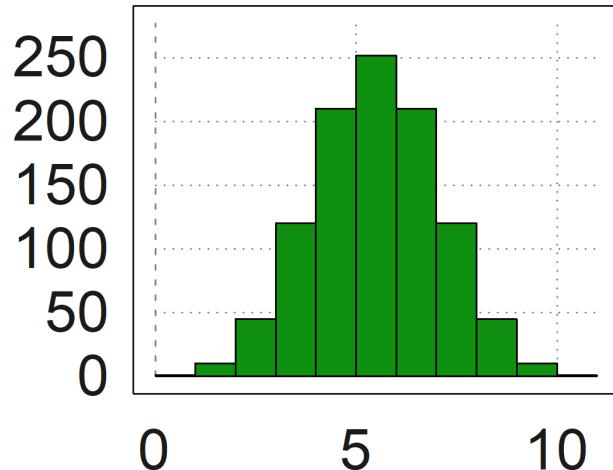


Itu juga dapat menampilkan string sebagai label.

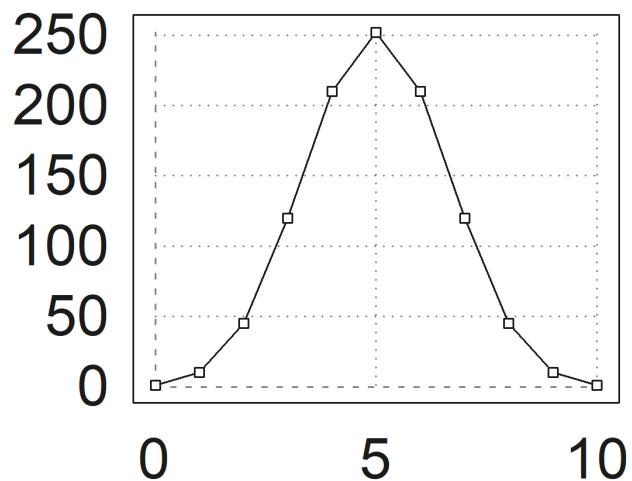
```
>months=["Jan", "Feb", "Mar", "Apr", "May", "Jun", ...
>    "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"];
>values=[10,12,12,18,22,28,30,26,22,18,12,8];
>columnsplot(values, lab=months, color=red, style="-");
>title("Temperature") :
```



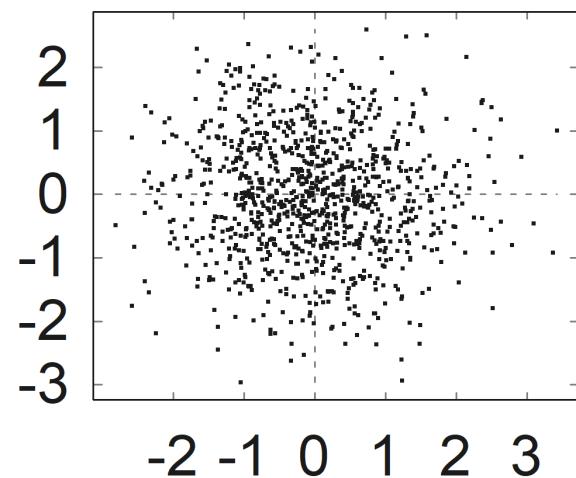
```
>k=0:10;  
>plot2d(k,bin(10,k),>bar):
```



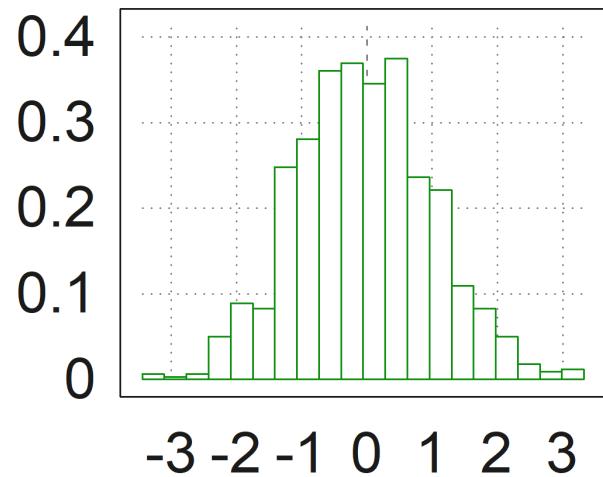
```
>plot2d(k,bin(10,k)); plot2d(k,bin(10,k),>points,>add):
```



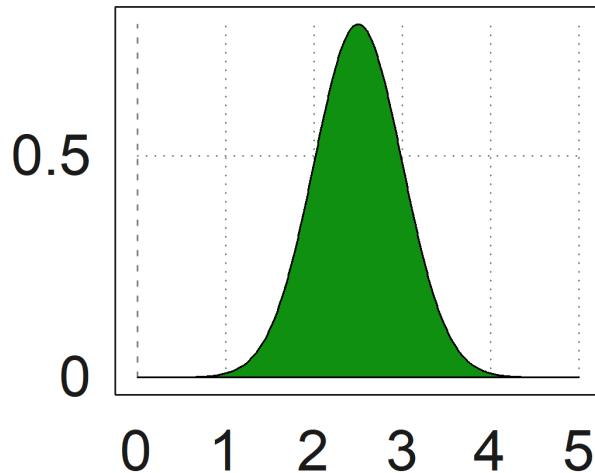
```
>plot2d(normal(1000),normal(1000),>points,grid=6,style=".."):
```



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

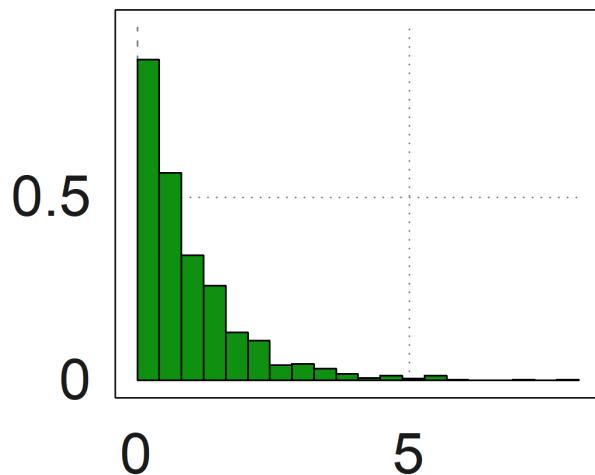


```
>plot2d("qnormal",0,5;2.5,0.5,>filled):
```



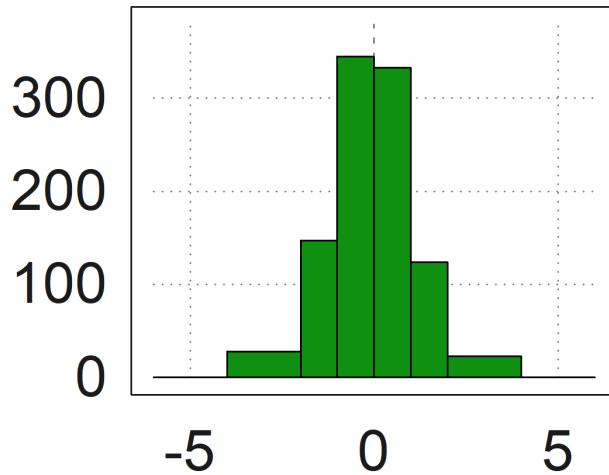
Untuk memplot distribusi statistik eksperimental, Anda dapat menggunakan distribution=n dengan plot2d.

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



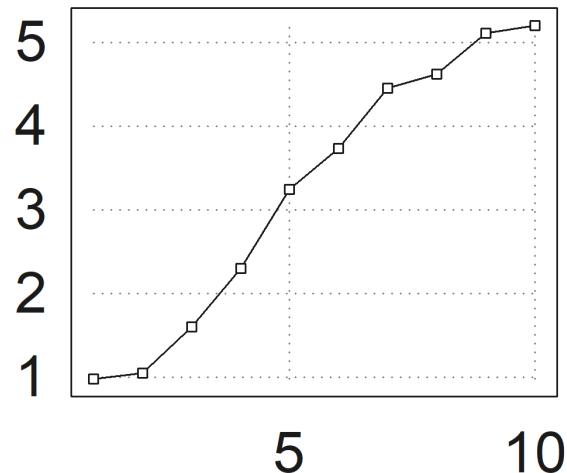
Atau Anda dapat menghitung distribusi dari data dan memplot hasilnya dengan >bar di plot3d, atau dengan plot kolom.

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

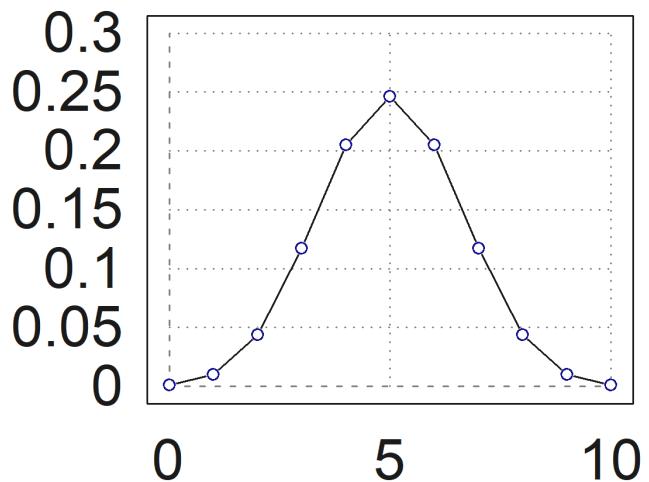


Fungsi statplot() menyetel gaya dengan string sederhana.

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



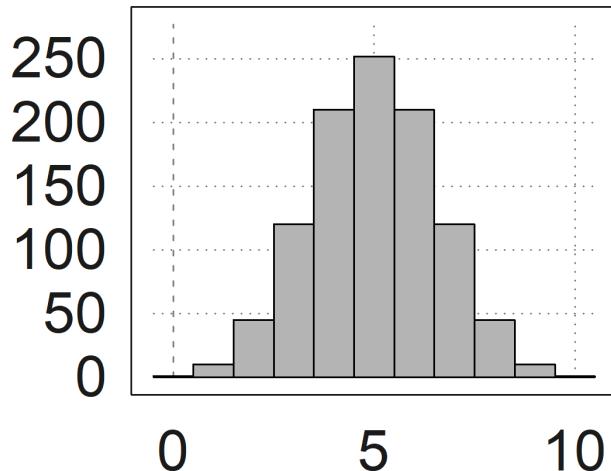
```
>n=10; i=0:n; ...
>plot2d(i,bin(n,i)/2^n,a=0,b=10,c=0,d=0.3); ...
>plot2d(i,bin(n,i)/2^n,points=true,style="ow",add=true,color=blue):
```



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

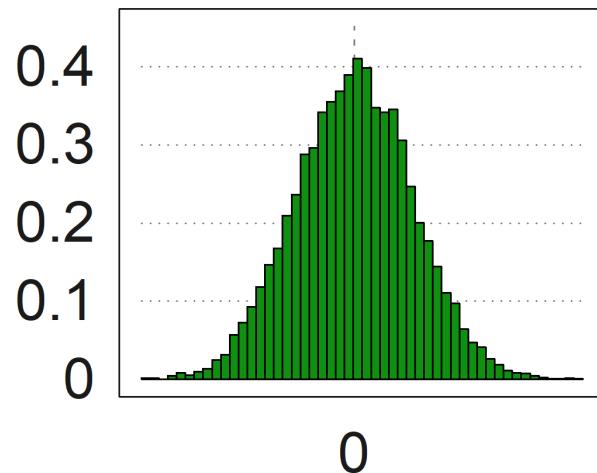
Gaya isian dapat digunakan seperti di atas.

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

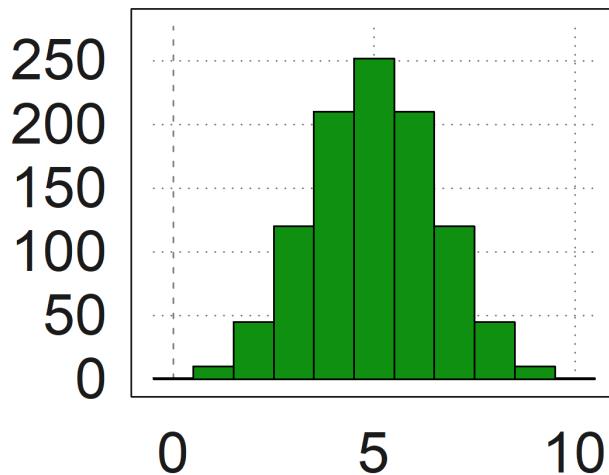


Data untuk plot batang (bar=1) dan histogram (histogram=1) dapat dinyatakan secara eksplisit dalam xv dan yv, atau dapat dihitung dari distribusi empiris dalam xv dengan >distribusi (atau distribusi=n). Histogram nilai xv akan dihitung secara otomatis dengan >histogram. Jika >genap ditentukan, nilai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

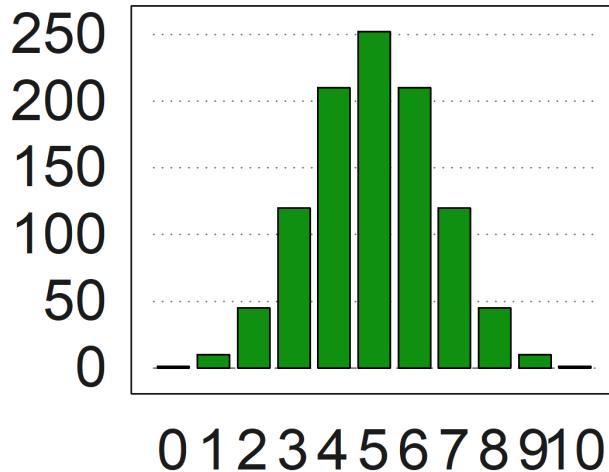
```
>plot2d(normal(10000),distribution=50):
```



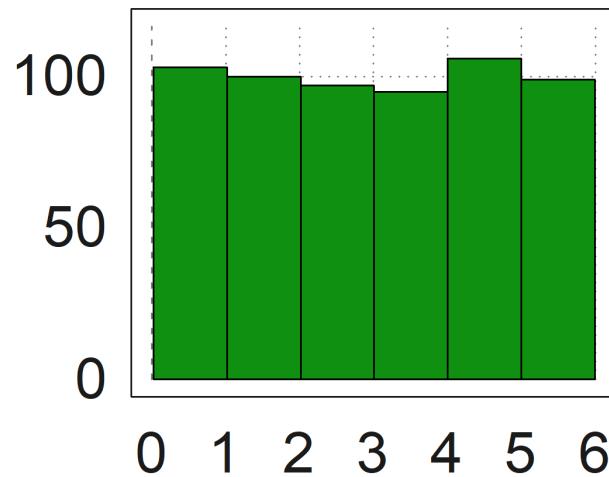
```
>k=0:10; m=bin(10,k); x=(0:11)-0.5; plot2d(x,m,>bar):
```



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

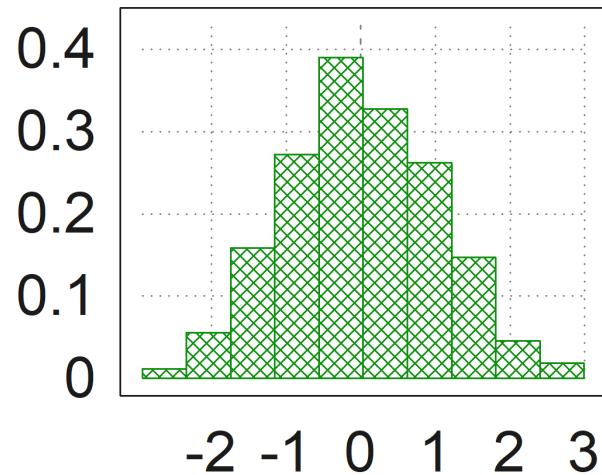


```
>plot2d(random(600)*6, histogram=6) :
```



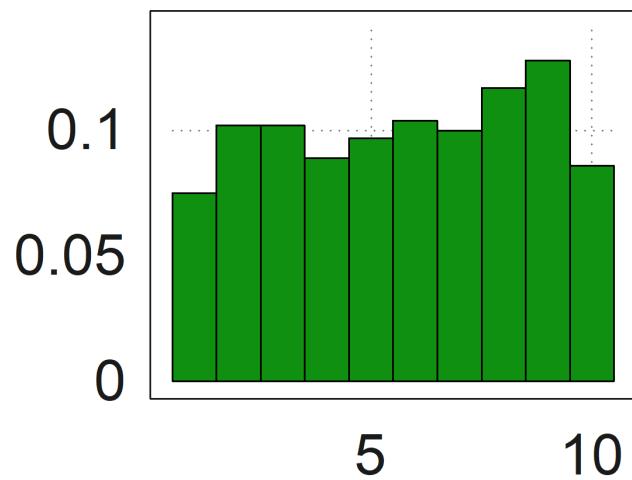
Untuk distribusi, ada parameter `distribution=n`, yang menghitung nilai secara otomatis dan mencetak distribusi relatif dengan `n` sub-interval.

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



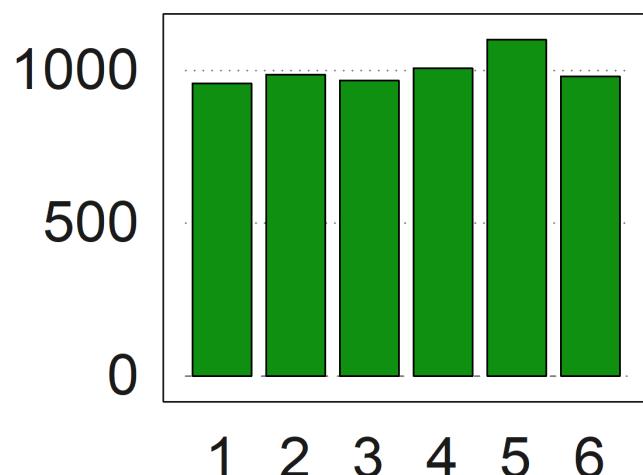
Dengan parameter even=true, ini akan menggunakan interval integer.

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

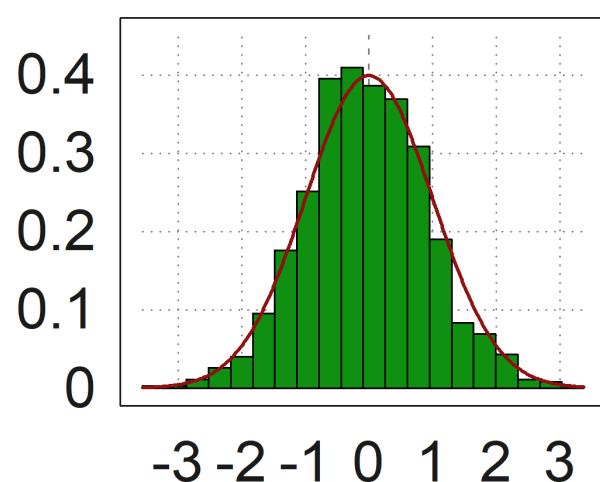


Perhatikan bahwa ada banyak plot statistik, yang mungkin berguna. Silahkan lihat tutorial tentang statistik.

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

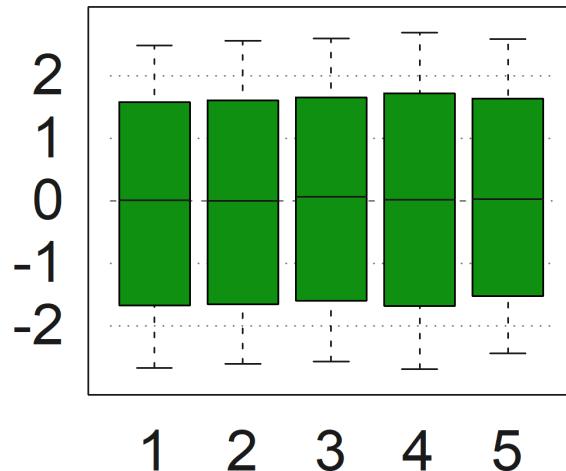


```
>plot2d(normal(1,1000),>distribution); ...
> plot2d("qnormal(x)",color=red,thickness=2,>add):
```



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

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



Fungsi Implisit

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

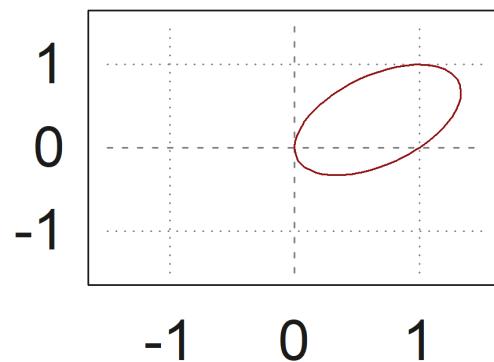
Euler dapat menandai garis level

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

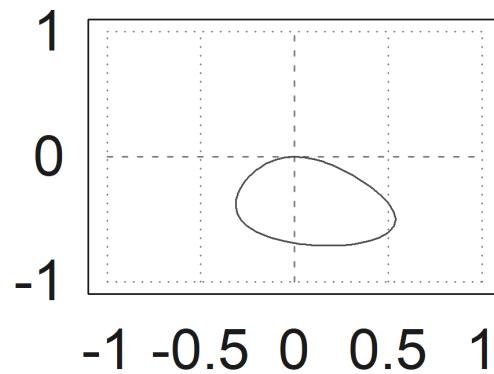
dari fungsi apapun.

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

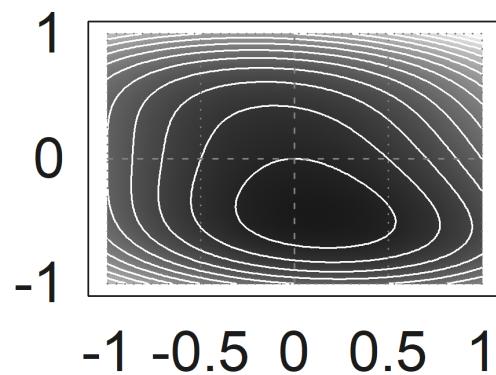
```
>aspect(1.5);
>plot2d("x^2+y^2-x*y-x", r=1.5, level=0, contourcolor=red):
```



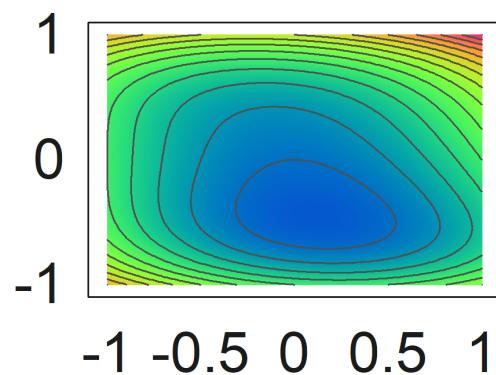
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=0); // Solutions of f(x,y)=0
```



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

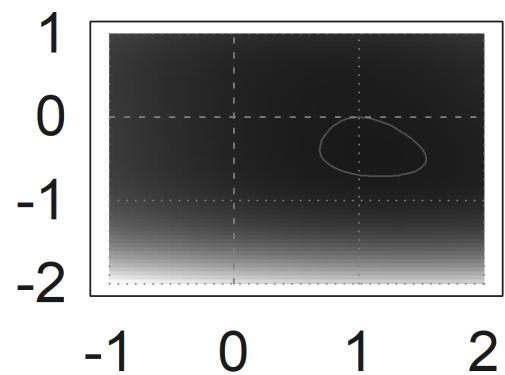


```
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=200,grid=4): // nicer
```

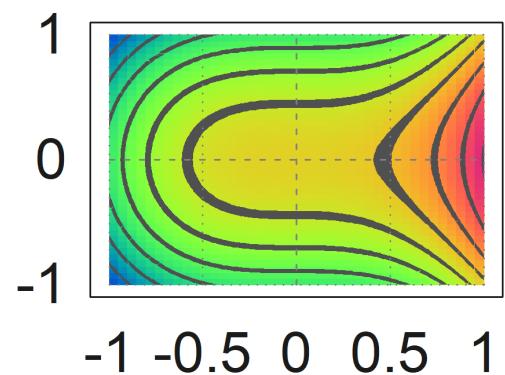


Ini berfungsi untuk plot data juga. Tetapi Anda harus menentukan rentangnya untuk label sumbu.

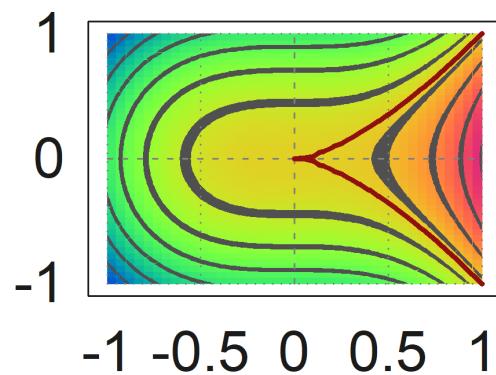
```
>x=-2:0.05:1; y=x'; z=expr(x,y);
>plot2d(z,level=0,a=-1,b=2,c=-2,d=1,>hue):
```



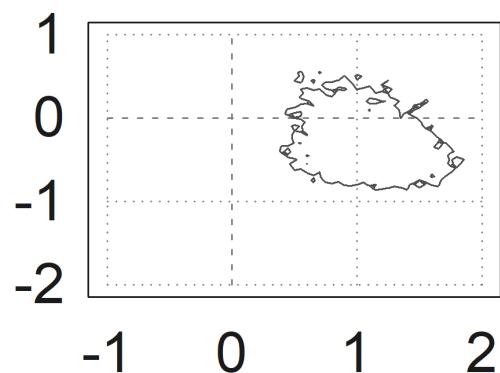
```
>plot2d("x^3-y^2",>contour,>hue,>spectral):
```



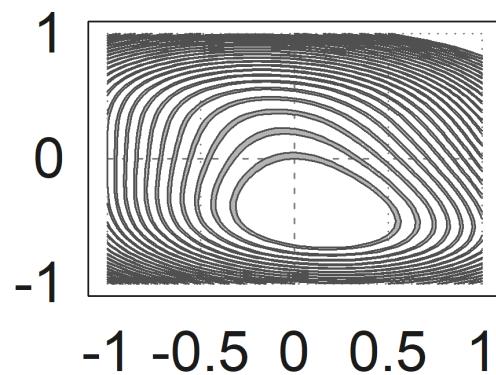
```
>plot2d("x^3-y^2",level=0,contourwidth=3,>add,contourcolor=red):
```



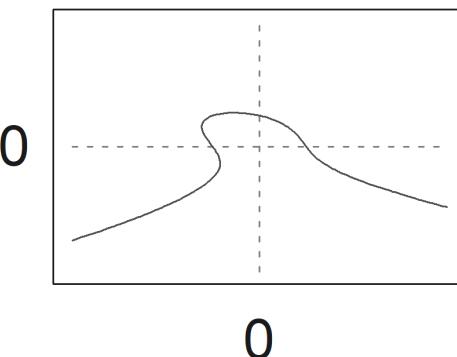
```
>z=z+normal(size(z))*0.2;  
>plot2d(z,level=0.5,a=-1,b=2,c=-2,d=1):
```



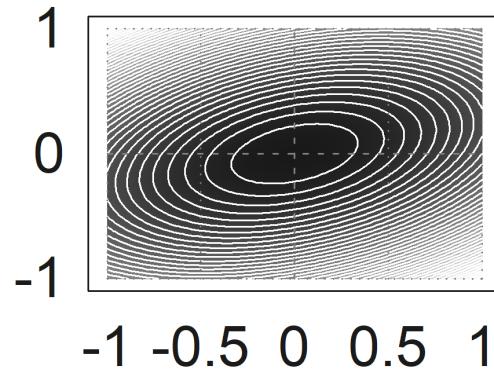
```
>plot2d(expr,level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
```



```
>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):
```



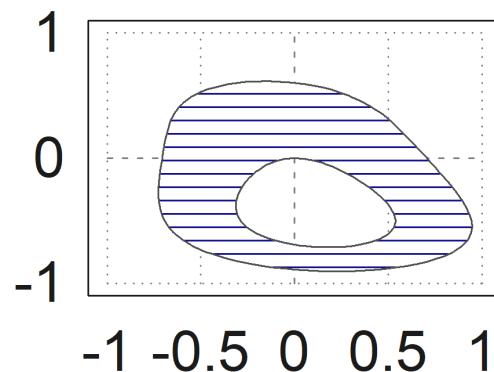
Juga dimungkinkan untuk mengisi set

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

dengan rentang tingkat.

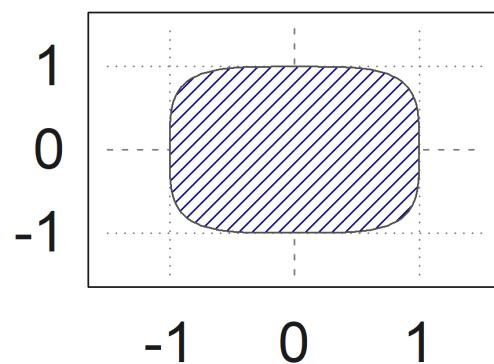
Dimungkinkan untuk mengisi wilayah nilai untuk fungsi tertentu. Untuk ini, level harus berupa matriks 2xn. Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

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

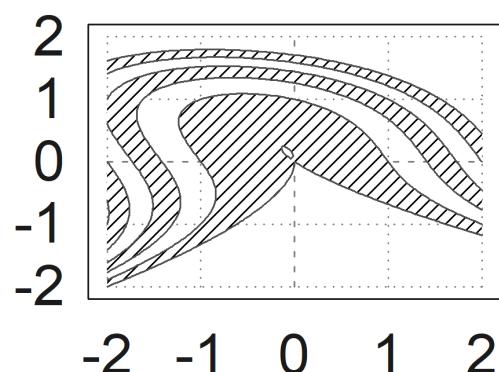


Plot implisit juga dapat menunjukkan rentang level. Kemudian level harus berupa matriks 2xn dari interval level, di mana baris pertama berisi awal dan baris kedua adalah akhir dari setiap interval. Atau, vektor baris sederhana dapat digunakan untuk level, dan parameter dl memperluas nilai level ke interval.

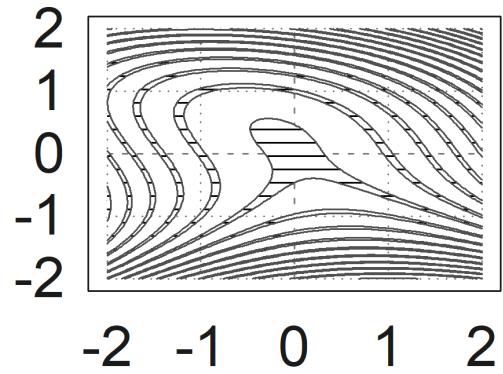
```
>plot2d("x^4+y^4",r=1.5,level=[0;1],color=blue,style="/"):
```



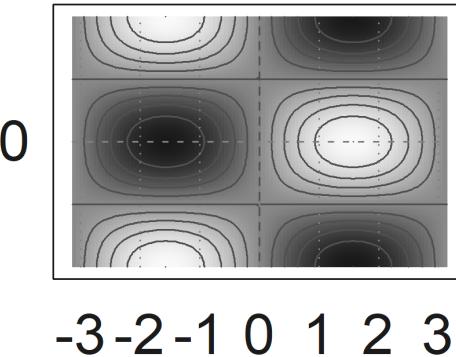
```
>plot2d("x^2+y^3+x*y",level=[0,2,4;1,3,5],style="/",r=2,n=100):
```



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

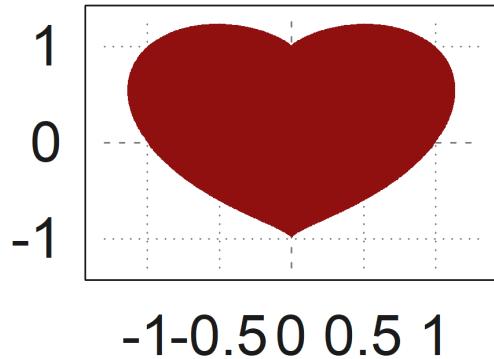


```
>plot2d("sin(x)*cos(y)", r=pi, >hue, >levels, n=100):
```



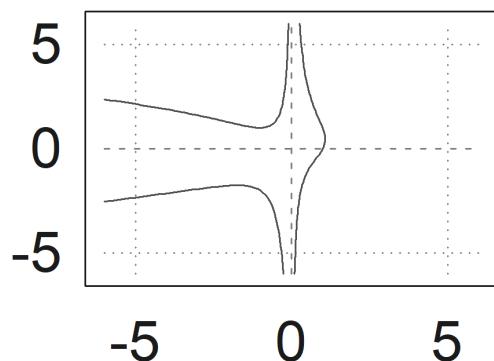
Dimungkinkan juga untuk menandai suatu wilayah
lateks: $a \leq f(x,y) \leq b$.
Ini dilakukan dengan menambahkan level dengan dua baris.

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



Dimungkinkan untuk menentukan level tertentu. Misalnya, kita dapat memplot solusi persamaan seperti lateks: $x^3-xy+x^2y^2=6$

```
>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
```

```

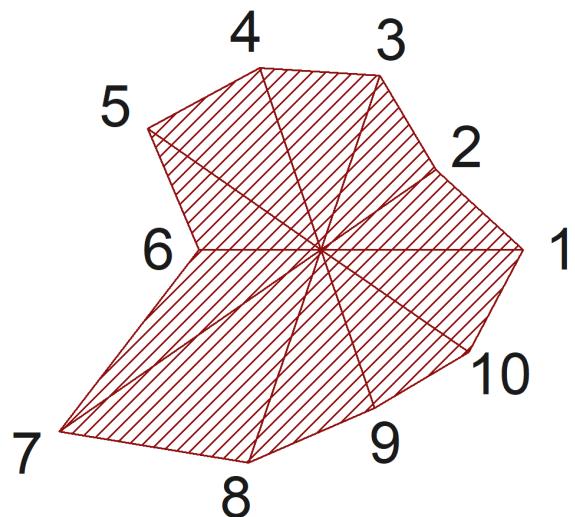
polygon([0,c[#],c[#+1]],[0,s[#],s[#+1]],1);
if lab!=none then
    rlab=v[#]+r*0.1;
    {col,row}=toscreen(cos(t[#])*rlab,sin(t[#])*rlab);
    ctext(""+lab[#],col,row-textheight()/2);
endif;
end;
barcolor(cl); barstyle(st);
holding(h);
window(w);
endfunction

```

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

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

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



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

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

```

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

```

```

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

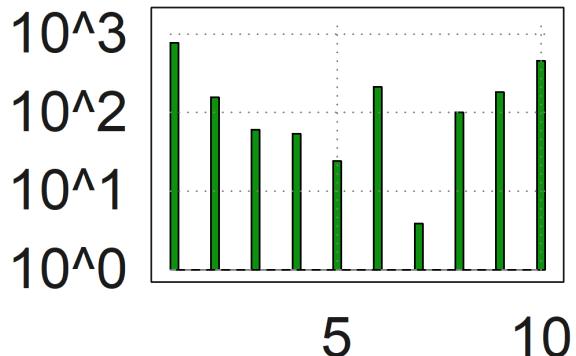
```

Mari kita uji dengan nilai yang terdistribusi secara eksponensial.

```

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

```



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

Fungsi wait(0) memaksa plot untuk muncul di jendela grafik. Jika tidak, menggambar ulang terjadi dalam interval waktu yang jarang.

```

>function animliss (n,m) ...

```

```

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

```

```
framecolor(c);
linewidth(1);
endfunction
```

Tekan sembarang tombol untuk menghentikan animasi ini.

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

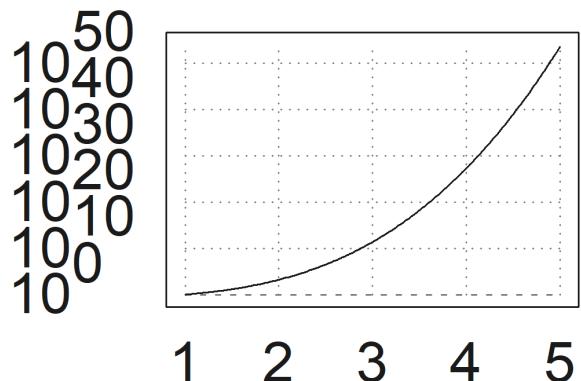
Plot Logaritmik

EMT menggunakan parameter "logplot" untuk skala logaritmik.

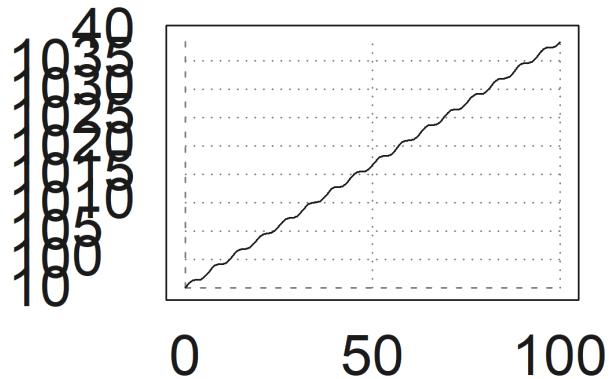
Plot logaritma dapat diplot baik menggunakan skala logaritma dalam y dengan logplot=1, atau menggunakan skala logaritma dalam x dan y dengan logplot=2, atau dalam x dengan logplot=3.

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

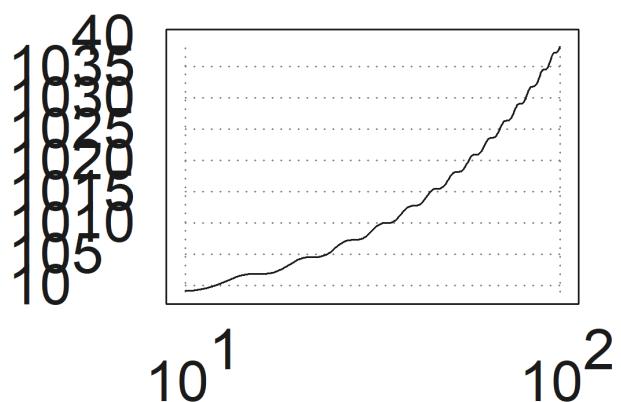
```
>plot2d("exp(x^3-x)*x^2",1,5,logplot=1):
```



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

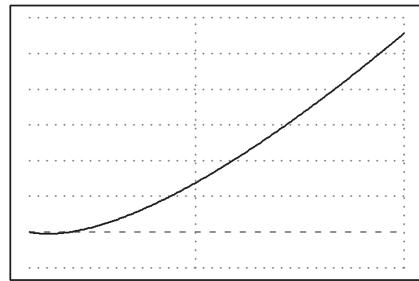


```
>plot2d("exp(x+sin(x))",10,100,logplot=2):
```



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

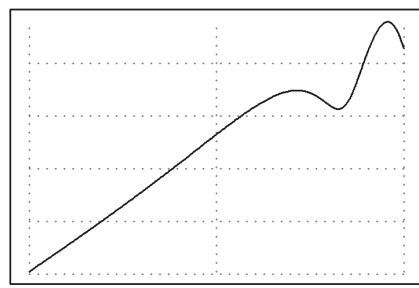
10^{-1}
 10^0
 10^1
 10^2
 10^3
 10^4
 10^5
 10^6



5 10

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

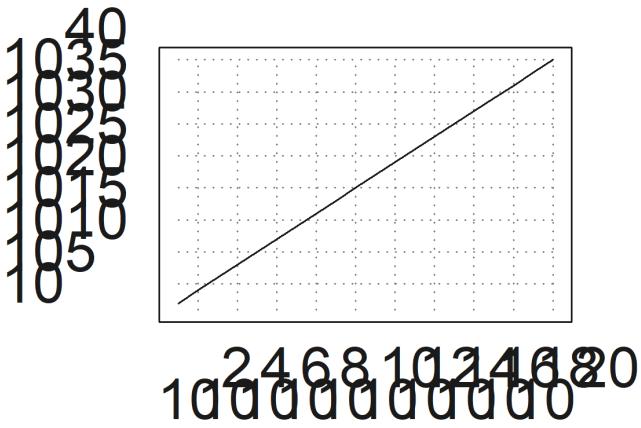
3
4
5
6
7



10^1 10^2 10^3

Ini juga berfungsi dengan plot data.

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



Rujukan Lengkap Fungsi plot2d()

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

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

Parameters

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

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

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

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

frame : 0 = no frame

framecolor: color of the frame and the grid

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

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

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

thickness : line thickness for curves

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

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

For points use

"[]", "<>", ".", "..", "...",
"*", "+", "|", "-", "o"
"[#]", "<>#", "o#" (filled shapes)
"[w]", "<>w", "ow" (non-transparent)

For lines use

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

For filled polygons or bar plots use

"#", "#O", "O", "/", "\", "\/",
"+", "|", "-", "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.

BAB 5

PENGGUNAAN SOFTWARE EMT UNTUK PLOT 3D

Menggambar Plot 3D dengan EMT

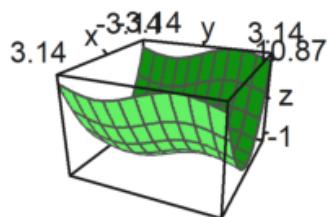
Ini adalah pengenalan plot 3D di Euler. Kita membutuhkan plot 3D untuk memvisualisasikan fungsi dari dua variabel.

Euler menggambar fungsi tersebut menggunakan algoritma pengurutan untuk menyembunyikan bagian di latar belakang. Secara umum, Euler menggunakan proyeksi pusat. Standarnya adalah dari kuadran x-y positif menuju titik asal $x=y=z=0$, tetapi sudut=0° terlihat dari arah sumbu y. Sudut pandang dan tinggi dapat diubah. Euler dapat merencanakan

- permukaan dengan bayangan dan garis level atau rentang level,
- awan poin,
- kurva parametrik,
- permukaan implisit.

Plot 3D dari suatu fungsi menggunakan plot3d. Cara termudah adalah dengan memplot ekspresi dalam x dan y. Parameter r mengatur kisaran plot di sekitar (0,0).

```
>aspect(1.5); plot3d("x^2+sin(y)",r=pi):
```



Fungsi dua Variabel

Untuk grafik fungsi, gunakan

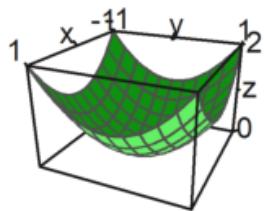
- ekspresi sederhana dalam x dan y,
- nama fungsi dari dua variabel
- atau matriks data.

Standarnya adalah kotak kawat yang diisi dengan warna berbeda di kedua sisi. Perhatikan bahwa jumlah default interval grid adalah 10, tetapi plot menggunakan jumlah default 40x40 persegi panjang untuk membangun permukaan. Ini bisa diubah.

- n=40, n=[40,40]: jumlah garis grid di setiap arah
- grid=10, grid=[10,10]: jumlah garis grid di setiap arah.

Kami menggunakan default n=40 dan grid=10.

```
>plot3d("x^2+y^2"):
```

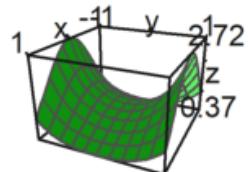


Interaksi pengguna dimungkinkan dengan >parameter pengguna. Pengguna dapat menekan tombol berikut.

- kiri, kanan, atas, bawah: putar sudut pandang
- +,-: memperbesar atau memperkecil
- a: menghasilkan anaglyph (lihat di bawah)
- l: beralih memutar sumber cahaya (lihat di bawah)
- spasi: reset ke default
- kembali: akhiri interaksi

```
>plot3d("exp(-x^2+y^2)",>user, ...
> title="Turn with the vector keys (press return to finish)":
```

ith the vector keys (press return to



Rentang plot untuk fungsi dapat ditentukan dengan

- a,b: rentang-x
- c,d: rentang-y
- r: persegi simetris di sekitar (0,0).
- n: jumlah subinterval untuk plot.

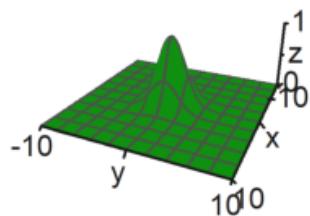
Ada beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.

fscale: skala ke nilai fungsi (defaultnya adalah <fscale>).

skala: angka atau vektor 1x2 untuk skala ke arah x dan y.

bingkai: jenis bingkai (default 1).

```
>plot3d("exp(-(x^2+y^2)/5)", r=10, n=80, fscale=4, scale=1.2, frame=3) :
```



Tampilan dapat diubah dengan berbagai cara.

- jarak: jarak pandang ke plot.
- zoom: nilai zoom.
- sudut: sudut terhadap sumbu y negatif dalam radian.
- tinggi: ketinggian tampilan dalam radian.

Nilai default dapat diperiksa atau diubah dengan fungsi view(). Ini mengembalikan parameter dalam urutan di atas.

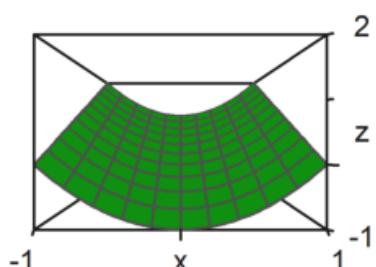
```
>view
```

```
[5, 2.6, 2, 0.4]
```

Jarak yang lebih dekat membutuhkan lebih sedikit zoom. Efeknya lebih seperti lensa sudut lebar.

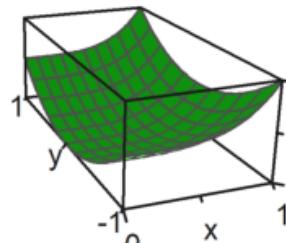
Dalam contoh berikut, sudut=0 dan tinggi=0 terlihat dari sumbu y negatif. Label sumbu untuk y disembunyikan dalam kasus ini.

```
>plot3d("x^2+y", distance=3, zoom=2, angle=0, height=0) :
```



Plot terlihat selalu ke pusat kubus plot. Anda dapat memindahkan pusat dengan parameter tengah.

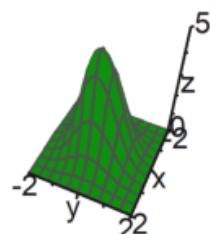
```
>plot3d("x^4+y^2",a=0,b=1,c=-1,d=1,angle=-20°,height=20°, ...
> center=[0.4,0,0],zoom=5):
```



Plot diskalakan agar sesuai dengan kubus satuan untuk dilihat. Jadi tidak perlu mengubah jarak atau zoom tergantung pada ukuran plot. Namun, label mengacu pada ukuran sebenarnya.

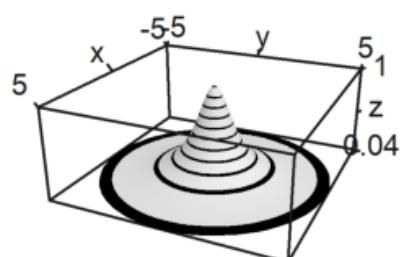
Jika Anda mematikannya dengan scale=false, Anda perlu berhati-hati, bahwa plot masih cocok dengan jendela plot, dengan mengubah jarak pandang atau zoom, dan memindahkan pusat.

```
>plot3d("5*exp(-x^2-y^2)",r=2,<fscale,<scale,distance=13,height=50°, ...
> center=[0,0,-2],frame=3):
```

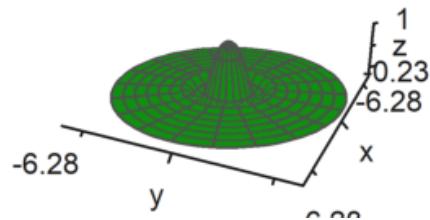


Sebuah plot kutub juga tersedia. Parameter polar=true menggambar plot polar. Fungsi tersebut harus tetap merupakan fungsi dari x dan y. Parameter "fscale" menskalakan fungsi dengan skala sendiri. Jika tidak, fungsi diskalakan agar sesuai dengan kubus.

```
>plot3d("1/(x^2+y^2+1)",r=5,>polar, ...
>fscale=2,>hue,n=100,zoom=4,>contour,color=gray):
```



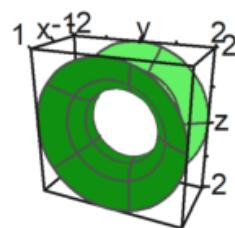
```
>function f(r) := exp(-r/2)*cos(r); ...
>plot3d("f(x^2+y^2)",>polar,scale=[1,1,0.4],r=2pi,frame=3,zoom=4):
```



Rotasi parameter memutar fungsi dalam x di sekitar sumbu x.

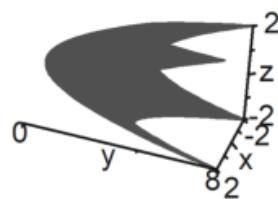
- rotate=1: Menggunakan sumbu x
- rotate=2: Menggunakan sumbu z

```
>plot3d("x^2+1",a=-1,b=1,rotate=true,grid=5):
```



Berikut adalah plot dengan tiga fungsi.

```
>plot3d("x", "x^2+y^2", "y", r=2, zoom=3.5, frame=3):
```



Plot Kontur

Untuk plot, Euler menambahkan garis grid. Sebagai gantinya dimungkinkan untuk menggunakan garis level dan rona satu warna atau rona berwarna spektral. Euler dapat menggambar tinggi fungsi pada plot dengan bayangan. Di semua plot 3D, Euler dapat menghasilkan anaglyph merah/sian.

-> hue: Menyalakan bayangan cahaya alih-alih kabel.

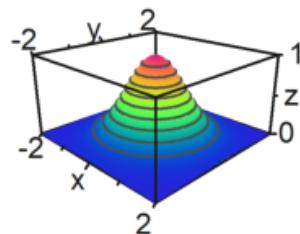
-> kontur: Memplot garis kontur otomatis pada plot.

- level=... (atau level): Sebuah vektor nilai untuk garis kontur.

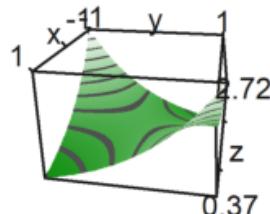
Standarnya adalah level="auto", yang menghitung beberapa garis level secara otomatis. Seperti yang Anda lihat di plot, level sebenarnya adalah rentang level.

Gaya default dapat diubah. Untuk plot kontur berikut, kami menggunakan grid yang lebih halus untuk 100x100 poin, skala fungsi dan plot, dan menggunakan sudut pandang yang berbeda.

```
>plot3d("exp(-x^2-y^2)",r=2,n=100,level="thin", ...
>>contour,>spectral,fscale=1,scale=1.1,angle=45°,height=20°) :
```



```
>plot3d("exp(x*y)",angle=100°,>contour,color=green) :
```



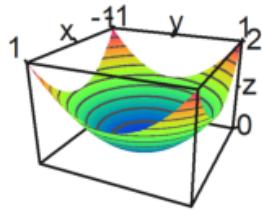
Bayangan default menggunakan warna abu-abu. Tetapi rentang warna spektral juga tersedia.

-> spektral: Menggunakan skema spektral default

- color=...: Menggunakan warna khusus atau skema spektral

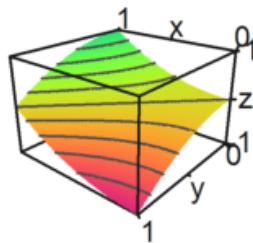
Untuk plot berikut, kami menggunakan skema spektral default dan menambah jumlah titik untuk mendapatkan tampilan yang sangat halus.

```
>plot3d("x^2+y^2",>spectral,>contour,n=100) :
```



Alih-alih garis level otomatis, kita juga dapat mengatur nilai garis level. Ini akan menghasilkan garis level tipis alih-alih rentang level.

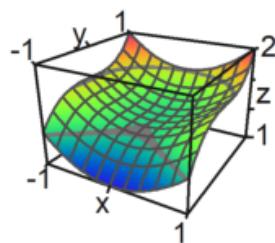
```
>plot3d("x^2-y^2",0,1,0,1,angle=220°,level=-1:0.2:1,color=redgreen):
```



Dalam plot berikut, kami menggunakan dua pita level yang sangat luas dari -0,1 hingga 1, dan dari 0,9 hingga 1. Ini dimasukkan sebagai matriks dengan batas level sebagai kolom.

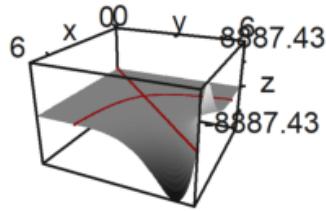
Selain itu, kami melapisi kisi dengan 10 interval di setiap arah.

```
>plot3d("x^2+y^3",level=[-0.1,0.9;0,1], ...
> >spectral,angle=30°,grid=10,contourcolor=gray):
```



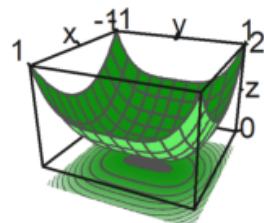
Dalam contoh berikut, kami memplot himpunan, di mana
lateks: $f(x,y) = x^y - y^x = 0$
Kami menggunakan satu garis tipis untuk garis level.

```
>plot3d("x^y-y^x",level=0,a=0,b=6,c=0,d=6,contourcolor=red,n=100):
```



Dimungkinkan untuk menunjukkan bidang kontur di bawah plot. Warna dan jarak ke plot dapat ditentukan.

```
>plot3d("x^2+y^4", >cp, cpcolor=green, cpdelta=0.2) :
```



Here are a few more styles. We always turn off the frame, and use various color schemes for the plot and the grid.

```
>figure(2,2); ...
```

Dimungkinkan untuk menunjukkan bidang kontur di bawah plot. Warna dan jarak ke plot dapat ditentukan.

```
>expr="y^3-x^2"; ...
>figure(1); ...
> plot3d(expr,<frame,>cp,cpcolor=spectral); ...
>figure(2); ...
> plot3d(expr,<frame,>spectral,grid=10,cp=2); ...
>figure(3); ...
> plot3d(expr,<frame,>contour,color=gray,nc=5,cp=3,cpcolor=greenred); ...
>figure(4); ...
> plot3d(expr,<frame,>hue,grid=10,>transparent,>cp,cpcolor=gray); ...
>figure(0):
```

Ada beberapa skema spektral lainnya, bermotor dari 1 hingga 9. Tetapi Anda juga dapat menggunakan warna=nilai, di mana nilai

- spektral: untuk rentang dari biru ke merah
- putih: untuk rentang yang lebih redup
- kuningbiru, ungu, hijau, birukuning, hijaumerah
- birukuning, hijau ungu, kuning biru, merah hijau

```
>figure(3,3); ...
>for i=1:9; ...
>  figure(i); plot3d("x^2+y^2",spectral=i,>contour,>cp,<frame,zoom=4); ...
>end; ...
>figure(0):
```

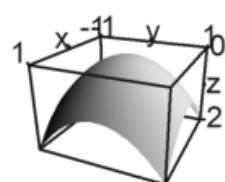
Sumber cahaya dapat diubah dengan l dan tombol kursor selama interaksi pengguna. Itu juga dapat diatur dengan parameter.

- cahaya: arah untuk cahaya
- amb: cahaya sekitar antara 0 dan 1

Perhatikan bahwa program tidak membuat perbedaan antara sisi plot. Tidak ada bayangan. Untuk ini, Anda perlu Povray.

```
>plot3d("-x^2-y^2", ...
>  hue=true,light=[0,1,1],amb=0,user=true, ...
>  title="Press l and cursor keys (return to exit)":
```

Press l and cursor keys (return to exit)



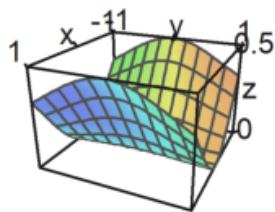
Parameter warna mengubah warna permukaan. Warna garis level juga dapat diubah.

```
>plot3d("-x^2-y^2",color=rgb(0.2,0.2,0),hue=true,frame=false, ...
>  zoom=3,contourcolor=red,level=-2:0.1:1,dl=0.01):
```



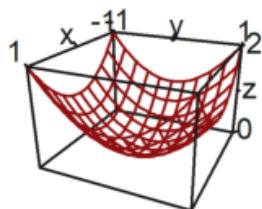
Warna 0 memberikan efek pelangi khusus.

```
>plot3d("x^2/(x^2+y^2+1)",color=0,hue=true,grid=10):
```



Permukaannya juga bisa transparan.

```
>plot3d("x^2+y^2",>transparent,grid=10,wirecolor=red):
```



Plot Implisit

Ada juga plot implisit dalam tiga dimensi. Euler menghasilkan pemotongan melalui objek. Fitur plot3d termasuk plot implisit. Plot-plot ini menunjukkan himpunan nol dari suatu fungsi dalam tiga variabel. Solusi dari

lateks: $f(x,y,z) = 0$

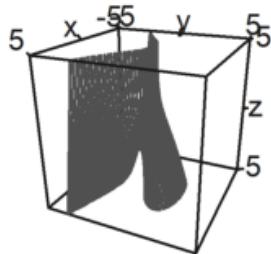
dapat divisualisasikan dalam potongan sejajar dengan bidang x-y-, x-z- dan y-z.

- implisit=1: potong sejajar dengan bidang y-z
- implisit=2: potong sejajar dengan bidang x-z
- implisit=4: potong sejajar dengan bidang x-y

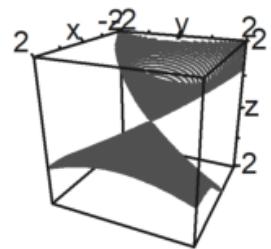
Tambahkan nilai-nilai ini, jika Anda suka. Dalam contoh kita plot

lateks: $M = \{(x,y,z) : x^2+y^3+zy=1\}$

```
>plot3d("x^2+y^3+z*y-1", r=5, implicit=3):
```



```
>plot3d("x^2+y^2+4*x*z+z^3", >implicit, r=2, zoom=2.5):
```



Merencanakan Data 3D

Sama seperti plot2d, plot3d menerima data. Untuk objek 3D, Anda perlu menyediakan matriks nilai x-, y- dan z, atau tiga fungsi atau ekspresi $fx(x,y)$, $fy(x,y)$, $fz(x,y)$.

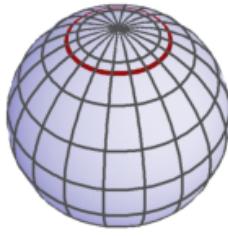
lateks: $\gamma(t,s) = (x(t,s), y(t,s), z(t,s))$

Karena x, y, z adalah matriks, kita asumsikan bahwa (t,s) melalui sebuah kotak persegi. Hasilnya, Anda dapat memplot gambar persegi panjang di ruang angkasa.

Anda dapat menggunakan bahasa matriks Euler untuk menghasilkan koordinat secara efektif.

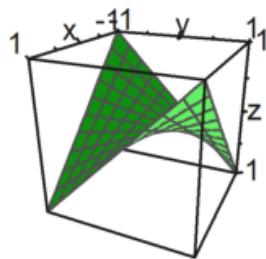
Dalam contoh berikut, kami menggunakan vektor nilai t dan vektor kolom nilai s untuk membuat parameter permukaan bola. Dalam gambar kita dapat menandai daerah, dalam kasus kita daerah kutub.

```
>t=linspace(0,2pi,180); s=linspace(-pi/2,pi/2,90)'; ...
>x=cos(s)*cos(t); y=cos(s)*sin(t); z=sin(s); ...
>plot3d(x,y,z,>hue, ...
>color=blue,<frame,grid=[10,20], ...
>values=s,contourcolor=red,level=[90°-24°;90°-22°], ...
>scale=1.4,height=50):
```



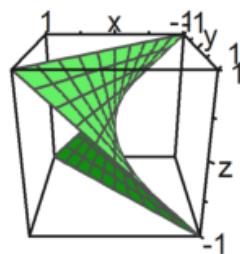
Berikut adalah contoh, yang merupakan grafik fungsi.

```
>t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,t*s,grid=10):
```



Namun, kita bisa membuat segala macam permukaan. Berikut adalah permukaan yang sama dengan fungsi lateks: $x = y \setminus z$

```
>plot3d(t*s,t,s,angle=180°,grid=10):
```



Dengan lebih banyak usaha, kami dapat menghasilkan banyak permukaan.

Dalam contoh berikut, kita membuat tampilan bayangan dari bola yang terdistorsi. Koordinat biasa untuk bola adalah

lateks: $\gamma(t,s) = (\cos(t)\cos(s), \sin(t)\cos(s), \sin(s))$

dengan

lateks: $0 \leq t \leq 2\pi, \quad \frac{-\pi}{2} \leq s \leq \frac{\pi}{2}$.

Kami mendistorsi ini dengan sebuah faktor

lateks: $d(t,s) = \frac{\cos(4t) + \cos(8s)}{4}$.

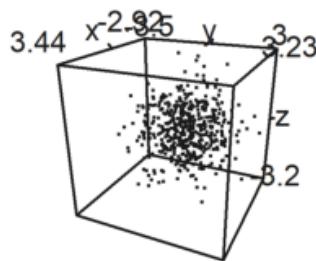
```
>t=linspace(0,2pi,320); s=linspace(-pi/2,pi/2,160)'; ...
>d=1+0.2*(cos(4*t)+cos(8*s)); ...
>plot3d(cos(t)*cos(s)*d,sin(t)*cos(s)*d,sin(s)*d,hue=1, ...
> light=[1,0,1],frame=0,zoom=5):
```



Tentu saja, titik cloud juga dimungkinkan. Untuk memplot data titik dalam ruang, kita membutuhkan tiga vektor untuk koordinat titik-titik tersebut.

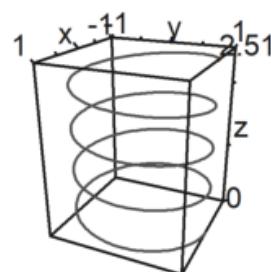
Gayanya sama seperti di plot2d dengan points=true;

```
>n=500; ...
> plot3d(normal(1,n),normal(1,n),normal(1,n),points=true,style="."):
```

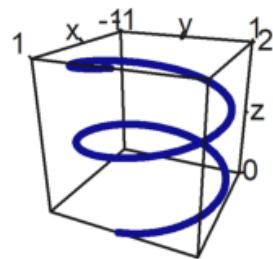


Dimungkinkan juga untuk memplot kurva dalam 3D. Dalam hal ini, lebih mudah untuk menghitung titik-titik kurva. Untuk kurva di pesawat kami menggunakan urutan koordinat dan parameter wire=true.

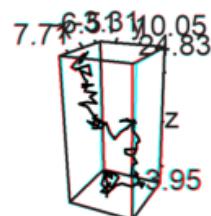
```
>t=linspace(0,8pi,500); ...
>plot3d(sin(t),cos(t),t/10,>wire,zoom=3):
```



```
>t=linspace(0,4pi,1000); plot3d(cos(t),sin(t),t/2pi,>wire, ...
>lineWidth=3, wirecolor=blue):
```

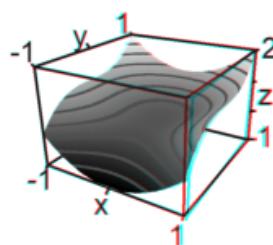


```
>X=cumsum(normal(3,100)); ...
> plot3d(X[1],X[2],X[3],>anaglyph,>wire):
```



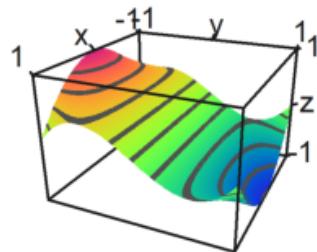
EMT juga dapat memplot dalam mode anaglyph. Untuk melihat plot seperti itu, Anda memerlukan kacamata merah/sian.

```
> plot3d("x^2+y^3",>anaglyph,>contour,angle=30°):
```



Seringkali, skema warna spektral digunakan untuk plot. Ini menekankan ketinggian fungsi.

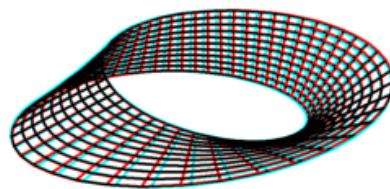
```
>plot3d("x^2*y^3-y",>spectral,>contour,zoom=3.2):
```



Euler juga dapat memplot permukaan berparameter, ketika parameternya adalah nilai x-, y-, dan z dari gambar kotak persegi panjang dalam ruang.

Untuk demo berikut, kami mengatur parameter u- dan v-, dan menghasilkan koordinat ruang dari ini.

```
>u=linspace(-1,1,10); v=linspace(0,2*pi,50)'; ...
>X=(3+u*cos(v/2))*cos(v); Y=(3+u*cos(v/2))*sin(v); Z=u*sin(v/2); ...
>plot3d(X,Y,Z,>anaglyph,<frame,>wire,scale=2.3):
```



Berikut adalah contoh yang lebih rumit, yang megah dengan kacamata merah/sian.

```
>u:=linspace(-pi,pi,160); v:=linspace(-pi,pi,400)'; ...
>x:=(4*(1+.25*sin(3*v))+cos(u))*cos(2*v); ...
>y:=(4*(1+.25*sin(3*v))+cos(u))*sin(2*v); ...
>z=sin(u)+2*cos(3*v); ...
>plot3d(x,y,z,frame=0,scale=1.5,hue=1,light=[1,0,-1],zoom=2.8,>anaglyph):
```



Plot Statistik

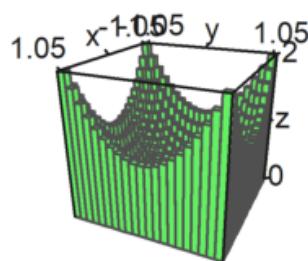
Plot bar juga dimungkinkan. Untuk ini, kita harus menyediakan

- x: vektor baris dengan n+1 elemen
- y: vektor kolom dengan n+1 elemen
- z: matriks nilai nxn.

z bisa lebih besar, tetapi hanya nilai nxn yang akan digunakan.

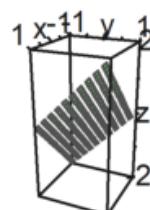
Dalam contoh, pertama-tama kita menghitung nilainya. Kemudian kita sesuaikan x dan y, sehingga vektor berpusat pada nilai yang digunakan.

```
>x=-1:0.1:1; y=x'; z=x^2+y^2; ...
>xa=(x|1.1)-0.05; ya=(y_1.1)-0.05; ...
>plot3d(xa,ya,z,bar=true);
```



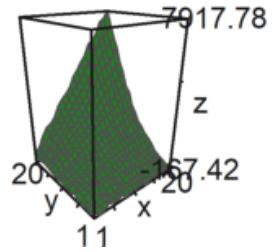
Dimungkinkan untuk membagi plot permukaan menjadi dua atau lebih bagian.

```
>x=-1:0.1:1; y=x'; z=x+y; d=zeros(size(x)); ...
>plot3d(x,y,z,disconnect=2:2:20);
```

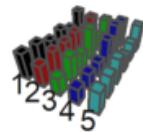


Jika memuat atau menghasilkan matriks data M dari file dan perlu memplotnya dalam 3D, Anda dapat menskalakan matriks ke [-1,1] dengan scale(M), atau menskalakan matriks dengan >zscale. Ini dapat dikombinasikan dengan faktor penskalaan individu yang diterapkan sebagai tambahan.

```
>i=1:20; j=i'; ...
>plot3d(i*j^2+100*normal(20,20),>zscale,scale=[1,1,1.5],angle=-40°,zoom=1.8);
```

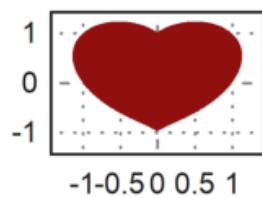


```
>Z=intrandom(5,100,6); v=zeros(5,6); ...
>loop 1 to 5; v[#]=getmultiplicities(1:6,Z[#]); end; ...
>columnsplot3d(v',scols=1:5,ccols=[1:5]):
```



Permukaan Benda Putar

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



```
>ekspressi &= (x^2+y^2-1)^3-x^2*y^3; $ekspressi
```

$$(y^2 + x^2 - 1)^3 - x^2 y^3$$

Kami ingin memutar kurva jantung di sekitar sumbu y. Berikut adalah ungkapan, yang mendefinisikan hati:
lateks: $f(x,y)=(x^2+y^2-1)^3-x^2.y^3$.

Selanjutnya kita atur

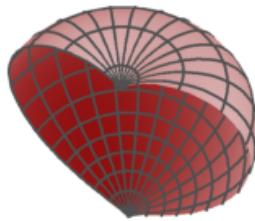
lateks: $x=r.\cos(a), \quad y=r.\sin(a)$.

```
>function fr(r,a) &= ekspresi with [x=r*cos(a),y=r*sin(a)] | trigreduce; $fr(r,a)
```

$$(r^2 - 1)^3 + \frac{(\sin(5a) - \sin(3a) - 2\sin a) r^5}{16}$$

Hal ini memungkinkan untuk mendefinisikan fungsi numerik, yang memecahkan r, jika a diberikan. Dengan fungsi itu kita dapat memplot jantung yang diputar sebagai permukaan parametrik.

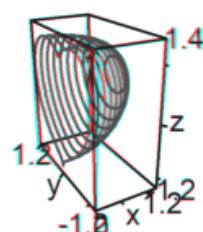
```
>function map f(a) := bisect("fr",0,2;a); ...
>t=linspace(-pi/2,pi/2,100); r=f(t); ...
>s=linspace(pi,2pi,100)'; ...
>plot3d(r*cos(t)*sin(s),r*cos(t)*cos(s),r*sin(t), ...
>>hue,<frame,color=red,zoom=4,amb=0,max=0.7,grid=12,height=50°):
```



Berikut ini adalah plot 3D dari gambar di atas yang diputar di sekitar sumbu z. Kami mendefinisikan fungsi, yang menggambarkan objek.

```
>function f(x,y,z) ...
r=x^2+y^2;
return (r+z^2-1)^3-r*z^3;
endfunction
```

```
>plot3d("f(x,y,z)", ...
>xmin=0,xmax=1.2,ymin=-1.2,ymax=1.2,zmin=-1.2,zmax=1.4, ...
>implicit=1,angle=-30°,zoom=2.5,n=[10,60,60],>anaglyph):
```



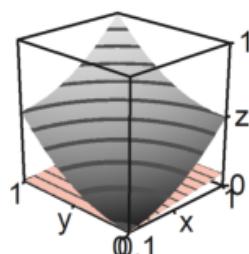
Fungsi plot3d bagus untuk dimiliki, tetapi tidak memenuhi semua kebutuhan. Selain rutinitas yang lebih mendasar, dimungkinkan untuk mendapatkan plot berbingkai dari objek apa pun yang Anda suka.

Meskipun Euler bukan program 3D, ia dapat menggabungkan beberapa objek dasar. Kami mencoba memvisualisasikan paraboloid dan garis singgungnya.

```
>function myplot ...
y=0:0.01:1; x=(0.1:0.01:1)';
plot3d(x,y,0.2*(x-0.1)/2,<scale,<frame,>hue, ...
    hues=0.5,>contour,color=orange);
h=holding(1);
plot3d(x,y,(x^2+y^2)/2,<scale,<frame,>contour,>hue);
holding(h);
endfunction
```

Sekarang framedplot() menyediakan frame, dan mengatur tampilan.

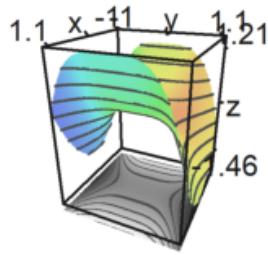
```
>framedplot("myplot", [0.1,1,0,1,0,1],angle=-45°, ...
> center=[0,0,-0.7],zoom=6):
```



Dengan cara yang sama, Anda dapat memplot bidang kontur secara manual. Perhatikan bahwa plot3d() menyetel jendela ke fullwindow() secara default, tetapi plotcontourplane() mengasumsikan itu.

```
>x=-1:0.02:1.1; y=x'; z=x^2-y^4;
>function myplot (x,y,z) ...
zoom(2);
wi=fullwindow();
plotcontourplane(x,y,z,level="auto",<scale);
plot3d(x,y,z,>hue,<scale,>add,color=white,level="thin");
window(wi);
reset();
endfunction

>myplot(x,y,z);
```

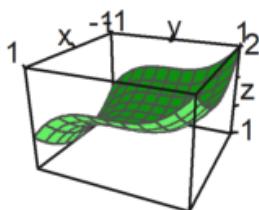


Animasi

Euler dapat menggunakan frame untuk menghitung animasi terlebih dahulu.

Salah satu fungsi yang memanfaatkan teknik ini adalah `rotate`. Itu dapat mengubah sudut pandang dan menggambar ulang plot 3D. Fungsi memanggil `addpage()` untuk setiap plot baru. Akhirnya itu menjawai plot. Silakan pelajari sumber rotasi untuk melihat lebih detail.

```
>function testplot () := plot3d("x^2+y^3"); ...
>rotate("testplot"); testplot();
```



*Menggambar Povray

Dengan bantuan file Euler povray.e, Euler dapat menghasilkan file Povray. Hasilnya sangat bagus untuk dilihat.

Anda perlu menginstal Povray (32bit atau 64bit) dari <http://www.povray.org/>, dan meletakkan sub-direktori "bin" dari Povray ke jalur lingkungan, atau mengatur variabel "defaultpovray" dengan path lengkap yang menunjuk ke "pvengine.exe".

Antarmuka Povray dari Euler menghasilkan file Povray di direktori home pengguna, dan memanggil Povray untuk mengurai file-file ini. Nama file default adalah `current.pov`, dan direktori default adalah `eulerhome()`, biasanya `c:\Users\Username\Euler`. Povray menghasilkan file PNG, yang dapat dimuat oleh Euler ke dalam buku catatan. Untuk membersihkan file-file ini, gunakan `povclear()`.

Fungsi `pov3d` memiliki semangat yang sama dengan `plot3d`. Ini dapat menghasilkan grafik fungsi $f(x,y)$, atau permukaan dengan koordinat X,Y,Z dalam matriks, termasuk garis level opsional. Fungsi ini memulai raytracer secara otomatis, dan memuat adegan ke dalam notebook Euler.

Selain `pov3d()`, ada banyak fungsi yang menghasilkan objek Povray. Fungsi-fungsi ini mengembalikan string, yang berisi kode Povray untuk objek. Untuk menggunakan fungsi ini, mulai file Povray dengan `povstart()`. Kemudian gunakan `writeln(...)` untuk menulis objek ke file adegan. Terakhir, akhiri file dengan `povend()`. Secara default, raytracer akan dimulai, dan PNG akan dimasukkan ke dalam notebook Euler.

Fungsi objek memiliki parameter yang disebut "look", yang membutuhkan string dengan kode Povray untuk tekstur dan hasil akhir objek. Fungsi `povlook()` dapat digunakan untuk menghasilkan string ini. Ini memiliki parameter untuk warna, transparansi, Phong Shading dll.

Perhatikan bahwa alam semesta Povray memiliki sistem koordinat lain. Antarmuka ini menerjemahkan semua koordinat ke sistem Povray. Jadi Anda dapat terus berpikir dalam sistem koordinat Euler dengan z menunjuk

vertikal ke atas, dan x,y,z sumbu dalam arti tangan kanan.
Anda perlu memuat file povray.

```
>load povray;
```

Pastikan, direktori bin Povray ada di jalur. Jika tidak, edit variabel berikut sehingga berisi path ke povray yang dapat dieksekusi.

```
>defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

```
C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe
```

Untuk kesan pertama, kami memplot fungsi sederhana. Perintah berikut menghasilkan file povray di direktori pengguna Anda, dan menjalankan Povray untuk ray tracing file ini.

Jika Anda memulai perintah berikut, GUI Povray akan terbuka, menjalankan file, dan menutup secara otomatis. Karena alasan keamanan, Anda akan ditanya, apakah Anda ingin mengizinkan file exe untuk dijalankan. Anda dapat menekan batal untuk menghentikan pertanyaan lebih lanjut. Anda mungkin harus menekan OK di jendela Povray untuk mengakui dialog awal Povray.

```
>pov3d("x^2+y^2", zoom=3);
```

Kita dapat membuat fungsi menjadi transparan dan menambahkan hasil akhir lainnya. Kami juga dapat menambahkan garis level ke plot fungsi.

```
>pov3d("x^2+y^3", axiscolor=red, angle=20°, ...
> look=povlook(blue, 0.2), level=-1:0.5:1, zoom=3.8);
```

Terkadang perlu untuk mencegah penskalaan fungsi, dan menskalakan fungsi dengan tangan.
Kami memplot himpunan titik di bidang kompleks, di mana produk dari jarak ke 1 dan -1 sama dengan 1.

```
>pov3d("((x-1)^2+y^2)*((x+1)^2+y^2)/40", r=1.5, ...
> angle=-120°, level=1/40, dlevel=0.005, light=[-1,1,1], height=45°, n=50, ...
> <fscale, zoom=3.8);
```

Merencanakan dengan Koordinat

Alih-alih fungsi, kita dapat memplot dengan koordinat. Seperti pada plot3d, kita membutuhkan tiga matriks untuk mendefinisikan objek.

Dalam contoh kita memutar fungsi di sekitar sumbu z.

```
>function f(x) := x^3-x+1; ...
>x=-1:0.01:1; t=linspace(0,2pi,8)'; ...
>Z=x; X=cos(t)*f(x); Y=sin(t)*f(x); ...
>pov3d(X,Y,Z,angle=40°,height=20°,axis=0,zoom=4,light=[10,-5,5]);
```

Dalam contoh berikut, kami memplot gelombang teredam. Kami menghasilkan gelombang dengan bahasa matriks Euler.

Kami juga menunjukkan, bagaimana objek tambahan dapat ditambahkan ke adegan pov3d. Untuk pembuatan objek, lihat contoh berikut. Perhatikan bahwa plot3d menskalakan plot, sehingga cocok dengan kubus satuan.

```
>r=linspace(0,1,80); phi=linspace(0,2pi,80)'; ...
>x=r*cos(phi); y=r*sin(phi); z=exp(-5*r)*cos(8*pi*r)/3; ...
>pov3d(x,y,z,zoom=5,axis=0,add=povsphere([0,0,0.5],0.1,povlook(green)), ...
> w=500,h=300);
```

Dengan metode bayangan canggih dari Povray, sangat sedikit titik yang dapat menghasilkan permukaan yang sangat halus. Hanya di perbatasan dan dalam bayang-bayang triknya mungkin menjadi jelas.

Untuk ini, kita perlu menambahkan vektor normal di setiap titik matriks.

```
>Z &= x^2*y^3
```

$$\begin{matrix} 2 & 3 \\ x & y \end{matrix}$$

Persamaan permukaannya adalah $[x,y,Z]$. Kami menghitung dua turunan ke x dan y ini dan mengambil produk silang sebagai normal.

```
>dx &= diff([x,y,Z],x); dy &= diff([x,y,Z],y);
```

Kami mendefinisikan normal sebagai produk silang dari turunan ini, dan mendefinisikan fungsi koordinat.

```
>N &= crossproduct(dx,dy); NX &= N[1]; NY &= N[2]; NZ &= N[3]; N,
```

$$\begin{matrix} 3 & 2 & 2 \\ [-2x^y, -3x^y, 1] \end{matrix}$$

Kami hanya menggunakan 25 poin.

```
>x=-1:0.5:1; y=x';
>pov3d(x,y,Z(x,y),angle=10°, ...
> xv=NX(x,y),yv=NY(x,y),zv=NZ(x,y),<shadow>;
```

Berikut ini adalah simpul Trefoil yang dilakukan oleh A. Busser di Povray. Ada versi yang ditingkatkan dari ini dalam contoh.

Lihat: Contoh\Trefoil Simpul | Simpul trefoil

Untuk tampilan yang bagus dengan tidak terlalu banyak titik, kami menambahkan vektor normal di sini. Kami menggunakan Maxima untuk menghitung normal bagi kami. Pertama, ketiga fungsi koordinat sebagai ekspresi simbolik.

```
>X &= ((4+sin(3*y))+cos(x))*cos(2*y); ...
>Y &= ((4+sin(3*y))+cos(x))*sin(2*y); ...
>Z &= sin(x)+2*cos(3*y);
```

Kemudian kedua vektor turunan ke x dan y.

```
>dx &= diff([X,Y,Z],x); dy &= diff([X,Y,Z],y);
```

Sekarang normal, yang merupakan produk silang dari dua turunan.

```
>dn &= crossproduct(dx,dy);
```

Kami sekarang mengevaluasi semua ini secara numerik.

```
>x:=linspace(-%pi,%pi,40); y:=linspace(-%pi,%pi,100)';
```

Vektor normal adalah evaluasi dari ekspresi simbolik dn[i] untuk i=1,2,3. Sintaks untuk ini adalah &"expression"(parameters). Ini adalah alternatif dari metode pada contoh sebelumnya, di mana kita mendefinisikan ekspresi simbolik NX, NY, NZ terlebih dahulu.

```
>pov3d(X(x,y),Y(x,y),Z(x,y),axis=0,zoom=5,w=450,h=350, ...
> <shadow, look=povlook(gray), ...
> xv=&"dn[1]"(x,y), yv=&"dn[2]"(x,y), zv=&"dn[3]"(x,y));
```

Kami juga dapat menghasilkan grid dalam 3D.

```
>povstart(zoom=4); ...
>x=-1:0.5:1; r=1-(x+1)^2/6; ...
>t=(0°:30°:360°)'; y=r*cos(t); z=r*sin(t); ...
>writeln(povgrid(x,y,z,d=0.02,dballs=0.05)); ...
>povend();
```

Dengan povgrid(), kurva dimungkinkan.

```

>povstart(center=[0,0,1],zoom=3.6); ...
>t=linspace(0,2,1000); r=exp(-t); ...
>x=cos(2*pi*10*t)*r; y=sin(2*pi*10*t)*r; z=t; ...
>writeln(povgrid(x,y,z,povlook(red))); ...
>writeAxis(0,2,axis=3); ...
>povend();

```

Objek Povray

Di atas, kami menggunakan pov3d untuk memplot permukaan. Antarmuka povray di Euler juga dapat menghasilkan objek Povray. Objek-objek ini disimpan sebagai string di Euler, dan perlu ditulis ke file Povray. Kami memulai output dengan povstart().

```
>povstart(zoom=4);
```

Pertama kita mendefinisikan tiga silinder, dan menyimpannya dalam string di Euler.

Fungsi povx() dll. hanya mengembalikan vektor [1,0,0], yang dapat digunakan sebagai gantinya.

```

>c1=povcylinder(-povx,povx,1,povlook(red)); ...
>c2=povcylinder(-povy,povy,1,povlook(green)); ...
>c3=povcylinder(-povz,povz,1,povlook(blue)); ...

```

Pertama kita mendefinisikan tiga silinder, dan menyimpannya dalam string di Euler.

Fungsi povx() dll. hanya mengembalikan vektor [1,0,0], yang dapat digunakan sebagai pengingat.

```
>c1
```

```

cylinder { <-1,0,0>, <1,0,0>, 1
texture { pigment { color rgb <0.564706,0.0627451,0.0627451> } }
finish { ambient 0.2 }
}
```

Seperti yang Anda lihat, kami menambahkan tekstur ke objek dalam tiga warna berbeda.

Itu dilakukan oleh povlook(), yang mengembalikan string dengan kode Povray yang relevan. Kita dapat menggunakan warna Euler default, atau menentukan warna kita sendiri. Kami juga dapat menambahkan transparansi, atau mengubah cahaya sekitar.

```
>povlook(rgb(0.1,0.2,0.3),0.1,0.5)
```

```

texture { pigment { color rgbf <0.101961,0.2,0.301961,0.1> } }
finish { ambient 0.5 }
```

Sekarang kita mendefinisikan objek persimpangan, dan menulis hasilnya ke file.

```
>writeln(povintersection([c1,c2,c3]));
```

Persimpangan tiga silinder sulit untuk divisualisasikan, jika Anda belum pernah melihatnya sebelumnya.

```
>povend;
```

Fungsi berikut menghasilkan fraktal secara rekursif.

Fungsi pertama menunjukkan, bagaimana Euler menangani objek Povray sederhana. Fungsi povbox() mengembalikan string, yang berisi koordinat kotak, tekstur, dan hasil akhir.

```
>function onebox(x,y,z,d) := povbox([x,y,z],[x+d,y+d,z+d],povlook());  
>function fractal(x,y,z,h,n) ...
```

```
if n==1 then writeln(onebox(x,y,z,h));  
else  
    h=h/3;  
    fractal(x,y,z,h,n-1);  
    fractal(x+2*h,y,z,h,n-1);  
    fractal(x,y+2*h,z,h,n-1);  
    fractal(x,y,z+2*h,h,n-1);  
    fractal(x+2*h,y+2*h,z,h,n-1);  
    fractal(x+2*h,y,z+2*h,h,n-1);  
    fractal(x,y+2*h,z+2*h,h,n-1);  
    fractal(x+2*h,y+2*h,z+2*h,h,n-1);  
    fractal(x+h,y+h,z+h,h,n-1);  
endif;  
endfunction
```

```
>povstart(fade=10,<shadow);  
>fractal(-1,-1,-1,2,4);  
>povend();
```

Perbedaan memungkinkan memotong satu objek dari yang lain. Seperti persimpangan, ada bagian dari objek CSG Povray.

```
>povstart(light=[5,-5,5],fade=10);
```

Untuk demonstrasi ini, kami mendefinisikan objek di Povray, alih-alih menggunakan string di Euler. Definisi ditulis ke file segera.

Koordinat kotak -1 berarti [-1,-1,-1].

```
>povdefine("mycube",povbox(-1,1));
```

Kita dapat menggunakan objek ini di povobject(), yang mengembalikan string seperti biasa.

```
>c1=povobject ("mycube", povlook (red)) ;
```

Kami menghasilkan kubus kedua, dan memutar dan menskalakannya sedikit.

```
>c2=povobject ("mycube", povlook (yellow), translate=[1,1,1], ...
>    rotate=xrotate(10°)+yrotate(10°), scale=1.2) ;
```

Kemudian kita ambil selisih kedua benda tersebut.

```
>writeln (povdifference (c1,c2)) ;
```

Sekarang tambahkan tiga sumbu.

```
>writeAxis(-1.2,1.2,axis=1); ...
>writeAxis(-1.2,1.2,axis=2); ...
>writeAxis(-1.2,1.2,axis=4); ...
>povend();
```

Fungsi Implisit

Povray dapat memplot himpunan di mana $f(x,y,z)=0$, seperti parameter implisit di plot3d. Namun, hasilnya terlihat jauh lebih baik.

Sintaks untuk fungsinya sedikit berbeda. Anda tidak dapat menggunakan output dari ekspresi Maxima atau Euler.

```
>povstart (angle=70°,height=50°,zoom=4) ;
```

Buat permukaan implisit. Perhatikan sintaks yang berbeda dalam ekspresi.

```
>writeln(povsurface ("pow(x,2)*y-pow(y,3)-pow(z,2)",povlook(green))); ...
>writeAxes(); ...
>povend();
```

Objek Jala

Dalam contoh ini, kami menunjukkan cara membuat objek mesh, dan menggambarnya dengan informasi tambahan.

Kami ingin memaksimalkan xy di bawah kondisi $x+y=1$ dan menunjukkan sentuhan tangensial dari garis level.

```
>povstart(angle=-10°,center=[0.5,0.5,0.5],zoom=7);
```

Kami tidak dapat menyimpan objek dalam string seperti sebelumnya, karena terlalu besar. Jadi kita mendefinisikan objek dalam file Povray menggunakan declare. Fungsi povtriangle() melakukan ini secara otomatis. Itu dapat menerima vektor normal seperti pov3d().

Berikut ini mendefinisikan objek mesh, dan langsung menulisnya ke dalam file.

```
>x=0:0.02:1; y=x'; z=x*y; vx=-y; vy=-x; vz=1;  
>mesh=povtriangles(x,y,z,"",vx,vy,vz);
```

Now we define two discs, which will be intersected with the surface.

```
>cl=povdisc([0.5,0.5,0],[1,1,0],2); ...  
>ll=povdisc([0,0,1/4],[0,0,1],2);
```

Write the surface minus the two discs.

```
>writeln(povdifference(mesh,povunion([cl,ll]),povlook(green)));
```

Write the two intersections.

```
>writeln(povintersection([mesh,cl],povlook(red))); ...  
>writeln(povintersection([mesh,ll],povlook(gray)));
```

Write a point at the maximum.

```
>writeln(povpoint([1/2,1/2,1/4],povlook(gray),size=2*defaultpointsize));
```

Add axes and finish.

```
>writeAxes(0,1,0,1,0,1,d=0.015); ...  
>povend();
```

Anaglyphs in Povray

To generate an anaglyph for a red/cyan glasses, Povray must run twice from different camera positions. It generates two Povray files and two PNG files, which are loaded with the function loadanaglyph().

Of course, you need red/cyan glasses to view the following examples properly.

The function pov3d() has a simple switch to generate anaglyphs.

```
>pov3d("-exp(-x^2-y^2)/2",r=2,height=45°,>anaglyph, ...
> center=[0,0,0.5],zoom=3.5);
```

If you create a scene with objects, you need to put the generation of the scene into a function, and run it twice with different values for the anaglyph parameter.

```
>function myscene ...
s=povsphere(povc,1);
cl=povcylinder(-povz,povz,0.5);
clk=povobject(cl,rotate=xrotate(90°));
cly=povobject(cl,rotate=yrotate(90°));
c=povbox([-1,-1,0],1);
un=povunion([cl,clk,cly,c]);
obj=povdifference(s,un,povlook(red));
writeln(obj);
writeAxes();
endfunction
```

The function povanaglyph() does all this. The parameters are like in povstart() and povend() combined.

```
>povanaglyph("myscene",zoom=4.5);
```

Defining own Objects

The povray interface of Euler contains a lot of objects. But you are not restricted to these. You can create own objects, which combine other objects, or are completely new objects.

We demonstrate a torus. The Povray command for this is "torus". So we return a string with this command and its parameters. Note that the torus is always centered at the origin.

```
>function povdonat (r1,r2,look "") ...
return "torus {" + r1 + "," + r2 + look + "}";
endfunction
```

Here is our first torus.

```
>t1=povdonat(0.8,0.2)
```

```
torus {0.8,0.2}
```

Let us use this object to create a second torus, translated and rotated.

```
>t2=povobject(t1,rotate=xrotate(90°),translate=[0.8,0,0])
object { torus {0.8,0.2}
rotate 90 *x
translate <0.8,0,0>
}
```

Now we place these objects into a scene. For the look, we use Phong Shading.

```
>povstart(center=[0.4,0,0],angle=0°,zoom=3.8,aspect=1.5); ...
>writeln(povobject(t1,povlook(green,phong=1))); ...
>writeln(povobject(t2,povlook(green,phong=1))); ...
```

```
>povend();
```

calls the Povray program. However, in case of errors, it does not display the error. You should therefore use

```
>povend(<exit);
```

if anything did not work. This will leave the Povray window open.

```
>povend(h=320,w=480);
```

```
Command was not allowed!
exec:
    return _exec(program,param,dir,print,hidden,wait);
povray:
    exec(program,params,defaulthome);
Try "trace errors" to inspect local variables after errors.
povend:
    povray(file,w,h,aspect,exit);
```

Here is a more elaborate example. We solve

$$Ax \leq b, \quad x \geq 0, \quad c \cdot x \rightarrow \text{Max.}$$

and show the feasible points and the optimum in a 3D plot.

```
>A=[10,8,4;5,6,8;6,3,2;9,5,6];
>b=[10,10,10,10]';
>c=[1,1,1];
```

First, let us check, if this example has a solution at all.

```
>x=simplex(A,b,c,>max,>check)'
```

```
[0, 1, 0.5]
```

Yes, it has.

Next we define two objects. The first is the plane

$$a \cdot x \leq b$$

```
>function oneplane (a,b,look="") ...
```

```
    return povplane(a,b,look)
endfunction
```

Then we define the intersection of all half spaces and a cube.

```
>function adm (A, b, r, look="") ...
```

```
ol=[];
loop 1 to rows(A); ol=ol|oneplane(A[#],b[#]); end;
ol=ol|povbox([0,0,0],[r,r,r]);
return povintersection(ol,look);
endfunction
```

We can now plot the scene.

```
>povstart(angle=120°,center=[0.5,0.5,0.5],zoom=3.5); ...
>writeln(adm(A,b,2,povlook(green,0.4))); ...
>writeAxes(0,1.3,0,1.6,0,1.5); ...
```

The following is a circle around the optimum.

```
>writeln(povintersection([povsphere(x,0.5),povplane(c,c.x')]), ...
>  povlook(red,0.9));
```

And an error in the direction of the optimum.

```
>writeln(povarrow(x,c*0.5,povlook(red)));
```

We add text to the screen. Text is just a 3D object. We need to place and turn it according to our view.

```
>writeln(povtext("Linear Problem", [0,0.2,1.3], size=0.05, rotate=125°)); ...
>povend();
```

More Examples

You can find some more examples for Povray in Euler in the following files.

See: Examples/Dandelin Spheres

See: Examples/Donat Math

See: Examples/Trefoil Knot

See: Examples/Optimization by Affine Scaling

BAB 6

PENGGUNAAN SOFTWARE EMT UNTUK KALKULUS

Kalkulus dengan EMT

Materi Kalkulus mencakup di antaranya:

- Fungsi (fungsi aljabar, trigonometri, eksponensial, logaritma, komposisi fungsi)
- Limit Fungsi,
- Turunan Fungsi,
- Integral Tak Tentu,
- Integral Tentu dan Aplikasinya,
- Barisan dan Deret (kekonvergenan barisan dan deret).

EMT (bersama Maxima) dapat digunakan untuk melakukan semua perhitungan di dalam kalkulus, baik secara numerik maupun analitik (eksak).

Mendefinisikan Fungsi

Terdapat beberapa cara mendefinisikan fungsi pada EMT, yakni:

- Menggunakan format `nama_fungsi := rumus fungsi` (untuk fungsi numerik),
- Menggunakan format `nama_fungsi &= rumus fungsi` (untuk fungsi simbolik, namun dapat dihitung secara numerik),
- Menggunakan format `nama_fungsi &&= rumus fungsi` (untuk fungsi simbolik murni, tidak dapat dihitung langsung),
- Fungsi sebagai program EMT.

Setiap format harus diawali dengan perintah `function` (bukan sebagai ekspresi).

Berikut adalah beberapa contoh cara mendefinisikan fungsi:

$$f(x) = 2x^2 + e^{\sin(x)}.$$

```
> function f(x) := 2*x^2+exp(sin(x)) // fungsi numerik
> f(0), f(1), f(pi)
```

```
1  
4.31977682472  
20.7392088022
```

```
> f(a) // tidak dapat dihitung nilainya
```

```
Variable or function a not found.  
Error in:  
f(a) // tidak dapat dihitung nilainya ...  
^
```

Silahkan Anda plot kurva fungsi di atas!

```
> plot2d("2*x^2+exp(sin(x))", -5, 5); plot2d(0,1,>points, style="ow", >add) :
```

Berikutnya kita definisikan fungsi:

$$g(x) = \frac{\sqrt{x^2 - 3x}}{x + 1}.$$

```
> function g(x) := sqrt(x^2-3*x)/(x+1)  
> g(3)
```

```
0
```

```
> g(0)
```

```
0
```

```
> g(1) // kompleks, tidak dapat dihitung oleh fungsi numerik
```

```
Floating point error!  
Error in sqrt  
Try "trace errors" to inspect local variables after errors.  
g:  
  useglobal; return sqrt(x^2-3*x)/(x+1)  
Error in:  
g(1) // kompleks, tidak dapat dihitung oleh fungsi numerik ...  
^
```

Silahkan Anda plot kurva fungsi di atas!

```
> plot2d("sqrt(x^2-3*x)/(x+1)", -5, 5); plot2d(0,0,>points, style="ow", >add) :  
> f(g(5)) // komposisi fungsi
```

```
2.20920171961
```

```
> g(f(5))
```

```
0.950898070639
```

```
> function h(x) := f(g(x)) // definisi komposisi fungsi  
> h(5) // sama dengan f(g(5))
```

```
2.20920171961
```

Silakan Anda plot kurva fungsi komposisi fungsi f dan g:

$$h(x) = f(g(x))$$

dan

$$u(x) = g(f(x))$$

bersama-sama kurva fungsi f dan g dalam satu bidang koordinat.

```
> plot2d("f(g(5))",-5,5); plot2d("g(f(5))",-5,5); plot2d(0,0,>points,style="ow",>add):  
> f(0:10) // nilai-nilai f(1), f(2), ..., f(10)
```

```
[2, 5.43656, 14.7781, 40.1711, 109.196, 296.826, 806.858,  
2193.27, 5961.92, 16206.2, 44052.9]
```

```
> fmap(0:10) // sama dengan f(0:10), berlaku untuk semua fungsi
```

```
[2, 5.43656, 14.7781, 40.1711, 109.196, 296.826, 806.858,  
2193.27, 5961.92, 16206.2, 44052.9]
```

```
> gmap(200:210)
```

```
[0.987534, 0.987596, 0.987657, 0.987718, 0.987778, 0.987837,  
0.987896, 0.987954, 0.988012, 0.988069, 0.988126]
```

Misalkan kita akan mendefinisikan fungsi

$$f(x) = \begin{cases} x^3 & x > 0 \\ x^2 & x \leq 0. \end{cases}$$

Fungsi tersebut tidak dapat didefinisikan sebagai fungsi numerik secara "inline" menggunakan format `:=`, melainkan didefinisikan sebagai program. Perhatikan, kata "map" digunakan agar fungsi dapat menerima vektor sebagai input, dan hasilnya berupa vektor. Jika tanpa kata "map" fungsinya hanya dapat menerima input satu nilai.

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

```
if x > 0 then return x^3
else return x^2
endif;
endfunction
```

```
> f(1)
```

1

```
> f(-2)
```

4

```
> f(-5:5)
```

[25, 16, 9, 4, 1, 0, 1, 8, 27, 64, 125]

```
> aspect(1.5); plot2d("f(x)", -5, 5):
> function f(x) &= 2*x // fungsi simbolik
```

$$\begin{matrix} x \\ 2x \end{matrix}$$

```
> $f(a) // nilai fungsi secara simbolik
> f(E) // nilai fungsi berupa bilangan desimal
```

30.308524483

```
> function g(x) &= 3*x + 1
```

$$3x + 1$$

```
> function h(x) &= f(g(x)) // komposisi fungsi
```

$$\begin{matrix} 3x + 1 \\ 2x \end{matrix}$$

```
> plot2d("h(x)", -1, 1):
```

Latihan

Bukalah buku Kalkulus. Cari dan pilih beberapa (paling sedikit 5 fungsi berbeda tipe/bentuk/jenis) fungsi dari buku tersebut, kemudian definisikan fungsi-fungsi tersebut dan komposisinya di EMT pada baris-baris perintah berikut (jika perlu tambahkan lagi). Untuk setiap fungsi, hitung beberapa nilainya, baik untuk satu nilai maupun vektor. Gambar grafik fungsi-fungsi tersebut dan komposisi-komposisi 2 fungsi.

Juga, carilah fungsi beberapa (dua) variabel. Lakukan hal sama seperti di atas.

1. Hitung nilai $f(x)$ jika diketahui $x = 2$ dan $x = 6$

$$f(x) = x^2 + 4x - 8$$

Sertakan gambar dari grafik fungsi $f(x)$!

```
> function f(x) := x^2 + 4*x - 8  
> f(2), f(6)
```

4
52

```
> plot2d("x^2 + 4*x - 8", -2, 2):
```

2. Hitunglah nilai $f(-3)$, $f(8)$, dan $f(0)$ untuk fungsi berikut!

$$f(x) = \frac{x^3 + 12}{x - 1}$$

```
> function f(x) := (x^3+12) / (x-1)  
> f(-3), f(8), f(0)
```

3.75
74.8571428571
-12

```
> plot2d("(x^3+12) / (x-1)", -2, 2):  
> function f(x) := sqrt(x) + 4*x  
> f(81)
```

```
> plot2d("sqrt(x)+4*x", -5, 5):
```

4. Perhatikan fungsi di bawah ini!

$$f(x) = x^4 + 2x^2 - 5x + 12$$

Tentukan nilai $f(9) + f(3)$!

```
> function f(x) := x^4+2*x^2-5*x+12  
> f(9), f(3), f(9) + f(3)
```

6690
96
6786

```
> plot2d("x^4+2*x^2-5*x+12", -2, 2):
```

5. Diketahui suatu fungsi

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

$$g(x) = x^2 - 8$$

Tentukan nilai $fog(-4)$ dan $gof(9)$!

```
> function f(x) := x^5 + 6  
> function g(x) := x^2 - 8  
> f(g(-4)), g(f(9))
```

32774
3487493017

```
> plot2d("x^5 + 6", -2, 2), plot2d("x^2-8", -2, 2):
```

Menghitung Limit

Perhitungan limit pada EMT dapat dilakukan dengan menggunakan fungsi Maxima, yakni "limit". Fungsi "limit" dapat digunakan untuk menghitung limit fungsi dalam bentuk ekspresi maupun fungsi yang sudah didefinisikan sebelumnya. Nilai limit dapat dihitung pada sebarang nilai atau pada tak hingga (-inf, minf, dan inf). Limit kiri dan limit kanan juga dapat dihitung, dengan cara memberi opsi "plus" atau "minus". Hasil limit dapat berupa nilai, "und" (tak definisi), "ind" (tak tentu namun terbatas), "infinity" (kompleks tak hingga). Perhatikan beberapa contoh berikut. Perhatikan cara menampilkan perhitungan secara lengkap, tidak hanya menampilkan hasilnya saja.

```
> $showev('limit(sqrt(x^2-3*x)/(x+1),x,inf))
> $limit((x^3-13*x^2+51*x-63)/(x^3-4*x^2-3*x+18),x,3)
```

maxima: $\lim_{x \rightarrow \infty} \frac{\sqrt{x^2 - 3x}}{x + 1}$ = $\lim_{x \rightarrow \infty} \frac{(x^3 - 13x^2 + 51x - 63)}{(x^3 - 4x^2 - 3x + 18)}$

Fungsi tersebut diskontinu di titik $x = 3$. Berikut adalah grafik fungsinya.

```
> plot2d("2*x*sin(x)/(1-cos(x))",-pi,pi); plot2d(0,4,>points,style="ow",>add):
> $limit(cot(7*h)/cot(5*h),h,0)
```

maxima: $\text{showev}(\lim_{h \rightarrow 0} \cot(7h)/\cot(5h), h, 0)$

Fungsi tersebut juga diskontinu (karena tidak terdefinisi) di $x=0$. Berikut adalah grafiknya.

```
> plot2d("cot(7*x)/cot(5*x)",-0.001,0.001); plot2d(0,5/7,>points,style="ow",>add):
> $showev('limit(((x/8)^(1/3)-1)/(x-8),x,8))
> plot2d("((x/8)^(1/3)-1)/(x-8)",-0.001,0.001); plot2d(8,1/24,>points,style="ow",>add):
> $showev('limit(1/(2*x-1),x,0))
> plot2d("1/(2*x-1)",-5,5); plot2d(0,-1,>points,style="ow",>add):
> $showev('limit((x^2-3*x-10)/(x-5),x,5))
> plot2d("(x^2-3*x-10)/(x-5)",-5,5); plot2d(5,7,>points,style="ow",>add):
> $showev('limit(sqrt(x^2+x)-x,x,inf))
> plot2d("sqrt(x^2+x)-x",-1,1); plot2d(0,1/2,>points,style="ow",>add):
> $showev('limit(abs(x-1)/(x-1),x,1,minus))
> plot2d("abs(x-1)/(x-1)",-5,5); plot2d(1,-1,>points,style="ow",>add):
> $showev('limit(sin(x)/x,x,0))
> plot2d("sin(x)/x",-pi,pi); plot2d(0,1,>points,style="ow",>add):
> $showev('limit(sin(x^3)/x,x,0))
> plot2d("sin(x^3)/x",-pi,pi); plot2d(0,0,>points,style="ow",>add):
> $showev('limit(log(x), x, minf))
> $showev('limit((-2)^x,x, inf))
> $showev('limit(t-sqrt(2-t),t,2,minus))
> $showev('limit(t-sqrt(2-t),t,2,plus))
> $showev('limit(t-sqrt(2-t),t,5,plus)) // Perhatikan hasilnya
> plot2d("x-sqrt(2-x)",0,2):
> $showev('limit((x^2-9)/(2*x^2-5*x-3),x,3))
> plot2d("(x^2-9)/(2*x^2-5*x-3)",0,2):
> $&showev('limit(2*x*sin(x)/(1-cos(x)),x,0))
```

Fungsi tersebut diskontinu di titik $x = 3$. Berikut adalah grafik fungsinya.

```
> plot2d("2*x*sin(x)/(1-cos(x))",-pi,pi); plot2d(0,4,>points,style="ow",>add):
> $&showev('limit(cot(7*h)/cot(5*h),h,0))
```

Fungsi tersebut juga diskontinu (karena tidak terdefinisi) di $x = 0$. Berikut adalah grafiknya.

```
> plot2d("cot(7*x)/cot(5*x)",-0.001,0.001); plot2d(0,5/7,>points,style="ow",>add) :  
> $showev('limit(((x/8)^(1/3))-1)/(x-8),x,8))  
> plot2d("((x/8)^(1/3))-1)/(x-8)",0,9); plot2d(8,1/24,>points,style="ow",>add) :  
> $showev('limit(1/(2*x-1),x,0))  
> plot2d("1/(2*x-1)",-1,1); plot2d(0,-1,>points,style="ow",>add) :  
> $showev('limit((x^2-3*x-10)/(x-5),x,5))  
> plot2d("(x^2-3*x-10)/(x-5)",-1,1); plot2d(5,7,>points,style="ow",>add) :  
> $showev('limit(sqrt(x^2+x)-x,x,inf))  
> plot2d("sqrt(x^2+x)-x",-1,1); plot2d(0,1/2,>points,style="ow",>add) :  
> $showev('limit(abs(x-1)/(x-1),x,1,minus))  
> plot2d("abs(x-1)/(x-1)",-5,5); plot2d(1,-1,>points,style="ow",>add) :  
> $showev('limit(sin(x)/x,x,0))  
> plot2d("sin(x)/x",-pi,pi); plot2d(0,1,>points,style="ow",>add) :  
> $showev('limit(sin(x^3)/x,x,0))  
> plot2d("sin(x^3)/x",-pi,pi); plot2d(0,0,>points,style="ow",>add) :  
> $showev('limit(log(x), x, minf))  
> plot2d("log(x)",-pi,pi); plot2d(0,0,>points,style="ow",>add) :  
> $showev('limit((-2)^x,x, inf))  
> $showev('limit(t-sqrt(2-t),t,2,minus))  
> $showev('limit(t-sqrt(2-t),t,5,plus)) // Perhatikan hasilnya  
> plot2d("x-sqrt(2-x)",0,2):  
> $showev('limit((x^2-9)/(2*x^2-5*x-3),x,3))  
> $showev('limit((1-cos(x))/x,x,0))  
> plot2d("(1-cos(x))/x",-pi,pi); plot2d(0,0,>points,style="ow",>add) :  
> $showev('limit((x^2+abs(x))/(x^2-abs(x)),x,0))  
> plot2d("(x^2+abs(x))/(x^2-abs(x))",-5,5); plot2d(0,-1,>points,style="ow",>add) :  
> $showev('limit((1+1/x)^x,x,inf))  
> plot2d("(1+1/x)^x",0,1000):  
> $showev('limit((1+k/x)^x,x,inf))  
> $showev('limit((1+x)^(1/x),x,0))  
> plot2d("(1+x)^(1/x)",-5,5); plot2d(0,2.7,>points,style="ow",>add) :  
> $showev('limit((x/(x+k))^x,x,inf))  
> $showev('limit((E^x-E^2)/(x-2),x,2))  
> plot2d("(E^x-E^2)/(x-2)",-5,5); plot2d(0,2.98,>points,style="ow",>add) :  
> $showev('limit(sin(1/x),x,0))  
> $showev('limit(sin(1/x),x,inf))  
> plot2d("sin(1/x)",-5,5):
```

Latihan

Bukalah buku Kalkulus. Cari dan pilih beberapa (paling sedikit 5 fungsi berbeda tipe/bentuk/jenis) fungsi dari buku tersebut, kemudian definisikan di EMT pada baris-baris perintah berikut (jika perlu tambahkan lagi). Untuk setiap fungsi, hitung nilai limit fungsi tersebut di beberapa nilai dan di tak hingga. Gambar grafik fungsi tersebut untuk mengkonfirmasi nilai-nilai limit tersebut.

1. Hitunglah nilai limit di bawah ini!

$$\lim_{x \rightarrow 5} (x - 4)$$

```
> $showev('limit((x-4),x,5))
```

2. Hitunglah nilai dari limit berikut!

$$\lim_{x \rightarrow 3} (x^3 + 2)$$

```
> $showev('limit((x^3+2),x,3))
```

3. Hitunglah nilai limit berikut dan gambarlah grafiknya.

$$\lim_{t \rightarrow 1} \frac{t^4 - 2}{\sin(t - 5)}$$

```
> $showev('limit((t^4-2)/sin(t-5),t,1))  
> (plot2d("(x^4-2)/sin(x-5)", -5, 5)):
```

4. Tentukan nilai limit berikut.

$$\lim_{x \rightarrow -5} \frac{\sqrt{3+6x}}{(2x-5)^4}$$

```
> $showev('limit((sqrt(3+6*x))/( (2*x-5)^4),x,-5))
```

5. Tentukan nilai limit berikut.

$$\lim_{x \rightarrow 0} \frac{(x - \cos(x))^3}{x^6}$$

```
> $showev('limit(((x-cos(x))^3)/(x^6),x,0))  
> (plot2d("((x-cos(x))^3)/(x^6)", -5, 5)):
```

Turunan Fungsi

Definisi turunan:

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

Berikut adalah contoh-contoh menentukan turunan fungsi dengan menggunakan definisi turunan (limit).

```

> $showev('limit(((x+h)^2-x^2)/h,h,0)) // turunan x^2
> p &= expand((x+h)^2-x^2)|simplify; $p // pembilang dijabarkan dan disederhanakan
> q &=ratsimp(p/h); $q // ekspresi yang akan dihitung limitnya disederhanakan
> $limit(q,h,0) // nilai limit sebagai turunan
> $showev('limit(((x+h)^n-x^n)/h,h,0)) // turunan x^n

```

Mengapa hasilnya seperti itu? Tuliskan atau tunjukkan bahwa hasil limit tersebut benar, sehingga benar turunan fungsinya benar. Tulis penjelasan Anda di komentar ini.

Sebagai petunjuk, gunakan sifat-sifat logaritma dan hasil limit pada bagian sebelumnya di atas.

PEMBUKTIAN

$$f'(x) = \lim_{h \rightarrow 0} \frac{(x+h)^n - x^n}{h}$$

$$\begin{aligned} &= \lim_{h \rightarrow 0} \frac{\frac{d}{dh}((x+h)^n - x^n)}{\frac{d}{dh}(h)} \\ &= \lim_{h \rightarrow 0} \frac{n(x+h)^{n-1}}{1} \\ &= \lim_{h \rightarrow 0} n(x+h)^{n-1} \\ &= nx^{n-1} \end{aligned}$$

TERBUKTI

Jadi, terbukti benar bahwa

$$f'(x) = \lim_{h \rightarrow 0} \frac{(x+h)^n - x^n}{h} = nx^{n-1}$$

```

> $showev('limit((sin(x+h)-sin(x))/h,h,0)) // turunan sin(x)

```

Mengapa hasilnya seperti itu? Tuliskan atau tunjukkan bahwa hasil limit tersebut benar, sehingga benar turunan fungsinya benar. Tulis penjelasan Anda di komentar ini.

Sebagai petunjuk, ekspansikan $\sin(x+h)$ dengan menggunakan rumus jumlah dua sudut.

PEMBUKTIAN

$$f'(x) = \lim_{h \rightarrow 0} \frac{\sin(x+h) - \sin x}{h}$$

$$\begin{aligned} &= \lim_{h \rightarrow 0} \frac{\frac{d}{dh}(\sin(x+h) - \sin x)}{\frac{d}{dh}(h)} \\ &= \lim_{h \rightarrow 0} \frac{\cos(x+h)}{1} \\ &= \lim_{h \rightarrow 0} \cos(x+h) \end{aligned}$$

Karena $\cos(x+h)$ kontinu di $h = 0$, maka:

$$= \cos x$$

TERBUKTI

Jadi, terbukti benar bahwa

$$f'(x) = \lim_{h \rightarrow 0} \frac{\sin(x+h) - \sin x}{h} = \cos x$$

```
> $showev('limit((log(x+h)-log(x))/h,h,0)) // turunan log(x)
```

Mengapa hasilnya seperti itu? Tuliskan atau tunjukkan bahwa hasil limit tersebut benar, sehingga benar turunan fungsinya benar. Tulis penjelasan Anda di komentar ini.

Sebagai petunjuk, gunakan sifat-sifat logaritma dan hasil limit pada bagian sebelumnya di atas.

PEMBUKTIAN

$$f'(x) = \lim_{h \rightarrow 0} \frac{\log(x+h) - \log x}{h}$$

$$\begin{aligned} &= \lim_{h \rightarrow 0} \frac{\frac{d}{dh}(\log(x+h) - \log x)}{\frac{d}{dh}(h)} \\ &= \lim_{h \rightarrow 0} \frac{\frac{1}{x+h}}{1} \\ &= \lim_{h \rightarrow 0} \frac{1}{x+h} \\ &= \frac{1}{x} \end{aligned}$$

TERBUKTI

Jadi, terbukti benar bahwa

$$f'(x) = \lim_{h \rightarrow 0} \frac{\log(x+h) - \log x}{h} = \frac{1}{x}$$

```
> $showev('limit((1/(x+h)-1/x)/h,h,0)) // turunan 1/x
> $showev('limit((E^(x+h)-E^x)/h,h,0)) // turunan f(x)=e^x
```

```
Answering "Is x an integer?" with "integer"
Maxima is asking
Acceptable answers are: yes, y, Y, no, n, N, unknown, uk
Is x an integer?
```

```
Use assume!
Error in:
$showev('limit((E^(x+h)-E^x)/h,h,0)) // turunan f(x)=e^x ...
^
```

Maxima bermasalah dengan limit:

$$\lim_{h \rightarrow 0} \frac{e^{x+h} - e^x}{h}.$$

Oleh karena itu diperlukan trik khusus agar hasilnya benar.

```
> $showev('limit((E^h-1)/h,h,0))
> $showev('factor(E^(x+h)-E^x))
> $showev('limit(factor((E^(x+h)-E^x)/h),h,0)) // turunan f(x)=e^x
> function f(x) &= x^x
```

$\frac{x}{x}$

```
> $showev('limit(f(x),x,0))
> function y := x^x
> plot2d("x^x",-5,5):
> $showev('limit((f(x+h)-f(x))/h,h,0)) // turunan f(x)=x^x
```

Di sini Maxima juga bermasalah terkait limit:

$$\lim_{h \rightarrow 0} \frac{(x+h)^{x+h} - x^x}{h}.$$

Dalam hal ini diperlukan asumsi nilai x.

```
> &assume(x>0); $showev('limit((f(x+h)-f(x))/h,h,0)) // turunan f(x)=x^x
```

Mengapa hasilnya seperti itu? Tuliskan atau tunjukkan bahwa hasil limit tersebut benar, sehingga benar turunan fungsinya benar. Tulis penjelasan Anda di komentar ini.

```
> &forget(x>0) // jangan lupa, lupakan asumsi untuk kembali ke semula
```

[x > 0]

```
> &forget(x<0)
```

[x < 0]

```
> &facts()
```

[]

```
> $showev('limit((asin(x+h)-asin(x))/h,h,0)) // turunan arcsin(x)
```

Mengapa hasilnya seperti itu? Tuliskan atau tunjukkan bahwa hasil limit tersebut benar, sehingga benar turunan fungsinya benar. Tulis penjelasan Anda di komentar ini.

```
> $showev('limit((tan(x+h)-tan(x))/h,h,0)) // turunan tan(x)
```

Mengapa hasilnya seperti itu? Tuliskan atau tunjukkan bahwa hasil limit tersebut benar, sehingga benar turunan fungsinya benar. Tulis penjelasan Anda di komentar ini.

```
> function f(x) &= sinh(x) // definisikan f(x)=sinh(x)
```

sinh(x)

```
> function df(x) &= limit((f(x+h)-f(x))/h,h,0); $df(x) // df(x) = f'(x)
```

Hasilnya adalah $\cosh(x)$, karena

$$\frac{e^x + e^{-x}}{2} = \cosh(x).$$

```
> plot2d(["f(x)", "df(x)"], -pi, pi, color=[blue, red]):  
> function f(x) &= sin(3*x^5+7)^2
```

$$\sin^2(3x^5 + 7)$$

```
> diff(f,3), diffc(f,3)
```

```
1198.32948904  
1198.72863721
```

Apakah perbedaan `diff` dan `diffc`?

```
> $showev('diff(f(x),x))  
> $% with x=3  
> $float(%)  
> plot2d(f,0,3.1):  
> function f(x) &= 5*cos(2*x)-2*x*sin(2*x) // mendefinisikan fungsi f
```

$$5 \cos(2x) - 2x \sin(2x)$$

```
> function df(x) &= diff(f(x),x) // fd(x) = f'(x)
```

$$- 12 \sin(2x) - 4x \cos(2x)$$

```
> $'f(1)=f(1), $float(f(1)), $'f(2)=f(2), $float(f(2)) // nilai f(1) dan f(2)  
> xp=solve("df(x)",1,2,0) // solusi f'(x)=0 pada interval [1, 2]
```

```
1.35822987384
```

```
> df(xp), f(xp) // cek bahwa f'(xp)=0 dan nilai ekstrim di titik tersebut
```

0
-5.67530133759

```
> plot2d(["f(x)", "df(x")], 0, 2*pi, color=[blue, red]): //grafik fungsi dan turunannya
```

Perhatikan titik-titik "puncak" grafik $y=f(x)$ dan nilai turunan pada saat grafik fungsinya mencapai titik "puncak" tersebut.

Latihan

Bukalah buku Kalkulus. Cari dan pilih beberapa (paling sedikit 5 fungsi berbeda tipe/bentuk/jenis) fungsi dari buku tersebut, kemudian definisikan di EMT pada baris-baris perintah berikut (jika perlu tambahkan lagi). Untuk setiap fungsi, tentukan turunannya dengan menggunakan definisi turunan (limit), menggunakan perintah diff, dan secara manual (langkah demi langkah yang dihitung dengan Maxima) seperti contoh-contoh di atas. Gambar grafik fungsi asli dan fungsi turunannya pada sumbu koordinat yang sama.

Contoh 1

```
> $showev('limit(((x+h)^n-x^n)/h,h,0)) // turunan x^n
```

Pembuktian

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

Untuk

$$f(x) = x^n, f(x+h) = (x+h)^n$$

$$\frac{d}{dx} x^n = \lim_{h \rightarrow 0} \frac{(x+h)^n - x^n}{h}$$

Dengan

$$(a+b)^n = \sum_{k=0}^n a^k b^{n-k}$$

maka

$$\begin{aligned} &= \lim_{h \rightarrow 0} \frac{(x^n + \frac{n}{1!}x^{n-1}h + \frac{n(n-1)}{2!}x^{n-2}h^2 + \frac{n(n-1)(n-2)}{3!}x^{n-3}h^3 + \dots) - x^n}{h} \\ &= \lim_{h \rightarrow 0} \frac{n.x^{n-1}h + \frac{n(n-1)}{2!}x^{n-2}h^2 + \frac{n(n-1)(n-2)}{3!}x^{n-3}h^3 + \dots}{h} \\ &= \lim_{h \rightarrow 0} n.x^{n-1} + \frac{n(n-1)}{2!}.x^{n-2}h + \frac{n(n-1)(n-2)}{3!}.x^{n-3}h^2 + \dots \\ &= n.x^{n-1} + 0 + 0 + \dots + 0 \\ &= n.x^{n-1} \end{aligned}$$

Jadi, terbukti benar bahwa

$$f'(x^n) = n \cdot x^{n-1}$$

```
# Visualisasi Grafik
```

```
> plot2d(["x^2", "2*x^(2-1)"], color=[blue, red]): //grafik fungsi dan turunannya
```

Contoh 2

```
> function f(x) &= sin(x); $f(x) // mendefinisikan fungsi f
> function df(x) &= diff(f(x), x); $df(x) // df(x) = f'(x)
```

```
# Pembuktian
```

$$f'(x) = \lim_{h \rightarrow 0} \frac{\sin(x+h) - \sin(x)}{h}$$

$$\sin(a+b) = \sin(a)\cos(b) + \cos(a)\sin(b)$$

$$= \lim_{h \rightarrow 0} \frac{\sin(x)\cos(h) + \cos(x)\sin(h) - \sin(x)}{h}$$

$$= \lim_{h \rightarrow 0} \sin(x) \cdot \frac{\cos(h) - 1}{h} + \lim_{h \rightarrow 0} \cos(x) \cdot \frac{\sin(h)}{h}$$

$$= \sin(x) \cdot 0 + \cos(x) \cdot 1$$

$$= \cos(x)$$

Jadi, terbukti benar bahwa

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

```
# Visualisasi Grafik
```

```
> plot2d(["f(x)", "df(x)"], color=[blue, red]): //grafik fungsi dan turunannya
```

Contoh 3

$$f(x) = \log(x)$$

```
> $showev('limit((log(x+h)-log(x))/h,h,0)) // turunan log(x)
```

Pembuktian

$$\begin{aligned} f'(x) &= \lim_{h \rightarrow 0} \frac{\log(x+h) - \log x}{h} \\ &= \lim_{h \rightarrow 0} \frac{\frac{d}{dh}(\log(x+h) - \log x)}{\frac{d}{dh}(h)} \\ &= \lim_{h \rightarrow 0} \frac{\frac{1}{x+h}}{1} \\ &= \lim_{h \rightarrow 0} \frac{1}{x+h} \\ &= \frac{1}{x} \end{aligned}$$

Jadi, terbukti benar bahwa

$$f'(x) = \lim_{h \rightarrow 0} \frac{\log(x+h) - \log x}{h} = \frac{1}{x}$$

Visualisasi Grafik

```
> plot2d(["log(x)", "1/x"], color=[blue, red]): //grafik fungsi dan turunannya
```

Contoh 4

```
> function f(x) &= x^2+1; $f(x)
> function g(x) &= x+5; $g(x)
> $showev('diff(f(g(x)),x))
```

Visualisasi Grafik

```
> plot2d(["f(x)", "g(x)", "2*(x+5)"], color=[blue, red, green]): //grafik fungsi dan turunannya
```

Contoh 5

$$f(x) = \frac{2x-5}{x+2}$$

```
> $showev('limit(((2*(x+h)-5)/((x+h)+2) - (2*x-5)/(x+2))/h,h,0)) // turunan 2x^2+5
> p &= expand((2*(x+h)-5)/((x+h)+2) - (2*x-5)/(x+2))|simplify; $p // pembilang dij
> q &=ratsimp(p/h); $q // ekspresi yang akan dihitung limitnya disederhanakan
> $limit(q,h,0) // nilai limit sebagai turunan
```

```
# Visualisasi Grafik
```

```
> plot2d(["f(x)","df(x)",color=[blue,red]): //grafik fungsi dan turunannya
```

Integral

EMT dapat digunakan untuk menghitung integral, baik integral tak tentu maupun integral tentu. Untuk integral tak tentu (simbolik) sudah tentu EMT menggunakan Maxima, sedangkan untuk perhitungan integral tentu EMT sudah menyediakan beberapa fungsi yang mengimplementasikan algoritma kuadratur (perhitungan integral tentu menggunakan metode numerik).

Pada notebook ini akan ditunjukkan perhitungan integral tentu dengan menggunakan Teorema Dasar Kalkulus:

$$\int_a^b f(x) dx = F(b) - F(a), \quad \text{dengan } F'(x) = f(x).$$

Fungsi untuk menentukan integral adalah integrate. Fungsi ini dapat digunakan untuk menentukan, baik integral tentu maupun tak tentu (jika fungsinya memiliki antiderivatif). Untuk perhitungan integral tentu fungsi integrate menggunakan metode numerik (kecuali fungsinya tidak integrabel, kita tidak akan menggunakan metode ini).

```
> $showev('integrate(x^n,x)')
```

Answering "Is n equal to -1?" with "no"

```
> $showev('integrate(1/(1+x),x)')
> $showev('integrate(1/(1+x^2),x)')
> $showev('integrate(1/sqrt(1-x^2),x)')
> $showev('integrate(sin(x),x,0,pi)')
> plot2d("sin(x)",0,2*pi):
> $showev('integrate(sin(x),x,a,b)')
> $showev('integrate(x^n,x,a,b)')
```

Answering "Is n positive, negative or zero?" with "positive"

```
> $showev('integrate(x^2*sqrt(2*x+1),x)')
> $showev('integrate(x^2*sqrt(2*x+1),x,0,2)')
> $ratsimp(%)
> $showev('integrate((sin(sqrt(x))+a)*E^sqrt(x))/sqrt(x),x,0,pi^2)')
> $factor(%)
> function map f(x) &= E^(-x^2)
```

$$\frac{2}{\sqrt{\pi}} \int_{-\infty}^x e^{-t^2} dt$$

```
> $showev('integrate(f(x),x))
```

Fungsi f tidak memiliki antiturunan, integralnya masih memuat integral lain.

$$erf(x) = \int \frac{e^{-x^2}}{\sqrt{\pi}} dx.$$

Kita tidak dapat menggunakan teorema Dasar kalkulus untuk menghitung integral tentu fungsi tersebut jika semua batasnya berhingga. Dalam hal ini dapat digunakan metode numerik (rumus kuadratur).

Misalkan kita akan menghitung:

maxima: 'integrate(f(x),x,0,pi)

```
> x=0:0.1:pi-0.1; plot2d(x,f(x+0.1),>bar); plot2d("f(x)",0,pi,>add):
```

Integral tentu

maxima: 'integrate(f(x),x,0,pi)

dapat dihampiri dengan jumlah luas persegi-persegi panjang di bawah kurva $y=f(x)$ tersebut. Langkah-langkahnya adalah sebagai berikut.

```
> t &= makelist(a,a,0,pi-0.1,0.1); // t sebagai list untuk menyimpan nilai-nilai x
> fx &= makelist(f(t[i]+0.1),i,1,length(t)); // simpan nilai-nilai f(x)
> // jangan menggunakan x sebagai list, kecuali Anda pakar Maxima!
```

Hasilnya adalah:

maxima: 'integrate(f(x),x,0,pi) = 0.1*sum(fx[i],i,1,length(fx))

Jumlah tersebut diperoleh dari hasil kali lebar sub-subinterval (=0.1) dan jumlah nilai-nilai $f(x)$ untuk $x = 0.1, 0.2, 0.3, \dots, 3.2$.

```
> 0.1*sum(f(x+0.1)) // cek langsung dengan perhitungan numerik EMT
```

0.836219610253

Untuk mendapatkan nilai integral tentu yang mendekati nilai sebenarnya, lebar sub-intervalnya dapat diperkecil lagi, sehingga daerah di bawah kurva tertutup semuanya, misalnya dapat digunakan lebar subinterval 0.001. (Silakan dicoba!)

Meskipun Maxima tidak dapat menghitung integral tentu fungsi tersebut untuk batas-batas yang berhingga, namun integral tersebut dapat dihitung secara eksak jika batas-batasnya tak hingga. Ini adalah salah satu keajaiban di dalam matematika, yang terbatas tidak dapat dihitung secara eksak, namun yang tak hingga malah dapat dihitung secara eksak.

```
> $showev('integrate(f(x),x,0,inf))
```

Tunjukkan kebenaran hasil di atas!

Berikut adalah contoh lain fungsi yang tidak memiliki antiderivatif, sehingga integral tentunya hanya dapat dihitung dengan metode numerik.

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

$$\frac{x}{x}$$

```
> $showev('integrate(f(x),x,0,1))
> x=0:0.1:1-0.01; plot2d(x,f(x+0.01),>bar); plot2d("f(x)",0,1,>add):
```

Maxima gagal menghitung integral tentu tersebut secara langsung menggunakan perintah integrate. Berikut kita lakukan seperti contoh sebelumnya untuk mendapat hasil atau pendekatan nilai integral tentu tersebut.

```
> t &= makelist(a,a,0,1-0.01,0.01);
> fx &= makelist(f(t[i]+0.01),i,1,length(t));
```

maxima: 'integrate(f(x),x,0,1) = 0.01*sum(fx[i],i,1,length(fx))

Apakah hasil tersebut cukup baik? perhatikan gambarnya.

```
> function f(x) &= sin(3*x^5+7)^2
```

$$\sin^2(3x^5 + 7)$$

```
> integrate(f,0,1)
```

0.542581176074

```
> &showev('integrate(f(x),x,0,1))
```

$$\int_0^1 \sin^2(3x^5 + 7) dx = \frac{\gamma(-\frac{1}{5}) \sin(\frac{14}{5}) \sin(\frac{\pi}{10})}{\Gamma(\frac{1}{5})}$$

```

/
          10 6
0
4/5           1           4/5           1
- ((6      gamma_incomplete(-, 6 I) + 6      gamma_incomplete(-, - 6 I))
      5           5
        4/5           1           pi
sin(14) + (6      I gamma_incomplete(-, 6 I)
      5
        4/5           1           pi
- 6      I gamma_incomplete(-, - 6 I)) cos(14)) sin(--)
      5           10

```

```
> &float(%)
```

```

1.0
/
[      2      5
I   sin (3.0 x  + 7.0) dx =
]
/
0.0
0.09820784258795788 - 0.008333333333333333
(0.3090169943749474 (0.1367372182078336
(4.192962712629476 I gamma_incomplete(0.2, 6.0 I)
- 4.192962712629476 I gamma_incomplete(0.2, - 6.0 I))
+ 0.9906073556948704 (4.192962712629476 gamma_incomplete(0.2, 6.0 I)
+ 4.192962712629476 gamma_incomplete(0.2, - 6.0 I))) - 60.0)

```

```
> $showev('integrate(x*exp(-x),x,0,1)) // Integral tentu (eksak)
```

Aplikasi Integral Tentu

```

> plot2d("x^3-x",-0.1,1.1); plot2d("-x^2",>add); ...
>b=solve("x^3-x+x^2",0.5); x=linspace(0,b,200); xi=flipx(x); ...
>plot2d(x|xi,x^3-x|-xi^2,>filled,style="|",fillcolor=1,>add); // Plot daerah antara 2 kurva
> a=solve("x^3-x+x^2",0), b=solve("x^3-x+x^2",1) // absis titik-titik potong kedua kurva

```

```
0
0.61803398875
```

```
> integrate("(-x^2)-(x^3-x)",a,b) // luas daerah yang diarsir
```

```
0.0758191713542
```

Hasil tersebut akan kita bandingkan dengan perhitungan secara analitik.

```
> a &= solve((-x^2)-(x^3-x),x); $a // menentukan absis titik potong kedua kurva secara eksak
> $showev('integrate(-x^2-x^3+x,x,0,(sqrt(5)-1)/2)) // Nilai integral secara eksak
> $float(%)
```

Panjang Kurva

Hitunglah panjang kurva berikut ini dan luas daerah di dalam kurva tersebut.

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

dengan

$$r(t) = 1 + \frac{\sin(3t)}{2}, \quad 0 \leq t \leq 2\pi.$$

```
> 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): // Kita gambar kurvanya terlebih dahulu
> function r(t) &= 1+sin(3*t)/2; $'r(t)=r(t)
> function fx(t) &= r(t)*cos(t); $'fx(t)=fx(t)
> function fy(t) &= r(t)*sin(t); $'fy(t)=fy(t)
> function ds(t) &= trigreduce(radcan(sqrt(diff(fx(t),t)^2+diff(fy(t),t)^2))); $'ds(t)=ds(t)
> $integrate(ds(x),x,0,2*pi) //panjang (keliling) kurva
```

Maxima gagal melakukan perhitungan eksak integral tersebut.

Berikut kita hitung integralnya secara numerik dengan perintah EMT.

```
> integrate("ds(x)",0,2*pi)
```

9.0749467823

Spiral Logaritmik

$$x = e^{ax} \cos x, \quad y = e^{ax} \sin x.$$

```
> a=0.1; plot2d("exp(a*x)*cos(x)","exp(a*x)*sin(x)",r=2,xmin=0,xmax=2*pi):
> &kill(a) // hapus expresi a
```

done

```
> function fx(t) &= exp(a*t)*cos(t); $'fx(t)=fx(t)
> function fy(t) &= exp(a*t)*sin(t); $'fy(t)=fy(t)
> function df(t) &= trigreduce(radcan(sqrt(diff(fx(t),t)^2+diff(fy(t),t)^2))); $'df(t)=df(t)
> S &= integrate(df(t),t,0,2*pi); $S // panjang kurva (spiral)
> S(a=0.1) // Panjang kurva untuk a=0.1
```

8.78817491636

Soal:

Tunjukkan bahwa keliling lingkaran dengan jari-jari r adalah $K=2\pi r$.

Berikut adalah contoh menghitung panjang parabola.

```
> plot2d("x^2",xmin=-1,xmax=1):
> $showev('integrate(sqrt(1+diff(x^2,x)^2),x,-1,1))
> $float(%)
> x=-1:0.2:1; y=x^2; plot2d(x,y); ...
> plot2d(x,y,points=1,style="o#",add=1):
```

Panjang tersebut dapat dihampiri dengan menggunakan jumlah panjang ruas-ruas garis yang menghubungkan titik-titik pada parabola tersebut.

```
> i=1:cols(x)-1; sum(sqrt((x[i+1]-x[i])^2+(y[i+1]-y[i])^2))
```

2.95191957027

Hasilnya mendekati panjang yang dihitung secara eksak. Untuk mendapatkan hampiran yang cukup akurat, jarak antar titik dapat diperkecil, misalnya 0.1, 0.05, 0.01, dan seterusnya. Cobalah Anda ulangi perhitungannya dengan nilai-nilai tersebut.

Koordinat Kartesius

Berikut diberikan contoh perhitungan panjang kurva menggunakan koordinat Kartesius. Kita akan hitung panjang kurva dengan persamaan implisit:

$$x^3 + y^3 - 3xy = 0.$$

```
> z &= x^3+y^3-3*x*y; $z
> plot2d(z,r=2,level=0,n=100):
```

Kita tertarik pada kurva di kuadran pertama.

```
> plot2d(z,a=0,b=2,c=0,d=2,level=[-10;0],n=100,contourwidth=3,style="/"':
```

Kita selesaikan persamaannya untuk x.

```
> $z with y=1*x, sol &= solve(% ,x); $sol
```

Kita gunakan solusi tersebut untuk mendefinisikan fungsi dengan Maxima.

```
> function f(l) &= rhs(sol[1]); $' f(l)=f(l)
```

Fungsi tersebut juga dapat digunakan untuk menggambar kurvanya. Ingat, bahwa fungsi tersebut adalah nilai x dan nilai $y=l^*x$, yakni $x=f(l)$ dan $y=l^*f(l)$.

```
> plot2d(&f(x),&x*f(x),xmin=-0.5,xmax=2,a=0,b=2,c=0,d=2,r=1.5):
```

Elemen panjang kurva adalah:

$$ds = \sqrt{f'(l)^2 + (lf'(l) + f(l))^2}.$$

```
> function ds(l) &= ratsimp(sqrt(diff(f(l),l)^2+diff(l*f(l),l)^2)); $' ds(l)=ds(l)
> $integrate(ds(l),l,0,1)
```

Integral tersebut tidak dapat dihitung secara eksak menggunakan Maxima. Kita hitung integral tersebut secara numerik dengan Euler. Karena kurva simetris, kita hitung untuk nilai variabel integrasi dari 0 sampai 1, kemudian hasilnya dikalikan 2.

```
> 2*integrate("ds(x)",0,1)
```

4.91748872168

```
> 2*romberg(&ds(x),0,1)// perintah Euler lain untuk menghitung nilai hampiran integral yan
```

4.91748872168

Perhitungan di atas dapat dilakukan untuk sebarang fungsi x dan y dengan mendefinisikan fungsi EMT, misalkan kita beri nama panjangkurva. Fungsi ini selalu memanggil Maxima untuk menurunkan fungsi yang diberikan.

```
> function panjangkurva(fx,fy,a,b) ...
```

```
ds=mxm("sqrt(diff(@fx,x)^2+diff(@fy,x)^2)");
return romberg(ds,a,b);
endfunction
```

```
> panjangkurva("x","x^2",-1,1) // cek untuk menghitung panjang kurva parabola sebelumnya
```

2.95788571509

Bandingkan dengan nilai eksak di atas.

```
> 2*panjangkurva(mxm("f(x)"),mxm("x*f(x)"),0,1) // cek contoh terakhir, bandingkan hasilnya
```

4.91748872168

Kita hitung panjang spiral Archimedes berikut ini dengan fungsi tersebut.

```
> plot2d("x*cos(x)", "x*sin(x)", xmin=0, xmax=2*pi, square=1):  
> panjangkurva("x*cos(x)", "x*sin(x)", 0, 2*pi)
```

21.2562941482

Berikut kita definisikan fungsi yang sama namun dengan Maxima, untuk perhitungan eksak.

```
> &kill(ds,x,fx,fy)
```

done

```
> function ds(fx,fy) &=& sqrt(diff(fx,x)^2+diff(fy,x)^2)
```

$$\sqrt{\left(\frac{d}{dx} f_y(x)\right)^2 + \left(\frac{d}{dx} f_x(x)\right)^2}$$

```
> sol &= ds(x*cos(x),x*sin(x)); $sol // Kita gunakan untuk menghitung panjang kurva terakhir  
> $sol | trigreduce | expand, $integrate(%,x,0,2*pi), %()
```

21.2562941482

Hasilnya sama dengan perhitungan menggunakan fungsi EMT.

Berikut adalah contoh lain penggunaan fungsi Maxima tersebut.

```
> plot2d("3*x^2-1", "3*x^3-1", xmin=-1/sqrt(3), xmax=1/sqrt(3), square=1):  
> sol &= radcan(ds(3*x^2-1, 3*x^3-1)); $sol  
> $showev('integrate(sol,x,0,1/sqrt(3))), $2*float(%) // panjang kurva di atas  
> x &= r*(t-sin(t))
```

$$r(t - \sin(t))$$

```
> y &= r*(1-cos(t))
```

$$r(1 - \cos(t))$$

Berikut kita gambar sikloid untuk r=1.

```
> ex &= x-sin(x); ey &= 1-cos(x); aspect(1);
> plot2d(ex,ey,xmin=0,xmax=4pi,square=1); ...
> plot2d("2+cos(x)","1+sin(x)",xmin=0,xmax=2pi,>add,color=blue); ...
> plot2d([2,ex(2)],[1,ey(2)],color=red,>add); ...
> plot2d(ex(2),ey(2),>points,>add,color=red); ...
> plot2d("2pi+cos(x)","1+sin(x)",xmin=0,xmax=2pi,>add,color=blue); ...
> plot2d([2pi,ex(2pi)],[1,ey(2pi)],color=red,>add); ...
> plot2d(ex(2pi),ey(2pi),>points,>add,color=red):
```

Berikut dihitung panjang lintasan untuk 1 putaran penuh. (Jangan salah menduga bahwa panjang lintasan 1 putaran penuh sama dengan keliling lingkaran!)

```
> ds &= radcan(sqrt(diff(ex,x)^2+diff(ey,x)^2)); $ds=trigsimp(ds) // elemen panjang kurva
> ds &= trigsimp(ds); $ds
> $showev('integrate(ds,x,0,2*pi)) // hitung panjang sikloid satu putaran penuh
> integrate(mxm("ds"),0,2*pi) // hitung secara numerik
```

8

```
> romberg(mxm("ds"),0,2*pi) // cara lain hitung secara numerik
```

8

Perhatikan, seperti terlihat pada gambar, panjang sikloid lebih besar daripada keliling lingkarannya, yakni:

2π .

Kurvatur (Kelengkungan) Kurva

image: Osculating.png

Aslinya, kelengkungan kurva diferensiabel (yakni, kurva mulus yang tidak lancip) di titik P didefinisikan melalui lingkaran oskulasi (yaitu, lingkaran yang melalui titik P dan terbaik memperkirakan, paling banyak menyinggung kurva di sekitar P). Pusat dan radius kelengkungan kurva di P adalah pusat dan radius lingkaran oskulasi. Kelengkungan adalah kebalikan dari radius kelengkungan:

$$\kappa = \frac{1}{R}$$

dengan R adalah radius kelengkungan. (Setiap lingkaran memiliki kelengkungan ini pada setiap titiknya, dapat diartikan, setiap lingkaran berputar 2π sejauh $2\pi R$.)

Definisi ini sulit dimanipulasi dan dinyatakan ke dalam rumus untuk kurva umum. Oleh karena itu digunakan definisi lain yang ekivalen.

Definisi Kurvatur dengan Fungsi Parametrik Panjang Kurva

Setiap kurva diferensiabel dapat dinyatakan dengan persamaan parametrik terhadap panjang kurva s:

$$\gamma(s) = (x(s), y(s)),$$

dengan x dan y adalah fungsi riil yang diferensiabel, yang memenuhi:

$$\|\gamma'(s)\| = \sqrt{x'(s)^2 + y'(s)^2} = 1.$$

Ini berarti bahwa vektor singgung

$$\mathbf{T}(s) = (x'(s), y'(s))$$

memiliki norm 1 dan merupakan vektor singgung satuan.

Apabila kurvanya memiliki turunan kedua, artinya turunan kedua x dan y ada, maka $\mathbf{T}'(s)$ ada. Vektor ini merupakan normal kurva yang arahnya menuju pusat kurvatur, norm-nya merupakan nilai kurvatur (kelengkungan):

$$\begin{aligned}\mathbf{T}(s) &= \gamma'(s), \\ \mathbf{T}^2(s) &= 1 \text{ (konstanta)} \Rightarrow \mathbf{T}'(s) \cdot \mathbf{T}(s) = 0 \\ \kappa(s) &= \|\mathbf{T}'(s)\| = \|\gamma''(s)\| = \sqrt{x''(s)^2 + y''(s)^2}.\end{aligned}$$

Nilai

$$R(s) = \frac{1}{\kappa(s)}$$

disebut jari-jari (radius) kelengkungan kurva.

Bilangan riil

$$k(s) = \pm \kappa(s)$$

disebut nilai kelengkungan bertanda.

Contoh:

Akan ditentukan kurvatur lingkaran

$$x = r \cos t, \quad y = r \sin t.$$

```
> fx &= r*cos(t); fy &= r*sin(t);
> &assume(t>0, r>0); s &=integrate(sqrt(diff(fx,t)^2+diff(fy,t)^2),t,0,t); s // elemen panj
```

r t

```
> &kill(s); fx &= r*cos(s/r); fy &= r*sin(s/r); // definisi ulang persamaan parametrik terh
> k &= trigsimp(sqrt(diff(fx,s,2)^2+diff(fy,s,2)^2)); $k // nilai kurvatur lingkaran dengan
```

Untuk representasi parametrik umum, misalkan

$$x = x(t), \quad y = y(t)$$

merupakan persamaan parametrik untuk kurva bidang yang terdiferensialkan dua kali. Kurvatur untuk kurva tersebut didefinisikan sebagai

$$\begin{aligned}\kappa &= \frac{d\phi}{ds} = \frac{\frac{d\phi}{dt}}{\frac{ds}{dt}} \quad (\phi \text{ adalah sudut kemiringan garis singgung dan } s \text{ adalah panjang kurva}) \\ &= \frac{\frac{d\phi}{dt}}{\sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2}} = \frac{\frac{d\phi}{dt}}{\sqrt{x'(t)^2 + y'(t)^2}}.\end{aligned}$$

Selanjutnya, pembilang pada persamaan di atas dapat dicari sebagai berikut.

$$\sec^2 \phi \frac{d\phi}{dt} = \frac{d}{dt} (\tan \phi) = \frac{d}{dt} \left(\frac{dy}{dx} \right) = \frac{d}{dt} \left(\frac{dy/dt}{dx/dt} \right) = \frac{d}{dt} \left(\frac{y'(t)}{x'(t)} \right) = \frac{x'(t)y''(t) - x''(t)y'(t)}{x'(t)^2}.$$

$$\begin{aligned}\frac{d\phi}{dt} &= \frac{1}{\sec^2 \phi} \frac{x'(t)y''(t) - x''(t)y'(t)}{x'(t)^2} \\ &= \frac{1}{1 + \tan^2 \phi} \frac{x'(t)y''(t) - x''(t)y'(t)}{x'(t)^2} \\ &= \frac{1}{1 + \left(\frac{y'(t)}{x'(t)}\right)^2} \frac{x'(t)y''(t) - x''(t)y'(t)}{x'(t)^2} \\ &= \frac{x'(t)y''(t) - x''(t)y'(t)}{x'(t)^2 + y'(t)^2}.\end{aligned}$$

Jadi, rumus kurvatur untuk kurva parametrik

$$x = x(t), \quad y = y(t)$$

adalah

$$\kappa(t) = \frac{x'(t)y''(t) - x''(t)y'(t)}{(x'(t)^2 + y'(t)^2)^{3/2}}.$$

Jika kurvanya dinyatakan dengan persamaan parametrik pada koordinat kutub

$$x = r(\theta) \cos \theta, \quad y = r(\theta) \sin \theta,$$

maka rumus kurvurnya adalah

$$\kappa(\theta) = \frac{r(\theta)^2 + 2r'(\theta)^2 - r(\theta)r''(\theta)}{(r'(\theta)^2 + r''(\theta)^2)^{3/2}}.$$

(Silakan Anda turunkan rumus tersebut!)

Contoh:

Lingkaran dengan pusat (0,0) dan jari-jari r dapat dinyatakan dengan persamaan parametrik

$$x = r \cos t, \quad y = r \sin t.$$

Nilai kelengkungan lingkaran tersebut adalah

$$\kappa(t) = \frac{x'(t)y''(t) - x''(t)y'(t)}{(x'(t)^2 + y'(t)^2)^{3/2}} = \frac{r^2}{r^3} = \frac{1}{r}.$$

Hasil cocok dengan definisi kurvatur suatu kelengkungan.

Kurva

$$y = f(x)$$

dapat dinyatakan ke dalam persamaan parametrik

$$x = t, \quad y = f(t), \quad \text{dengan } x'(t) = 1, \quad x''(t) = 0,$$

sehingga kurvaturnya adalah

$$\kappa(t) = \frac{y''(t)}{(1 + y'(t)^2)^{3/2}}.$$

Contoh:

Akan ditentukan kurvatur parabola

$$y = ax^2 + bx + c.$$

```
> function f(x) &= a*x^2+b*x+c; $y=f(x)
> function k(x) &= (diff(f(x),x,2))/(1+diff(f(x),x)^2)^(3/2); $('k(x)=k(x)') // kelengkungan
> function f(x) &= x^2+x+1; $y=f(x) // akan kita plot kelengkungan parabola untuk a=b=c=1
> function k(x) &= (diff(f(x),x,2))/(1+diff(f(x),x)^2)^(3/2); '$k(x)=k(x)' // kelengkungan
```

Berikut kita gambar parabola tersebut beserta kurva kelengkungan, kurva jari-jari kelengkungan dan salah satu lingkaran oskulasi di titik puncak parabola. Perhatikan, puncak parabola dan jari-jari lingkaran oskulasi di puncak parabola adalah

$$(-1/2, 3/4), \quad 1/k(2) = 1/2,$$

sehingga pusat lingkaran oskulasi adalah $(-1/2, 5/4)$.

```
> plot2d(["f(x)", "k(x)"], -2, 1, color=[blue, red]); plot2d("1/k(x)", -1.5, 1, color=green, >add
> plot2d("-1/2+1/k(-1/2)*cos(x)", "5/4+1/k(-1/2)*sin(x)", xmin=0, xmax=2pi, >add, color=blue):
```

Untuk kurva yang dinyatakan dengan fungsi implisit

$$F(x, y) = 0$$

dengan turunan-turunan parsial

$$F_x = \frac{\partial F}{\partial x}, \quad F_y = \frac{\partial F}{\partial y}, \quad F_{xy} = \frac{\partial}{\partial y} \left(\frac{\partial F}{\partial x} \right), \quad F_{xx} = \frac{\partial}{\partial x} \left(\frac{\partial F}{\partial x} \right), \quad F_{yy} = \frac{\partial}{\partial y} \left(\frac{\partial F}{\partial y} \right),$$

berlaku

$$F_x dx + F_y dy = 0 \text{ atau } \frac{dy}{dx} = -\frac{F_x}{F_y},$$

sehingga kurvturnya adalah

$$\kappa = \frac{F_y^2 F_{xx} - 2F_x F_y F_{xy} + F_x^2 F_{yy}}{(F_x^2 + F_y^2)^{3/2}}.$$

(Silakan Anda turunkan sendiri!)

Contoh 1:

Parabola

$$y = ax^2 + bx + c$$

dapat dinyatakan ke dalam persamaan implisit

$$ax^2 + bx + c - y = 0.$$

```
> function F(x,y) &= a*x^2+b*x+c-y; $F(x,y)
> Fx &= diff(F(x,y),x), Fxx &= diff(F(x,y),x,2), FY &= diff(F(x,y),y), Fxy &= diff(diff(F(x,y),x),y)
```

$$2 \ a \ x + b$$

$$2 \ a$$

$$- 1$$

$$0$$

$$0$$

```
> function k(x) &= (FY^2*Fxx-2*Fx*FY*Fxy+Fx^2*Fyy)/(Fx^2+Fy^2)^(3/2); $' k(x)=k(x) // kurva
```

Hasilnya sama dengan sebelumnya yang menggunakan persamaan parabola biasa.

Latihan

- Bukalah buku Kalkulus.
- Cari dan pilih beberapa (paling sedikit 5 fungsi berbeda tipe/bentuk/jenis) fungsi dari buku tersebut, kemudian definisikan di EMT pada baris-baris perintah berikut (jika perlu tambahkan lagi).
- Untuk setiap fungsi, tentukan anti turunannya (jika ada), hitunglah integral tentu dengan batas-batas yang menarik (Anda tentukan sendiri), seperti contoh-contoh tersebut.
- Lakukan hal yang sama untuk fungsi-fungsi yang tidak dapat diintegralkan (cari sedikitnya 3 fungsi).
- Gambar grafik fungsi dan daerah integrasinya pada sumbu koordinat yang sama.
- Gunakan integral tentu untuk mencari luas daerah yang dibatasi oleh dua kurva yang berpotongan di dua

titik. (Cari dan gambar kedua kurva dan arsir (warnai) daerah yang dibatasi oleh keduanya.)

- Gunakan integral tentu untuk menghitung volume benda putar kurva $y = f(x)$ yang diputar mengelilingi sumbu x dari $x=a$ sampai $x=b$, yakni

$$V = \int_a^b \pi(f(x)^2) dx.$$

(Pilih fungsinya dan gambar kurva dan benda putar yang dihasilkan. Anda dapat mencari contoh-contoh bagaimana cara menggambar benda hasil perputaran suatu kurva.)

- Gunakan integral tentu untuk menghitung panjang kurva $y=f(x)$ dari $x=a$ sampai $x=b$ dengan menggunakan rumus:

$$S = \int_a^b \sqrt{1 + (f'(x))^2} dx.$$

(Pilih fungsi dan gambar kurvanya.)

- Apabila fungsi dinyatakan dalam koordinat kutub $x=f(r,t)$, $y=g(r,t)$, $r=h(t)$, $x=a$ bersesuaian dengan $t=t_0$ dan $x=b$ bersesuaian dengan $t=t_1$, maka rumus di atas akan menjadi:

$$S = \int_{t_0}^{t_1} \sqrt{x'(t)^2 + y'(t)^2} dt.$$

- Pilih beberapa kurva menarik (selain lingkaran dan parabola) dari buku kalkulus. Nyatakan setiap kurva tersebut dalam bentuk:

- a. koordinat Kartesius (persamaan $y=f(x)$)
- b. koordinat kutub ($r=r(\theta)$)
- c. persamaan parametrik $x=x(t)$, $y=y(t)$
- d. persamaan implisit $F(x, y)=0$

- Tentukan kurvatur masing-masing kurva dengan menggunakan keempat representasi tersebut (hasilnya harus sama).

- Gambarlah kurva asli, kurva kurvatur, kurva jari-jari lingkaran oskulasi, dan salah satu lingkaran oskulasinya.

1. Tentukan panjang kurva dan volume benda putar kurva $y=f(x)$ yang diputar mengelilingi sumbu x dari $x=0$ sampai $x=4$

$$f(x) = x^3$$

```
> function f(x) &= x^3; $f(x)
> $showev('integrate(pi*(f(x))^2, x, 0, 4))
```

turunan fungsi $f(x)$

```
> function df(x) &= limit((f(x+h)-f(x))/h, h, 0); $df(x) // df(x) = f'(x)
```

panjang kurva

```
> $showev('integrate(sqrt((1+df(x)^2)),x,0,4))
```

grafik benda putar mengelilingi sumbu-x

```
> plot3d("x^3",2,0,2,rotate=2):
```

2. Tentukan panjang kurva dan volume benda putar kurva $y=f(x)$ yang diputar mengelilingi sumbu x dari $x=-1$ sampai $x=2$

$$f(x) = 3x^2 + 2x + 1$$

volume benda putar

```
> function f(x) &= 3*x^2+2*x+1; $f(x)
> $showev('integrate(pi*(f(x))^2,x,-1,2))
```

menentukan turunan fungsi $f(x)$

```
> function df(x) &= limit((f(x+h)-f(x))/h,h,0); $df(x) // df(x) = f'(x)
```

menentukan panjang kurva

```
> $showev('integrate(sqrt((1+df(x)^2)),x,-1,2))
```

menggambar plot benda putar mengelilingi sumbu-x

```
> plot3d("3x^2+2x+1",2,-1,2,rotate=2):
```

3. Integral tentu dengan batas $[-1,1]$ dari fungsi berikut

$$f(x) = x^2 + 8x - 9$$

```
> function map f(x) &= (x^2+8*x-9); $f(x)
```

mencari nilai dari integral tentu fungsi $f(x)$ dengan batas $[-1,1]$

```
> $showev('integrate(f(x), x, -1, 1))
```

4. Tentukan panjang kurva dan volume benda putar kurva $y=f(x)$ yang diputar mengelilingi sumbu x dari $x=0$ sampai $x=2$

$$f(x) = 4x^2 + 1$$

```
> $showev('integrate(pi*(f(x))^2, x, 0, 2))  
> function df(x) &= limit((f(x+h)-f(x))/h, h, 0); $df(x) // df(x) = f'(x)
```

menentukan pangjang kurva

```
> $showev('integrate(sqrt((1+df(x)^2)), x, 0, 2))
```

menggambar plot

```
> plot3d("4x^2+1", 2, 0, 2, rotate=2):
```

5. Tentukan integral fungsi berikut.

$$f(x) = \frac{\sqrt{2x}}{x} + \frac{3}{x^5}$$

```
> function f(x) &= (sqrt(2*x))/x + 3/x^5; $f(x)  
> $showev('integrate(((sqrt(2*x))/x)+(3/x^5), x))  
> x=0:0.1:5-0.1; plot2d(x, f(x+0.1), >bar); plot2d("f(x)", 0, 5, >add):
```

Barisan dan Deret

(Catatan: bagian ini belum lengkap. Anda dapat membaca contoh-contoh penggunaan EMT dan Maxima untuk menghitung limit barisan, rumus jumlah parsial suatu deret, jumlah tak hingga suatu deret konvergen, dan sebagainya. Anda dapat mengeksplor contoh-contoh di EMT atau perbagai panduan penggunaan Maxima di software Maxima atau dari Internet.)

Barisan dapat didefinisikan dengan beberapa cara di dalam EMT, di antaranya:

- dengan cara yang sama seperti mendefinisikan vektor dengan elemen-elemen beraturan (menggunakan titik dua ":");
- menggunakan perintah "sequence" dan rumus barisan (suku ke -n);
- menggunakan perintah "iterate" atau "niterate";
- menggunakan fungsi Maxima "create_list" atau "makelist" untuk menghasilkan barisan simbolik;
- menggunakan fungsi biasa yang inputnya vektor atau barisan;
- menggunakan fungsi rekursif.

EMT menyediakan beberapa perintah (fungsi) terkait barisan, yakni:

- sum: menghitung jumlah semua elemen suatu barisan
- cumsum: jumlah kumulatif suatu barisan
- differences: selisih antar elemen-elemen berturutan

EMT juga dapat digunakan untuk menghitung jumlah deret berhingga maupun deret tak hingga, dengan menggunakan perintah (fungsi) "sum". Perhitungan dapat dilakukan secara numerik maupun simbolik dan eksak.

Berikut adalah beberapa contoh perhitungan barisan dan deret menggunakan EMT.

```
> 1:10 // barisan sederhana
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
> 1:2:30
```

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29]
```

Iterasi dan Barisan

EMT menyediakan fungsi iterate("g(x)", x0, n) untuk melakukan iterasi

$$x_{k+1} = g(x_k), \quad x_0 = x_0, \quad k = 1, 2, 3, \dots, n.$$

Berikut ini disajikan contoh-contoh penggunaan iterasi dan rekursi dengan EMT. Contoh pertama menunjukkan pertumbuhan dari nilai awal 1000 dengan laju pertambahan 5%, selama 10 periode.

```
> q=1.05; iterate("x*q",1000,n=10)'
```

```
1000
1050
1102.5
1157.63
1215.51
1276.28
1340.1
1407.1
1477.46
1551.33
1628.89
```

Contoh berikutnya memperlihatkan bahaya menabung di bank pada masa sekarang! Dengan bunga tabungan sebesar 6% per tahun atau 0.5% per bulan dipotong pajak 20%, dan biaya administrasi 10000 per bulan, tabungan sebesar 1 juta tanpa diambil selama sekitar 10 tahunan akan habis diambil oleh bank!

```
> r=0.005; plot2d(iterate("(1+0.8*r)*x-10000",1000000,n=130)):
```

Silakan Anda coba-coba, dengan tabungan minimal berapa agar tidak akan habis diambil oleh bank dengan ketentuan bunga dan biaya administrasi seperti di atas.

Berikut adalah perhitungan minimal tabungan agar aman di bank dengan bunga sebesar r dan biaya administrasi a , pajak bunga 20%.

```
> $solve(0.8*r*A-a,A), $% with [r=0.005, a=10]
```

Berikut didefinisikan fungsi untuk menghitung saldo tabungan, kemudian dilakukan iterasi.

```
> function saldo(x,r,a) := round((1+0.8*r)*x-a,2);
> iterate({{"saldo",0.005,10}},1000,n=6)
```

```
[1000, 994, 987.98, 981.93, 975.86, 969.76, 963.64]
```

```
> iterate({{"saldo",0.005,10}},2000,n=6)
```

```
[2000, 1998, 1995.99, 1993.97, 1991.95, 1989.92, 1987.88]
```

```
> iterate({{"saldo",0.005,10}},2500,n=6)
```

```
[2500, 2500, 2500, 2500, 2500, 2500, 2500]
```

Tabungan senilai 2,5 juta akan aman dan tidak akan berubah nilai (jika tidak ada penarikan), sedangkan jika tabungan awal kurang dari 2,5 juta, lama kelamaan akan berkurang meskipun tidak pernah dilakukan penarikan uang tabungan.

```
> iterate({{"saldo",0.005,10}},3000,n=6)
```

```
[3000, 3002, 3004.01, 3006.03, 3008.05, 3010.08, 3012.12]
```

Tabungan yang lebih dari 2,5 juta baru akan bertambah jika tidak ada penarikan.

Untuk barisan yang lebih kompleks dapat digunakan fungsi "sequence()". Fungsi ini menghitung nilai-nilai $x[n]$ dari semua nilai sebelumnya, $x[1], \dots, x[n-1]$ yang diketahui.

Berikut adalah contoh barisan Fibonacci.

$$x_n = x_{n-1} + x_{n-2}, \quad x_1 = 1, \quad x_2 = 1$$

```
> sequence("x[n-1]+x[n-2]", [1,1], 15)
```

```
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610]
```

Barisan Fibonacci memiliki banyak sifat menarik, salah satunya adalah akar pangkat ke-n suku ke-n akan konvergen ke pecahan emas:

```
> $' (1+sqrt(5))/2=float((1+sqrt(5))/2)
> plot2d(sequence("x[n-1]+x[n-2]", [1,1], 250)^(1/(1:250))):
```

Barisan yang sama juga dapat dihasilkan dengan menggunakan loop.

```
> x=ones(500); for k=3 to 500; x[k]=x[k-1]+x[k-2]; end;
```

Rekursi dapat dilakukan dengan menggunakan rumus yang tergantung pada semua elemen sebelumnya. Pada contoh berikut, elemen ke-n merupakan jumlah (n-1) elemen sebelumnya, dimulai dengan 1 (elemen ke-1). Jelas, nilai elemen ke-n adalah 2^{n-2} , untuk n=2, 4, 5,

```
> sequence("sum(x)", 1, 10)
```

```
[1, 1, 2, 4, 8, 16, 32, 64, 128, 256]
```

Selain menggunakan ekspresi dalam x dan n, kita juga dapat menggunakan fungsi.

Pada contoh berikut, digunakan iterasi

$$x_n = A \cdot x_{n-1},$$

dengan A suatu matriks 2x2, dan setiap x[n] merupakan matriks/vektor 2x1.

```
> A=[1,1;1,2]; function suku(x,n) := A.x[,n-1]
> sequence("suku", [1;1], 6)
```

Real 2 x 6 matrix

1	2	5	13	...
1	3	8	21	...

Hasil yang sama juga dapat diperoleh dengan menggunakan fungsi perpangkatan matriks "matrixpower()". Cara ini lebih cepat, karena hanya menggunakan perkalian matriks sebanyak $\log_2(n)$.

$$x_n = A \cdot x_{n-1} = A^2 \cdot x_{n-2} = A^3 \cdot x_{n-3} = \dots = A^{n-1} \cdot x_1.$$

```
> sequence("matrixpower(A, n) . [1;1]", 1, 6)
```

Real 2 x 6 matrix

1	5	13	34	...
1	8	21	55	...

image: Spiral_of_Theodorus.png

Spiral Theodorus (spiral segitiga siku-siku) dapat digambar secara rekursif. Rumus rekursifnya adalah:

$$x_n = \left(1 + \frac{i}{\sqrt{n-1}}\right) x_{n-1}, \quad x_1 = 1,$$

yang menghasilkan barisan bilangan kompleks.

```
> function g(n) := 1+I/sqrt(n)
```

Rekursinya dapat dijalankan sebanyak 17 untuk menghasilkan barisan 17 bilangan kompleks, kemudian digambar bilangan-bilangan kompleksnya.

```
> x=sequence("g(n-1)*x[n-1]",1,17); plot2d(x,r=3.5); textbox(latex("Spiral\\ Theodorus"),0,
```

Selanjutnya dihubungkan titik 0 dengan titik-titik kompleks tersebut menggunakan loop.

```
> for i=1:cols(x); plot2d([0,x[i]],>add); end;
```

Spiral tersebut juga dapat didefinisikan menggunakan fungsi rekursif, yang tidak memerlukan indeks dan bilangan kompleks. Dalam hal ini digunakan vektor kolom pada bidang.

```
> function gstep (v) ...
```

```
w=[-v[2];v[1]];
return v+w/norm(w);
endfunction
```

Jika dilakukan iterasi 16 kali dimulai dari [1;0] akan didapatkan matriks yang memuat vektor-vektor dari setiap iterasi.

```
> x=iterate("gstep", [1;0],16); plot2d(x[1],x[2],r=3.5,>points):
```

Kekonvergenan

Terkadang kita ingin melakukan iterasi sampai konvergen. Apabila iterasinya tidak konvergen setelah ditunggu lama, Anda dapat menghentikannya dengan menekan tombol [ESC].

```
> iterate("cos(x)",1) // iterasi x(n+1)=cos(x(n)), dengan x(0)=1.
```

0.739085133216

Iterasi tersebut konvergen ke penyelesaian persamaan

$$x = \cos(x).$$

Iterasi ini juga dapat dilakukan pada interval, hasilnya adalah barisan interval yang memuat akar tersebut.

```
> hasil := iterate("cos(x)",~1,2~) //iterasi x(n+1)=cos(x(n)), dengan interval awal (1, 2)
```

```
~0.739085133211, 0.7390851332133~
```

Jika interval hasil tersebut sedikit diperlebar, akan terlihat bahwa interval tersebut memuat akar persamaan $x=\cos(x)$.

```
> h=expand(hasil,100), cos(h) << h
```

```
~0.73908513309, 0.73908513333~
```

```
1
```

Iterasi juga dapat digunakan pada fungsi yang didefinisikan.

```
> function f(x) := (x+2/x)/2
```

Iterasi $x(n+1)=f(x(n))$ akan konvergen ke akar kuadrat 2.

```
> iterate("f",2), sqrt(2)
```

```
1.41421356237
```

```
1.41421356237
```

Jika pada perintah iterate diberikan tambahan parameter n, maka hasil iterasinya akan ditampilkan mulai dari iterasi pertama sampai ke-n.

```
> iterate("f",2,5)
```

```
[2, 1.5, 1.41667, 1.41422, 1.41421, 1.41421]
```

Untuk iterasi ini tidak dapat dilakukan terhadap interval.

```
> niterate("f",~1,2~,5)
```

```
[ ~1,2~, ~1,2~, ~1,2~, ~1,2~, ~1,2~, ~1,2~ ]
```

Perhatikan, hasil iterasinya sama dengan interval awal. Alasannya adalah perhitungan dengan interval bersifat terlalu longgar. Untuk meningkatkan perhitungan pada ekspresi dapat digunakan pembagian intervalnya, menggunakan fungsi ieval().

```
> function s(x) := ieval("(x+2/x)/2",x,10)
```

Selanjutnya dapat dilakukan iterasi hingga diperoleh hasil optimal, dan intervalnya tidak semakin mengecil. Hasilnya berupa interval yang memuat akar persamaan:

$$x = \frac{1}{2} \left(x + \frac{2}{x} \right).$$

Satu-satunya solusi adalah

$$x = \sqrt{2}.$$

```
> iterate("s", ~1, 2~)
```

~1.41421356236, 1.41421356239~

Fungsi "iterate()" juga dapat bekerja pada vektor. Berikut adalah contoh fungsi vektor, yang menghasilkan rata-rata aritmetika dan rata-rata geometri.

$$(a_{n+1}, b_{n+1}) = \left(\frac{a_n + b_n}{2}, \sqrt{a_n b_n} \right)$$

Iterasi ke-n disimpan pada vektor kolom x[n].

```
> function g(x) := [(x[1]+x[2])/2; sqrt(x[1]*x[2])]
```

Iterasi dengan menggunakan fungsi tersebut akan konvergen ke rata-rata aritmetika dan geometri dari nilai-nilai awal.

```
> iterate("g", [1; 5])
```

2.60401
2.60401

Hasil tersebut konvergen agak cepat, seperti kita cek sebagai berikut.

```
> iterate("g", [1; 5], 4)
```

1	3	2.61803	2.60403	2.60401
5	2.23607	2.59002	2.60399	2.60401

Iterasi pada interval dapat dilakukan dan stabil, namun tidak menunjukkan bahwa limitnya pada batas-batas yang dihitung.

```
> iterate("g", [~1~; ~5~], 4)
```

Interval 2 x 5 matrix

~0.99999999999999778, 1.0000000000000022~ ...
~4.999999999999911, 5.0000000000000089~ ...

Iterasi berikut konvergen sangat lambat.

$$x_{n+1} = \sqrt{x_n}.$$

```
> iterate("sqrt(x)", 2, 10)
```

```
[2, 1.41421, 1.18921, 1.09051, 1.04427, 1.0219, 1.01089,  
1.00543, 1.00271, 1.00135, 1.00068]
```

Kekonvergenan iterasi tersebut dapat dipercepat dengan percepatan Steffenson:

```
> steffenson("sqrt(x)", 2, 10)
```

```
[1.04888, 1.00028, 1, 1]
```

Iterasi menggunakan Loop yang ditulis Langsung

Berikut adalah beberapa contoh penggunaan loop untuk melakukan iterasi yang ditulis langsung pada baris perintah.

```
> x=2; repeat x=(x+2/x)/2; until x^2~=2; end; x,
```

```
1.41421356237
```

Penggabungan matriks menggunakan tanda "|" dapat digunakan untuk menyimpan semua hasil iterasi.

```
> v=[1]; for i=2 to 8; v=v|(v[i-1]*i); end; v,
```

```
[1, 2, 6, 24, 120, 720, 5040, 40320]
```

Hasil iterasi juga dapat disimpan pada vektor yang sudah ada.

```
> v=ones(1,100); for i=2 to cols(v); v[i]=v[i-1]*i; end; ...  
> plot2d(v, logplot=1); textbox(latex(&log(n)), x=0.5):  
> A =[0.5, 0.2; 0.7, 0.1]; b=[2; 2]; ...  
> x=[1; 1]; repeat xnew=A.x-b; until all(xnew~≈x); x=xnew; end; ...  
> x,
```

```
-7.09677  
-7.74194
```

Iterasi di dalam Fungsi

Fungsi atau program juga dapat menggunakan iterasi dan dapat digunakan untuk melakukan iterasi. Berikut adalah beberapa contoh iterasi di dalam fungsi.

Contoh berikut adalah suatu fungsi untuk menghitung berapa lama suatu iterasi konvergen. Nilai fungsi tersebut adalah hasil akhir iterasi dan banyak iterasi sampai konvergen.

```
> function map hiter(f$,x0) ...
```

```
x=x0;
maxiter=0;
repeat
    xnew=f$(x);
    maxiter=maxiter+1;
    until xnew~≈x;
    x=xnew;
end;
return maxiter;
endfunction
```

Misalnya, berikut adalah iterasi untuk mendapatkan hampiran akar kuadrat 2, cukup cepat, konvergen pada iterasi ke-5, jika dimulai dari hampiran awal 2.

```
> hiter("(x+2/x)/2",2)
```

5

Karena fungsinya didefinisikan menggunakan "map". maka nilai awalnya dapat berupa vektor.

```
> x=1.5:0.1:10; hasil=hiter("(x+2/x)/2",x); ...
> plot2d(x,hasil):
```

Dari gambar di atas terlihat bahwa kekonvergenan iterasinya semakin lambat, untuk nilai awal semakin besar, namun penambahannya tidak kontinu. Kita dapat menemukan kapan maksimum iterasinya bertambah.

```
> hasil[1:10]
```

[4, 5, 5, 5, 5, 5, 6, 6, 6]

```
> x[nonzeros(differences(hasil))]
```

[1.5, 2, 3.4, 6.6]

maksimum iterasi sampai konvergen meningkat pada saat nilai awalnya 1.5, 2, 3.4, dan 6.6.

Contoh berikutnya adalah metode Newton pada polinomial kompleks berderajat 3.

```
> p &= x^3-1; newton &= x-p/diff(p,x); $newton
```

Selanjutnya didefinisikan fungsi untuk melakukan iterasi (aslinya 10 kali).

```
> function iterasi(f$,x,n=10) ...
```

```
loop 1 to n; x=f$(x); end;
return x;
endfunction
```

Kita mulai dengan menentukan titik-titik grid pada bidang kompleksnya.

```
> r=1.5; x=linspace(-r,r,501); z=x+I*x'; W=iterasi(newton,z);
```

Berikut adalah akar-akar polinomial di atas.

```
> z=&solve(p)()
```

```
[ -0.5+0.866025i, -0.5-0.866025i, 1+0i ]
```

Untuk menggambar hasil iterasinya, dihitung jarak dari hasil iterasi ke-10 ke masing-masing akar, kemudian digunakan untuk menghitung warna yang akan digambar, yang menunjukkan limit untuk masing-masing nilai awal.

Fungsi plotrgb() menggunakan jendela gambar terkini untuk menggambar warna RGB sebagai matriks.

```
> C=rgb(max(abs(W-z[1]),1),max(abs(W-z[2]),1),max(abs(W-z[3]),1)); ...
> plot2d(None,-r,r,-r,r); plotrgb(C);
```

Iterasi Simbolik

Seperti sudah dibahas sebelumnya, untuk menghasilkan barisan ekspresi simbolik dengan Maxima dapat digunakan fungsi makelist().

```
> &powerdisp:true // untuk menampilkan deret pangkat mulai dari suku berpangkat terkecil
```

```
true
```

```
> deret &= makelist(taylor(exp(x),x,0,k),k,1,3); $deret // barisan deret Taylor untuk e^x
```

Untuk mengubah barisan deret tersebut menjadi vektor string di EMT digunakan fungsi mxm2str(). Selanjutnya, vektor string/ekspresi hasilnya dapat digambar seperti menggambar vektor ekspresi pada EMT.

```
> plot2d("exp(x)",0,3); // plot fungsi aslinya, e^x
> plot2d(mxm2str("deret"),>add,color=4:6); // plot ketiga deret taylor hampiran fungsi ter
```

Selain cara di atas dapat juga dengan cara menggunakan indeks pada vektor/list yang dihasilkan.

```
> $deret[3]
> plot2d(["exp(x)",&deret[1],&deret[2],&deret[3]],0,3,color=1:4):
> $sum(sin(k*x)/k,k,1,5)
```

Berikut adalah cara menggambar kurva

$$y = \sin(x) + \frac{\sin 3x}{3} + \frac{\sin 5x}{5} + \dots$$

```
> plot2d(&sum(sin((2*k+1)*x)/(2*k+1),k,0,20),0,2pi):
```

Hal serupa juga dapat dilakukan dengan menggunakan matriks, misalkan kita akan menggambar kurva

$$y = \sum_{k=1}^{100} \frac{\sin(kx)}{k}, \quad 0 \leq x \leq 2\pi.$$

```
> x=linspace(0,2pi,1000); k=1:100; y=sum(sin(k*x')/k)'; plot2d(x,y):
```

Tabel Fungsi

Terdapat cara menarik untuk menghasilkan barisan dengan ekspresi Maxima. Perintah mxmtable() berguna untuk menampilkan dan menggambar barisan dan menghasilkan barisan sebagai vektor kolom.

Sebagai contoh berikut adalah barisan turunan ke-n x^n di $x=1$.

```
> mxmtable("diffat(x^n,x=1,n)","n",1,8,frac=1);
```

```
1  
2  
3  
8  
10  
54  
-42  
944
```

```
> $' sum(k, k, 1, n) = factor(ev(sum(k, k, 1, n),simpsum=true)) // simpsum:menghitung deret  
> $' sum(1/(3^k+k), k, 0, inf) = factor(ev(sum(1/(3^k+k), k, 0, inf),simpsum=true))
```

Di sini masih gagal, hasilnya tidak dihitung.

```
> $' sum(1/x^2, x, 1, inf)= ev(sum(1/x^2, x, 1, inf),simpsum=true) // ev: menghitung nilai  
> $' sum((-1)^(k-1)/k, k, 1, inf) = factor(ev(sum((-1)^(x-1)/x, x, 1, inf),simpsum=true))
```

Di sini masih gagal, hasilnya tidak dihitung.

```
> $' sum((-1)^k/(2*k-1), k, 1, inf) = factor(ev(sum((-1)^k/(2*k-1), k, 1, inf),simpsum=true)  
> $ev(sum(1/n!, n, 0, inf),simpsum=true)
```

Di sini masih gagal, hasilnya tidak dihitung, harusnya hasilnya e.

```
> &assume(abs(x)<1); \$' sum(a*x^k, k, 0, inf)=ev(sum(a*x^k, k, 0, inf),simpsum=true), &forg
```

Deret geometri tak hingga, dengan asumsi rasional antara -1 dan 1.

```
> \$' sum(x^k/k!,k,0,inf)=ev(sum(x^k/k!,k,0,inf),simpsum=true)
> $limit(sum(x^k/k!,k,0,n),n,inf)
> function d(n) &= sum(1/(k^2-k),k,2,n); \$'d(n)=d(n)
> $d(10)=ev(d(10),simpsum=true)
> $d(100)=ev(d(100),simpsum=true)
```

Deret Taylor

Deret Taylor suatu fungsi f yang diferensiabel sampai tak hingga di sekitar $x=a$ adalah:

$$f(x) = \sum_{k=0}^{\infty} \frac{(x-a)^k f^{(k)}(a)}{k!}.$$

```
> \$' e^x =taylor(exp(x),x,0,10) // deret Taylor e^x di sekitar x=0, sampai suku ke-11
> \$' log(x)=taylor(log(x),x,1,10)// deret log(x) di sekitar x=1
```

BAB 7

PENGGUNAAN SOFTWARE EMT UNTUK GEOMETRI

Visualisasi dan Perhitungan Geometri dengan EMT

Euler menyediakan beberapa fungsi untuk melakukan visualisasi dan perhitungan geometri, baik secara numerik maupun analitik (seperti biasanya tentunya, menggunakan Maxima). Fungsi-fungsi untuk visualisasi dan perhitungan geometri tersebut disimpan di dalam file program "geometry.e", sehingga file tersebut harus dipanggil sebelum menggunakan fungsi-fungsi atau perintah-perintah untuk geometri.

```
> load geometry
```

Numerical and symbolic geometry.

Fungsi-fungsi Geometri

Fungsi-fungsi untuk Menggambar Objek Geometri:

```
defaultd:=textheight()*1.5: nilai asli untuk parameter d  
setPlotrange(x1,x2,y1,y2): menentukan rentang x dan y pada bidang
```

koordinat

```
setPlotRange(r): pusat bidang koordinat (0,0) dan batas-batas  
sumbu-x dan y adalah -r sd r
```

```
plotPoint (P, "P"): menggambar titik P dan diberi label "P"  
plotSegment (A,B, "AB", d): menggambar ruas garis AB, diberi label
```

"AB" sejauh d

```

plotLine (g, "g", d): menggambar garis g diberi label "g" sejauh d
plotCircle (c,"c",v,d): Menggambar lingkaran c dan diberi label "c"
plotLabel (label, P, V, d): menuliskan label pada posisi P

```

Fungsi-fungsi Geometri Analitik (numerik maupun simbolik):

```

turn(v, phi): memutar vektor v sejauh phi
turnLeft(v): memutar vektor v ke kiri
turnRight(v): memutar vektor v ke kanan
normalize(v): normal vektor v
crossProduct(v, w): hasil kali silang vektorv dan w.
lineThrough(A, B): garis melalui A dan B, hasilnya [a,b,c] sdh.

```

$ax+by=c$.

```

lineWithDirection(A,v): garis melalui A searah vektor v
getLineDirection(g): vektor arah (gradien) garis g
getNormal(g): vektor normal (tegak lurus) garis g
getPointOnLine(g): titik pada garis g
perpendicular(A, g): garis melalui A tegak lurus garis g
parallel (A, g): garis melalui A sejajar garis g
lineIntersection(g, h): titik potong garis g dan h
projectToLine(A, g): proyeksi titik A pada garis g
distance(A, B): jarak titik A dan B
distanceSquared(A, B): kuadrat jarak A dan B
quadrance(A, B): kuadrat jarak A dan B
areaTriangle(A, B, C): luas segitiga ABC
computeAngle(A, B, C): besar sudut <ABC
angleBisector(A, B, C): garis bagi sudut <ABC
circleWithCenter (A, r): lingkaran dengan pusat A dan jari-jari r
getCircleCenter(c): pusat lingkaran c
getCircleRadius(c): jari-jari lingkaran c
circleThrough(A,B,C): lingkaran melalui A, B, C
middlePerpendicular(A, B): titik tengah AB
lineCircleIntersections(g, c): titik potong garis g dan lingkran c
circleCircleIntersections (c1, c2): titik potong lingkaran c1 dan

```

c2

```
planeThrough(A, B, C): bidang melalui titik A, B, C
```

Fungsi-fungsi Khusus Untuk Geometri Simbolik:

```

getLineEquation (g,x,y): persamaan garis g dinyatakan dalam x dan y
getHesseForm (g,x,y,A): bentuk Hesse garis g dinyatakan dalam x dan
y dengan titik A pada

```

```

sisi positif (kanan/atas) garis
quad(A,B): kuadrat jarak AB
spread(a,b,c): Spread segitiga dengan panjang sisi-sisi a,b,c, yakni

```

$\sin(\alpha)^2$ dengan

alpha sudut yang menghadap sisi a.

crosslaw(a,b,c,sa) : persamaan 3 quads dan 1 spread pada segitiga

dengan panjang sisi a, b, c.

triplespread(sa,sb,sc) : persamaan 3 spread sa, sb, sc yang memebntuk

suatu segitiga

doublespread(sa) : Spread sudut rangkap Spread 2ϕ , dengan

$sa = \sin(\phi)^2$ spread a.

Contoh 1: Luas, Lingkaran Luar, Lingkaran Dalam Segitiga

Untuk menggambar objek-objek geometri, langkah pertama adalah menentukan rentang sumbu-sumbu koordinat. Semua objek geometri akan digambar pada satu bidang koordinat, sampai didefinisikan bidang koordinat yang baru.

```
> setPlotRange(-0.5, 2.5, -0.5, 2.5); // mendefinisikan bidang koordinat baru
```

Sekarang tetapkan tiga titik dan plotkan.

```
> A=[1,0]; plotPoint(A, "A"); // definisi dan gambar tiga titik
> B=[0,1]; plotPoint(B, "B");
> C=[2,2]; plotPoint(C, "C");
```

Kemudian tiga segmen.

```
> plotSegment(A,B, "c"); // c=AB
> plotSegment(B,C, "a"); // a=BC
> plotSegment(A,C, "b"); // b=AC
```

Fungsi geometri mencakup fungsi untuk membuat garis dan lingkaran. Format untuk garis adalah [a,b,c], yang merepresentasikan garis dengan persamaan $ax+by=c$.

```
> lineThrough(B,C) // garis yang melalui B dan C
```

$[-1, 2, 2]$

Hitung garis tegak lurus yang melalui A pada BC.

```
> h=perpendicular(A,lineThrough(B,C)); // garis h tegak lurus BC melalui A
```

Dan persimpangannya dengan BC.

```
> D=lineIntersection(h,lineThrough(B,C)); // D adalah titik potong h dan BC
```

Plot itu.

```
> plotPoint(D,value=1); // koordinat D ditampilkan  
> aspect(1); plotSegment(A,D); // tampilkan semua gambar hasil plot...()
```

Hitung luas ABC:

$$L_{\triangle ABC} = \frac{1}{2} AD \cdot BC.$$

```
> norm(A-D)*norm(B-C)/2 // AD=norm(A-D), BC=norm(B-C)
```

1.5

Bandingkan dengan rumus determinan.

```
> areaTriangle(A,B,C) // hitung luas segitiga langsung dengan fungsi
```

1.5

Cara lain menghitung luas segitiga ABC:

```
> distance(A,D)*distance(B,C)/2
```

1.5

Sudut pada C.

```
> degprint(computeAngle(B,C,A))
```

36°52'11.63''

Sekarang, lingkarilah segitiga tersebut.

```
> c=circleThrough(A,B,C); // lingkaran luar segitiga ABC  
> R=getCircleRadius(c); // jari2 lingkaran luar  
> O=getCircleCenter(c); // titik pusat lingkaran c  
> plotPoint(O,"O"); // gambar titik "O"  
> plotCircle(c,"Lingkaran luar segitiga ABC");
```

Tampilkan koordinat titik pusat dan jari-jari lingkaran luar.

```
> O, R
```

```
[1.16667, 1.16667]  
1.17851130198
```

Sekarang akan digambar lingkaran dalam segitiga ABC. Titik pusat lingkaran dalam adalah titik potong garis-garis bagi sudut.

```
> l=angleBisector(A,C,B); // garis bagi <ACB  
> g=angleBisector(C,A,B); // garis bagi <CAB  
> P=lineIntersection(l,g) // titik potong kedua garis bagi sudut
```

```
[0.86038, 0.86038]
```

Tambahkan semuanya ke plot.

```
> color(5); plotLine(l); plotLine(g); color(1); // gambar kedua garis bagi sudut  
> plotPoint(P, "P"); // gambar titik potongnya  
> r=norm(P-projectToLine(P,lineThrough(A,B))) // jari-jari lingkaran dalam
```

```
0.509653732104
```

```
> plotCircle(circleWithCenter(P,r), "Lingkaran dalam segitiga ABC"); // gambar lingkaran dalam
```

Latihan 1

1. Tentukan ketiga titik singgung lingkaran dalam dengan sisi-sisi segitiga ABC.

```
> setPlotRange(-2.5,4.5,-2.5,4.5);  
> A=[-2,1]; plotPoint(A, "A");  
> B=[1,-2]; plotPoint(B, "B");  
> C=[4,4]; plotPoint(C, "C");
```

2. Gambar segitiga dengan titik-titik sudut ketiga titik singgung tersebut. Merupakan segitiga apakah itu?

```
> plotSegment(A,B, "c")  
> plotSegment(B,C, "a")  
> plotSegment(A,C, "b")  
> aspect(1):
```

Segitiga Sama Kaki

3. Hitung luas segitiga tersebut.

```
> areaTriangle(A, B, C)
```

13.5

4. Tunjukkan bahwa garis bagi sudut yang ke tiga juga melalui titik pusat lingkaran dalam.

```
> l=angleBisector(A,C,B);  
> g=angleBisector(C,A,B);  
> P=lineIntersection(l,g)
```

[0.581139, 0.581139]

```
> color(5); plotLine(l); plotLine(g); color(1);  
> plotPoint(P,"P");  
> r=norm(P-projectToLine(P,lineThrough(A,B)))
```

1.52896119631

```
> plotCircle(circleWithCenter(P,r),"Lingkaran dalam segitiga ABC"):
```

Jadi, terbukti bahwa garis bagi sudut yang ketiga juga melalui titik pusat lingkaran dalam.

4. Gambar jari-jari lingkaran dalam.

```
> r=norm(P-projectToLine(P,lineThrough(A,B)))
```

1.52896119631

```
> plotCircle(circleWithCenter(P,r),"Lingkaran dalam segitiga ABC"):
```

6. Hitung luas lingkaran luar dan luas lingkaran dalam segitiga ABC. Adakah hubungan antara luas kedua lingkaran tersebut dengan luas segitiga ABC?

Menghitung luas lingkaran luar m secara manual menggunakan rumus:

$$L = \pi \times r^2$$

Luas lingkaran dalam

```
> pi*(1.52896119631)^2
```

7.34417132892

```
> m=circleThrough(A,B,C);  
> R=getCircleRadius(m);  
> O=getCircleCenter(m);  
> plotPoint(O,"O");  
> color(2); plotCircle(m,"Lingkaran Luar m");
```

Jari-jari lingkaran luar

```
> O, R
```

```
[1.5, 1.5]  
3.53553390593
```

Luas lingkaran luar

```
> pi*(3.53553390593)^2
```

```
39.2699081698
```

Latihan 2

Kita akan menghitung luas segitiga ABC dan lingkaran luar ABC. Sebelum itu, kita perlu menggambarkan objek segitiga ABC pada bidang koordinat terlebih dahulu.

Berikut adalah langkah-langkah menggambarkan segitiga siku-siku.

Langkah 1: Menentukan luas bidang koordinat

Langkah 1: Menghitung panjang ruas garis BC sebagai alas segitiga

```
> setPlotRange(5);  
> setPlotRange(5);
```

Langkah 2: Menentukan titik-titik pada bidang koordinat

```
> A=[2,4]; plotPoint(A,"A");  
> B=[-3,0]; plotPoint(B,"B");  
> C=[2,0]; plotPoint(C,"C");
```

Langkah 3: Menggambarkan ruas garis pada titik A, B, dan C

```
> plotSegment(A,B,"c",-30);  
> plotSegment(B,C,"a");  
> plotSegment(A,C,"b");
```

Setelah menggambarkan objek geometri yang diinginkan, kita akan menghitung luas dari segitiga siku-siku tersebut menggunakan fungsi geometri yang terdapat pada Euler.

```
> areaTriangle(A,B,C)
```

10

Untuk membuktikan hitungan menggunakan fungsi geometri Euler bernilai benar, kita akan membandingkannya dengan hasil luas segitiga siku-siku yang dihitung secara manual.

Rumus Luas Segitiga:

$$L_{\triangle ABC} = \frac{1}{2} BC \cdot AC$$

```
> norm(B-C)
```

5

Langkah 2: Menghitung panjang ruas garis AC sebagai tinggi segitiga

```
> norm(A-C)
```

4

Langkah 3: Menghitung luas segitiga siku-siku menggunakan rumus luas segitiga

```
> norm(B-C) *norm(A-C) /2
```

10

Jadi, hasil dari perhitungan secara manual dan menggunakan fungsi areaTriangle(A,B,C) bernilai sama. Luas segitiga ABC bernilai 10 satuan.

Selanjutnya, kita akan menggambarkan suatu lingkaran yang mengelilingi segitiga siku-siku ABC. Lingkaran tersebut kita beri nama lingkaran luar m.

```
> m=circleThrough(A,B,C);  
> R=getCircleRadius(m);  
> O=getCircleCenter(m);  
> plotPoint(O,"O");  
> color(2); plotCircle(m,"Lingkaran Luar m");
```

Untuk menghitung luas lingkaran, kita akan mencari jari-jari dari lingkaran luar m.

```
> O, R
```

[-0.5, 2]
3.20156211872

Selanjutnya, kita akan menghitung luas lingkaran luar m secara manual menggunakan rumus:

$$L_m = \pi \times r^2$$

```
> pi*(3.20156211872)^2
```

32.2013246994

Contoh 2: Geometri Smbolik

Kita dapat menghitung geometri eksak dan simbolik menggunakan Maxima.

File geometri.e menyediakan fungsi-fungsi yang sama (dan lebih banyak lagi) di Maxima. Namun, kita dapat menggunakan komputasi simbolik sekarang.

```
> A &= [1,0]; B &= [0,1]; C &= [2,2]; // menentukan tiga titik A, B, C
```

Fungsi untuk garis dan lingkaran bekerja seperti fungsi Euler, tetapi menyediakan komputasi simbolis.

```
> c &= lineThrough(B,C) // c=BC
```

[- 1, 2, 2]

Kita bisa mendapatkan persamaan untuk sebuah garis dengan mudah.

```
> $getLineEquation(c,x,y), $solve(%,y) | expand // persamaan garis c
> $getLineEquation(lineThrough([x1,y1],[x2,y2]),x,y), $solve(%,y) // persamaan garis melalui A dan (x1, y1)
> $getLineEquation(lineThrough(A,[x1,y1]),x,y) // persamaan garis melalui A dan (x1, y1)
> h &= perpendicular(A,lineThrough(B,C)) // h melalui A tegak lurus BC
```

[2, 1, 2]

```
> Q &= lineIntersection(c,h); $Q // Q titik potong garis c=BC dan h
> $projectToLine(A,lineThrough(B,C)) // proyeksi A pada BC
> $distance(A,Q) // jarak AQ
> cc &= circleThrough(A,B,C); $cc // (titik pusat dan jari-jari) lingkaran melalui A, B, C
> r&=getCircleRadius(cc); $r , $float(r) // tampilkan nilai jari-jari
> $computeAngle(A,C,B) // nilai <ACB
> $solve(getLineEquation(angleBisector(A,C,B),x,y),y)[1] // persamaan garis bagi <ACB
> P &= lineIntersection(angleBisector(A,C,B),angleBisector(C,B,A)); $P // titik potong 2 garis
> P() // hasilnya sama dengan perhitungan sebelumnya
```

[0.86038, 0.86038]

Garis dan Lingkaran yang Berpotongan

Tentu saja, kita juga bisa memotong garis dengan lingkaran, dan lingkaran dengan lingkaran.

```
> A &:= [1,0]; c=circleWithCenter(A,4);
> B &:= [1,2]; C &:= [2,1]; l=lineThrough(B,C);
> setPlotRange(5); plotCircle(c); plotLine(l);
```

Perpotongan garis dengan lingkaran menghasilkan dua titik dan jumlah titik perpotongan.

```
> {P1,P2,f}=lineCircleIntersections(l,c);
> P1, P2, f
```

```
[4.64575, -1.64575]
[-0.645751, 3.64575]
2
```

```
> plotPoint(P1); plotPoint(P2);
```

Hal yang sama pada Maxima.

```
> c &= circleWithCenter(A,4) // lingkaran dengan pusat A jari-jari 4
```

```
[1, 0, 4]
```

```
> l &= lineThrough(B,C) // garis l melalui B dan C
```

```
[1, 1, 3]
```

```
> $lineCircleIntersections(l,c) | radcan, // titik potong lingkaran c dan garis l
```

Akan ditunjukkan bahwa sudut-sudut yang menghadap bsuusr yang sama adalah sama besar.

```
> C=A+normalize([-2,-3])*4; plotPoint(C); plotSegment(P1,C); plotSegment(P2,C);
> degprint(computeAngle(P1,C,P2))
```

```
69°17'42.68''
```

```
> C=A+normalize([-4,-3])*4; plotPoint(C); plotSegment(P1,C); plotSegment(P2,C);
> degprint(computeAngle(P1,C,P2))
```

```
69°17'42.68''
```

```
> insimg;
```

Garis Sumbu

Berikut adalah langkah-langkah menggambar garis sumbu ruas garis AB:

1. Gambar lingkaran dengan pusat A melalui B.
2. Gambar lingkaran dengan pusat B melalui A.
3. Tarik garis melalui kedua titik potong kedua lingkaran tersebut. Garis ini merupakan garis sumbu (melalui titik tengah dan tegak lurus) AB.

```
> A=[2,2]; B=[-1,-2];
> c1=circleWithCenter(A,distance(A,B));
> c2=circleWithCenter(B,distance(A,B));
> {P1,P2,f}=circleCircleIntersections(c1,c2);
> l=lineThrough(P1,P2);
> setPlotRange(5); plotCircle(c1); plotCircle(c2);
> plotPoint(A); plotPoint(B); plotSegment(A,B); plotLine(l):
```

Selanjutnya, kami melakukan hal yang sama di Maxima dengan koordinat umum.

```
> A &= [a1,a2]; B &= [b1,b2];
> c1 &= circleWithCenter(A,distance(A,B));
> c2 &= circleWithCenter(B,distance(A,B));
> P &= circleCircleIntersections(c1,c2); P1 &= P[1]; P2 &= P[2];
```

Persamaan untuk persimpangan cukup rumit. Tetapi kita dapat menyederhanakannya, jika kita menyelesaikan untuk y.

```
> g &= getLineEquation(lineThrough(P1,P2),x,y);
> $solve(g,y)
```

Ini memang sama dengan tegak lurus tengah, yang dihitung dengan cara yang sama sekali berbeda.

```
> $solve(getLineEquation(middlePerpendicular(A,B),x,y),y)
> h &= getLineEquation(lineThrough(A,B),x,y);
> $solve(h,y)
```

Perhatikan hasil kali gradien garis g dan h adalah:

$$\frac{-(b_1 - a_1)}{(b_2 - a_2)} \times \frac{(b_2 - a_2)}{(b_1 - a_1)} = -1.$$

Artinya kedua garis tegak lurus.

Contoh 3: Rumus Heron

Rumus Heron menyatakan bahwa luas segitiga dengan panjang sisi-sisi a , b dan c adalah:

$$L = \sqrt{s(s-a)(s-b)(s-c)} \quad \text{dengan } s = (a+b+c)/2,$$

atau bisa ditulis dalam bentuk lain:

$$L = \frac{1}{4} \sqrt{(a+b+c)(b+c-a)(a+c-b)(a+b-c)}$$

Untuk membuktikan hal ini kita misalkan $C(0,0)$, $B(a,0)$ dan $A(x,y)$, $b=AC$, $c=AB$. Luas segitiga ABC adalah

$$L_{\triangle ABC} = \frac{1}{2}a \times y.$$

Nilai y didapat dengan menyelesaikan sistem persamaan:

$$x^2 + y^2 = b^2, \quad (x-a)^2 + y^2 = c^2.$$

```
> setPlotRange(-1,10,-1,8); plotPoint([0,0], "C(0,0)"); plotPoint([5.5,0], "B(a,0)"); ...
> plotPoint([7.5,6], "A(x,y)");
> plotSegment([0,0],[5.5,0], "a",25); plotSegment([5.5,0],[7.5,6],"c",15); ...
> plotSegment([0,0],[7.5,6],"b",25);
> plotSegment([7.5,6],[7.5,0],"t=y",25):
> &assume(a>0); sol &= solve([x^2+y^2=b^2, (x-a)^2+y^2=c^2], [x,y])
```

$$\begin{aligned} & \left[\begin{aligned} & x = \frac{-c^2 + b^2 + a^2}{2a}, \quad y = \\ & \frac{\sqrt{-c^4 + 2b^2c^2 + 2a^2c^2 - b^4 + 2a^2b^2 - a^4}}{2a} \\ & - \frac{\sqrt{-c^4 + 2b^2c^2 + 2a^2c^2 - b^4 + 2a^2b^2 - a^4}}{2a} \end{aligned} \right], \\ & \left[\begin{aligned} & x = \frac{-c^2 + b^2 + a^2}{2a}, \quad y = \\ & \frac{\sqrt{-c^4 + 2b^2c^2 + 2a^2c^2 - b^4 + 2a^2b^2 - a^4}}{2a} \end{aligned} \right] \end{aligned}$$

Ekstrak larutan y .

```
> ysol &= y with sol[2][2]; \$'y=sqrt(factor(ysol^2))
```

Kami mendapatkan rumus Heron.

```
> function H(a,b,c) &= sqrt(factor((ysol*a/2)^2)); \$'H(a,b,c)=H(a,b,c)
> \$'Luas=H(2,5,6) // luas segitiga dengan panjang sisi-sisi 2, 5, 6
```

Tentu saja, setiap segitiga persegi panjang adalah kasus yang terkenal.

```
> H(3,4,5) //luas segitiga siku-siku dengan panjang sisi 3, 4, 5
```

6

Dan juga jelas, bahwa ini adalah segitiga dengan luas maksimal dan kedua sisi 3 dan 4.

```
> aspect(1.5); plot2d(&H(3,4,x),1,7); // Kurva luas segitiga sengan panjang sisi 3, 4, x
```

Kasus umum juga bisa digunakan.

```
> $solve(diff(H(a,b,c)^2,c)=0,c)
```

Sekarang mari kita cari himpunan semua titik di mana $b+c=d$ untuk suatu konstanta d . Sudah diketahui bahwa ini adalah sebuah ellips.

```
> s1 &= subst(d-c,b,sol[2]); $s1
```

Dan membuat fungsi-fungsi ini.

```
> function fx(a,c,d) &= rhs(s1[1]); $fx(a,c,d), function fy(a,c,d) &= rhs(s1[2]); $fy(a,c,d)
```

Sekarang kita dapat menggambar himpunan tersebut. Sisi b bervariasi dari 1 hingga 4. Sudah diketahui bahwa kita mendapatkan sebuah ellips.

```
> aspect(1); plot2d(&fx(3,x,5),&fy(3,x,5),xmin=1,xmax=4,square=1);
```

Kita dapat memeriksa persamaan umum untuk ellips ini, yaitu

$$\frac{(x - x_m)^2}{u^2} + \frac{(y - y_m)^2}{v^2} = 1,$$

di mana (x_m, y_m) adalah pusat, dan u serta v adalah setengah sumbu.

```
> $ratsimp((fx(a,c,d)-a/2)^2/u^2+fy(a,c,d)^2/v^2 with [u=d/2,v=sqrt(d^2-a^2)/2])
```

Kita melihat bahwa tinggi dan luas segitiga adalah maksimal untuk $x=0$. Dengan demikian, luas segitiga dengan $a+b+c=d$ adalah maksimal, jika segitiga tersebut sama sisi. Kita ingin membuktikannya secara analitis.

```
> eqns &= [diff(H(a,b,d-(a+b))^2,a)=0, diff(H(a,b,d-(a+b))^2,b)=0]; $eqns
```

Kita mendapatkan beberapa minima, yang termasuk dalam segitiga dengan satu sisi 0, dan solusi $a = b = c = d / 3$.

```
> $solve(eqns, [a,b])
```

Ada juga metode Lagrange, yang memaksimalkan $H(a,b,c)^2$ sehubungan dengan $a+b+d=d$.

```
> &solve([diff(H(a,b,c)^2,a)=la,diff(H(a,b,c)^2,b)=la, ...
>         diff(H(a,b,c)^2,c)=la,a+b+c=d],[a,b,c,la])
```

$$\begin{aligned} & \left[\left[a = 0, \frac{d}{2}, \frac{d}{2}, \frac{d}{2}, la = 0 \right], \left[a = -\frac{d}{2}, 0, \frac{d}{2}, \frac{d}{2}, la = 0 \right], \right. \\ & \left. \left[a = -\frac{d}{2}, -\frac{d}{2}, 0, la = 0 \right], \left[a = -\frac{d}{3}, -\frac{d}{3}, -\frac{d}{3}, la = \frac{-d}{108} \right] \right] \end{aligned}$$

Kita bisa membuat plot situasi

Pertama-tama, tetapkan titik-titik di Maxima.

```
> A &= at([x,y],sol[2]); $A
> B &= [0,0]; $B, C &= [a,0]; $C
```

Kemudian, tetapkan kisaran plot, dan plot titik-titiknya.

```
> setPlotRange(0,5,-2,3); ...
> a=4; b=3; c=2; ...
> plotPoint(mxmeval("B"),"B"); plotPoint(mxmeval("C"),"C"); ...
> plotPoint(mxmeval("A"),"A"):
```

Plot segmen-segmen tersebut.

```
> plotSegment(mxmeval("A"),mxmeval("C")); ...
> plotSegment(mxmeval("B"),mxmeval("C")); ...
> plotSegment(mxmeval("B"),mxmeval("A")):
```

Hitung garis tegak lurus tengah dalam Maxima.

```
> h &= middlePerpendicular(A,B); g &= middlePerpendicular(B,C);
```

Dan bagian tengah lingkar.

```
> U &= lineIntersection(h,g);
```

Kita mendapatkan rumus untuk jari-jari lingkaran.

```
> &assume(a>0,b>0,c>0); $distance(U,B) | radcan
```

Mari kita tambahkan ini ke dalam plot.

```
> plotPoint(U()); ...
> plotCircle(circleWithCenter(mxmeval("U")),mxmeval("distance(U,C)")):
```

Dengan menggunakan geometri, kami memperoleh rumus sederhana

$$\frac{a}{\sin(\alpha)} = 2r$$

untuk radius. Kita bisa mengecek, apakah hal ini benar dengan Maxima. Maxima akan memperhitungkannya hanya jika kita mengkuadratkannya.

```
> $c^2/sin(computeAngle(A,B,C))^2 | factor
```

Contoh 4: Garis Euler dan Parabola

Garis Euler adalah garis yang ditentukan dari segitiga apa pun yang tidak sama sisi. Garis ini merupakan garis tengah segitiga, dan melewati beberapa titik penting yang ditentukan dari segitiga, termasuk ortosentrum, circumcentrum, centroid, titik Exeter, dan pusat lingkaran sembilan titik segitiga.

Sebagai demonstrasi, kami menghitung dan memplot garis Euler dalam sebuah segitiga.

Pertama, kita mendefinisikan sudut-sudut segitiga dalam Euler. Kami menggunakan definisi, yang terlihat dalam ekspresi simbolis.

```
> A:=[-1,-1]; B:=[2,0]; C:=[1,2];
```

Untuk memplot objek geometris, kita menyiapkan area plot, dan menambahkan titik-titiknya. Semua plot objek geometris ditambahkan ke plot saat ini.

```
> setPlotRange(3); plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C");
```

Kita juga bisa menambahkan sisi-sisi segitiga.

```
> plotSegment(A,B,""); plotSegment(B,C,""); plotSegment(C,A,"");
```

Berikut ini adalah luas area segitiga, dengan menggunakan rumus determinan. Tentu saja, kita harus mengambil nilai absolut dari hasil ini.

```
> $areaTriangle(A,B,C)
```

Kita dapat menghitung koefisien dari sisi c.

```
> c &= lineThrough(A,B)
```

```
[- 1, 3, - 2]
```

Dan juga mendapatkan formula untuk baris ini.

```
> $getLineEquation(c,x,y)
```

Untuk bentuk Hesse, kita perlu menentukan sebuah titik, sehingga titik tersebut berada di sisi positif dari bentuk Hesse. Memasukkan titik tersebut akan menghasilkan jarak positif ke garis.

```
> $getHesseForm(c,x,y,C), $at(%,[x=C[1],y=C[2]])
```

Sekarang kita menghitung keliling ABC.

```
> LL &= circleThrough(A,B,C); $getCircleEquation(LL,x,y)
> O &= getCircleCenter(LL); $O
```

Plot lingkaran dan pusatnya. Cu dan U adalah simbolik. Kami mengevaluasi ekspresi ini untuk Euler.

```
> plotCircle(LL()); plotPoint(O(),"O"):
```

Kita dapat menghitung perpotongan ketinggian di ABC (pusat ortosentrum) secara numerik dengan perintah berikut ini.

```
> H &= lineIntersection(perpendicular(A,lineThrough(C,B)),...
> perpendicular(B,lineThrough(A,C))); $H
```

Sekarang kita dapat menghitung garis Euler dari segitiga tersebut.

```
> el &= lineThrough(H,O); $getLineEquation(el,x,y)
```

Tambahkan ke plot kami.

```
> plotPoint(H(), "H"); plotLine(el(), "Garis Euler");
```

Pusat gravitasi harus berada pada garis ini.

```
> M &= (A+B+C)/3; $getLineEquation(el,x,y) with [x=M[1],y=M[2]]  
> plotPoint(M(), "M"); // titik berat
```

Teori mengatakan bahwa $MH = 2*MO$. Kita perlu menyederhanakan dengan radcan untuk mencapai hal ini.

```
> $distance(M,H)/$distance(M,O) | radcan
```

Fungsi-fungsi ini juga mencakup fungsi untuk sudut.

```
> $computeAngle(A,C,B), degprint(%())
```

$60^\circ 15' 18.43''$

Persamaan untuk bagian tengah lingkaran tidak terlalu bagus.

```
> Q &= lineIntersection(angleBisector(A,C,B), angleBisector(C,B,A)) | radcan; $Q
```

Mari kita hitung juga ekspresi untuk jari-jari lingkaran yang tertulis.

```
> r &= distance(Q,projectToLine(Q,lineThrough(A,B))) | ratsimp; $r  
> LD &= circleWithCenter(Q,r); // Lingkaran dalam
```

Mari kita tambahkan ini ke dalam plot.

```
> color(5); plotCircle(LD());
```

Parabola

Selanjutnya akan dicari persamaan tempat kedudukan titik-titik yang berjarak sama ke titik C dan ke garis AB.

```
> p &= getHesseForm(lineThrough(A,B),x,y,C)-distance([x,y],C); $p='0
```

Persamaan tersebut dapat digambar menjadi satu dengan gambar sebelumnya.

```
> plot2d(p, level=0, add=1, contourcolor=6) :
```

Ini seharusnya merupakan suatu fungsi, tetapi solver default Maxima hanya bisa menemukan solusinya jika kita mengkuadratkan persamaannya. Akibatnya, kita mendapatkan solusi palsu.

```
> akar &= solve(getHesseForm(lineThrough(A,B),x,y,C)^2-distance([x,y],C)^2,y)
```

$$\begin{aligned}y &= -3x - \sqrt{70} \sqrt{9 - 2x} + 26, \\y &= -3x + \sqrt{70} \sqrt{9 - 2x} + 26\end{aligned}$$

Solusi pertama adalah

maxima: akar[1]

Menambahkan solusi pertama ke dalam plot menunjukkan, bahwa itu memang jalur yang kita cari. Teori mengatakan bahwa ini adalah parabola yang diputar.

```
> plot2d(&rhs(akar[1]),add=1):  
> function g(x) &= rhs(akar[1]); $'g(x)=g(x)// fungsi yang mendefinisikan kurva di atas  
> T &=[-1, g(-1)]; // ambil sebarang titik pada kurva tersebut  
> dTC &= distance(T,C); $fullratsimp(dTC), $float(%) // jarak T ke C  
> U &= projectToLine(T,lineThrough(A,B)); $U // proyeksi T pada garis AB  
> dU2AB &= distance(T,U); $fullratsimp(dU2AB), $float(%) // jarak T ke AB
```

Ternyata jarak T ke C sama dengan jarak T ke AB. Coba Anda pilih titik T yang lain dan ulangi perhitungan-perhitungan di atas untuk menunjukkan bahwa hasilnya juga sama.

Contoh 5: Trigonometri Rasional

Ini terinspirasi dari sebuah ceramah N.J. Wildberger. Dalam bukunya "Proporsi Ilahi", Wildberger mengusulkan untuk mengganti gagasan klasik tentang jarak dan sudut dengan kuadransi dan penyebaran. Dengan menggunakan ini, memang memungkinkan untuk menghindari fungsi trigonometri dalam banyak contoh, dan tetap "rasional".

Berikut ini, saya akan memperkenalkan konsep-konsep tersebut, dan memecahkan beberapa masalah. Saya menggunakan komputasi simbolik Maxima di sini, yang menyembunyikan keuntungan utama dari trigonometri rasional yaitu komputasi dapat dilakukan dengan kertas dan pensil saja. Anda dipersilakan untuk memeriksa hasilnya tanpa komputer.

Intinya adalah bahwa komputasi rasional simbolik sering kali memberikan hasil yang sederhana. Sebaliknya, trigonometri klasik menghasilkan hasil trigonometri yang rumit, yang dievaluasi dengan pendekatan numerik saja.

```
> load geometry;
```

Untuk pengenalan pertama, kita menggunakan segitiga persegi panjang dengan proporsi Mesir yang terkenal 3, 4, dan 5. Perintah berikut ini adalah perintah Euler untuk memplot geometri bidang yang terdapat pada file Euler "geometry.e".

```

> C:=[0,0]; A:=[4,0]; B:=[0,3]; ...
> setPlotRange(-1,5,-1,5); ...
> plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C"); ...
> plotSegment(B,A,"c"); plotSegment(A,C,"b"); plotSegment(C,B,"a"); ...
> insimg(30);

```

Tentu saja,

$$\sin(w_a) = \frac{a}{c},$$

di mana w_a adalah sudut di A. Cara biasa untuk menghitung sudut ini, adalah dengan mengambil kebalikan dari fungsi sinus. Hasilnya adalah sudut yang tidak dapat dicerna, yang hanya dapat dicetak kira-kira.

```

> wa := arcsin(3/5); degprint(wa)

```

$36^\circ 52' 11.63''$

Trigonometri rasional mencoba menghindari hal ini.

Gagasan pertama trigonometri rasional adalah kuadrat, yang menggantikan jarak. Sebenarnya, ini hanyalah jarak yang dikuadratkan. Berikut ini, a, b, dan c menunjukkan kuadran sisi-sisinya.

Teorema Pythagoras menjadi $a + b = c$.

```

> a &= 3^2; b &= 4^2; c &= 5^2; &a+b=c

```

$$25 = 25$$

Gagasan kedua dari trigonometri rasional adalah penyebaran. Penyebaran mengukur bukaan di antara garis-garis. Ini adalah 0, jika garis-garisnya sejajar, dan 1, jika garis-garisnya persegi panjang. Ini adalah kuadrat dari sinus sudut antara dua garis.

Penyebaran garis AB dan AC pada gambar di atas didefinisikan sebagai

$$s_a = \sin(\alpha)^2 = \frac{a}{c},$$

di mana a dan c adalah kuadran dari segitiga persegi panjang dengan satu sudut di A.

```

> sa &= a/c; $sa

```

Tentu saja, hal ini lebih mudah dihitung daripada sudut. Tetapi Anda kehilangan sifat bahwa sudut dapat ditambahkan dengan mudah.

Tentu saja, kita bisa mengonversi nilai perkiraan kita untuk sudut wa ke sprad, dan mencetaknya sebagai pecahan.

```

> fracprint(sin(wa)^2)

```

Hukum kosinus trigonometri klasik diterjemahkan ke dalam "hukum silang" berikut ini.

$$(c + b - a)^2 = 4bc(1 - s_a)$$

Di sini a , b , dan c adalah kuadran dari sisi-sisi segitiga, dan s_a adalah penyebaran di sudut A . Sisi a , seperti biasa, berlawanan dengan sudut A .

Hukum-hukum ini diimplementasikan dalam file `geometri.e` yang kita masukkan ke dalam Euler.

```
> $crosslaw(aa,bb,cc,saa)
```

Dalam kasus kami, kami mendapatkan

```
> $crosslaw(a,b,c,sa)
```

Mari kita gunakan `crosslaw` ini untuk mencari sebaran di A . Untuk melakukannya, kita buat `crosslaw` untuk kuadran a , b , dan c , dan selesaikan untuk sebaran sa yang tidak diketahui.

Anda bisa melakukan ini dengan tangan dengan mudah, tapi saya menggunakan Maxima. Tentu saja, kita mendapatkan hasil yang sudah kita dapatkan.

```
> $crosslaw(a,b,c,x), $solve(% ,x)
```

Kita sudah mengetahui hal ini. Definisi penyebaran adalah kasus khusus dari `crosslaw`.

Kita juga dapat menyelesaiannya untuk a , b , c secara umum. Hasilnya adalah sebuah rumus yang menghitung penyebaran sudut sebuah segitiga dengan kuadran ketiga sisinya.

```
> $solve(crosslaw(aa,bb,cc,x),x)
```

Kita dapat membuat sebuah fungsi dari hasil tersebut. Fungsi seperti itu sudah didefinisikan dalam file `geometry.e` dari Euler.

```
> $spread(a,b,c)
```

Sebagai contoh, kita dapat menggunakannya untuk menghitung sudut segitiga dengan sisi

$$a, \quad a, \quad \frac{4a}{7}$$

Hasilnya adalah rasional, yang tidak mudah didapat jika kita menggunakan trigonometri klasik.

```
> $spread(a,a,4*a/7)
```

Ini adalah sudut dalam derajat.

```
> degprint(arcsin(sqrt(6/7)))
```

67°47'32.44''

Contoh Lain

Sekarang, mari kita coba contoh yang lebih lanjut.

Kita tentukan tiga sudut segitiga sebagai berikut.

```
> A&:=[1,2]; B&:=[4,3]; C&:=[0,4]; ...
> setPlotRange(-1,5,1,7); ...
> plotPoint(A, "A"); plotPoint(B, "B"); plotPoint(C, "C"); ...
> plotSegment(B,A,"c"); plotSegment(A,C,"b"); plotSegment(C,B,"a"); ...
> insimg;
```

Dengan menggunakan Pythagoras, mudah untuk menghitung jarak antara dua titik. Pertama-tama saya menggunakan jarak fungsi dari file Euler untuk geometri. Jarak fungsi menggunakan geometri klasik.

```
> $distance(A,B)
```

Euler juga memiliki fungsi untuk kuadranan antara dua titik.

Pada contoh berikut, karena $c+b$ bukan a , maka segitiga tersebut tidak berbentuk persegi panjang.

```
> c &= quad(A,B); $c, b &= quad(A,C); $b, a &= quad(B,C); $a,
```

Pertama, mari kita menghitung sudut tradisional. Fungsi computeAngle menggunakan metode yang biasa berdasarkan hasil kali titik dari dua vektor. Hasilnya adalah beberapa perkiraan titik mengambang.

$$A = \langle 1, 2 \rangle \quad B = \langle 4, 3 \rangle, \quad C = \langle 0, 4 \rangle$$

$$\mathbf{a} = C - B = \langle -4, 1 \rangle, \quad \mathbf{c} = A - B = \langle -3, -1 \rangle, \quad \beta = \angle ABC$$

$$\mathbf{a} \cdot \mathbf{c} = |\mathbf{a}| \cdot |\mathbf{c}| \cos \beta$$

$$\cos \angle ABC = \cos \beta = \frac{\mathbf{a} \cdot \mathbf{c}}{|\mathbf{a}| \cdot |\mathbf{c}|} = \frac{12 - 1}{\sqrt{17} \sqrt{10}} = \frac{11}{\sqrt{17} \sqrt{10}}$$

```
> wb &= computeAngle(A,B,C); $wb, $(wb/pi*180)()
```

32.4711922908

Dengan menggunakan pensil dan kertas, kita dapat melakukan hal yang sama dengan hukum silang. Kita masukkan kuadran a, b, dan c ke dalam hukum silang dan selesaikan untuk x.

```
> $crosslaw(a,b,c,x), $solve(% ,x), // (b+c-a)^=4b.c(1-x)
```

Itulah yang dilakukan oleh fungsi spread yang didefinisikan dalam "geometry.e".

```
> sb &= spread(b,a,c); $sb
```

Maxima mendapatkan hasil yang sama dengan menggunakan trigonometri biasa, jika kita memaksakannya. Ia menyelesaikan suku $\sin(\arccos(...))$ menjadi hasil pecahan. Sebagian besar siswa tidak dapat melakukan ini.

```
> $sin(computeAngle(A,B,C))^2
```

Setelah kita memiliki penyebaran di B, kita dapat menghitung tinggi ha di sisi a. Ingatlah bahwa

$$s_b = \frac{h_a}{c}$$

menurut definisi.

```
> ha &= c*sb; $ha
```

Gambar berikut ini dibuat dengan program geometri C.a.R., yang dapat menggambar kuadran dan penyebaran.

image: (20) Rational_Geometry_CaR.png

Menurut definisi, panjang ha adalah akar kuadrat dari kuadrannya.

```
> $sqrt(ha)
```

Sekarang kita dapat menghitung luas segitiga. Jangan lupa, bahwa kita berurusan dengan kuadran!

```
> $sqrt(ha)*sqrt(a)/2
```

Rumus penentu yang biasa menghasilkan hasil yang sama.

```
> $areaTriangle(B,A,C)
```

Rumus Bangau

Sekarang, mari kita selesaikan masalah ini secara umum!

```
> &remvalue(a,b,c,sb,ha);
```

Pertama-tama kita menghitung penyebaran di B untuk segitiga dengan sisi a, b, dan c. Kemudian kita menghitung luas kuadrat ("quadrea"?), memfaktorkannya dengan Maxima, dan kita mendapatkan rumus Heron yang terkenal.

Memang, hal ini sulit dilakukan dengan pensil dan kertas.

```
> $spread(b^2,c^2,a^2), $factor(%*c^2*a^2/4)
```

Aturan Triple Spread

Kerugian dari spread adalah bahwa mereka tidak lagi hanya menambahkan sudut seperti. Namun, tiga spread dari sebuah segitiga memenuhi aturan "triple spread" berikut ini.

```
> &remvalue(sa,sb,sc); $triplespread(sa,sb,sc)
```

Aturan ini berlaku untuk tiga sudut yang berjumlah 180° .

$$\alpha + \beta + \gamma = \pi$$

Karena spread dari

$$\alpha, \pi - \alpha$$

adalah sama, aturan triple spread juga benar, jika

$$\alpha + \beta = \gamma$$

Karena penyebaran sudut negatif adalah sama, aturan triple spread juga berlaku, jika

$$\alpha + \beta + \gamma = 0$$

Sebagai contoh, kita bisa menghitung penyebaran sudut 60° . Ini adalah $3/4$. Namun, persamaan ini memiliki solusi kedua, di mana semua penyebarannya adalah 0.

```
> $solve(triplespread(x,x,x),x)
```

Penyebaran 90° jelas adalah 1. Jika dua sudut ditambahkan ke 90° , penyebarannya akan menyelesaikan persamaan penyebaran tiga dengan a, b, 1. Dengan perhitungan berikut, kita mendapatkan $a + b = 1$.

```
> $triplespread(x,y,1), $solve(%,x)
```

Karena penyebaran $180^\circ - t$ sama dengan penyebaran t , rumus penyebaran tiga kali lipat juga berlaku, jika satu sudut adalah jumlah atau selisih dari dua sudut lainnya.

Jadi kita dapat menemukan penyebaran sudut dua kali lipat. Perhatikan bahwa ada dua solusi lagi. Kita jadikan ini sebuah fungsi.

```
> $solve(triplespread(a,a,x),x), function doublespread(a) &= factor(rhs(%[1]))
```

$$- 4(a - 1)a$$

Pembagi Sudut

Ini adalah situasi yang sudah kita ketahui.

```
> C:=[0,0]; A:=[4,0]; B:=[0,3]; ...
> setPlotRange(-1,5,-1,5); ...
> plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C"); ...
> plotSegment(B,A,"c"); plotSegment(A,C,"b"); plotSegment(C,B,"a"); ...
> insimg;
```

Mari kita hitung panjang garis bagi sudut di A. Tetapi kita ingin menyelesaikannya untuk a, b, c secara umum.

```
> &remvalue(a,b,c);
```

Jadi, pertama-tama kita menghitung penyebaran sudut yang dibelah dua di A, menggunakan rumus penyebaran tiga.

Masalah dengan rumus ini muncul lagi. Rumus ini memiliki dua solusi. Kita harus memilih salah satu yang benar. Solusi lainnya mengacu pada sudut terbagi dua $180^\circ - wa$.

```
> $triplespread(x,x,a/(a+b)), $solve(% ,x), sa2 &= rhs(%[1]); $sa2
```

Mari kita periksa persegi panjang Mesir.

```
> $sa2 with [a=3^2,b=4^2]
```

Kita bisa mencetak sudut dalam Euler, setelah mentransfer penyebaran ke radian.

```
> wa2 := arcsin(sqrt(1/10)); degprint(wa2)
```

$$18^\circ 26' 5.82''$$

Titik P adalah perpotongan garis bagi sudut dengan sumbu y.

```
> P := [0, tan(wa2)*4]
```

```
[0, 1.33333]
```

```
> plotPoint(P, "P"); plotSegment(A, P);
```

Mari kita periksa sudut-sudutnya dalam contoh spesifik kita.

```
> computeAngle(C, A, P), computeAngle(P, A, B)
```

```
0.321750554397  
0.321750554397
```

Sekarang kita hitung panjang garis bagi AP.

Kita menggunakan teorema sinus dalam segitiga APC. Teorema ini menyatakan bahwa

$$\frac{BC}{\sin(w_a)} = \frac{AC}{\sin(w_b)} = \frac{AB}{\sin(w_c)}$$

berlaku untuk semua segitiga. Kuadratkan, ini diterjemahkan ke dalam apa yang disebut "hukum penyebaran"

$$\frac{a}{s_a} = \frac{b}{s_b} = \frac{c}{s_b}$$

di mana a, b, c menunjukkan qudrance.

Karena spread CPA adalah $1 - sa^2$, kita mendapatkan $bisa / 1 = b / (1 - sa^2)$ dan bisa menghitung bisa (kuadransi dari garis-bagi sudut).

```
> &factor(ratsimp(b / (1 - sa2))); bisa &= %; $bisa
```

Mari kita periksa rumus ini untuk nilai-nilai Egyptian kita.

```
> sqrt(mxmeval("at(bisa, [a=3^2, b=4^2])")), distance(A, P)
```

```
4.21637021356  
4.21637021356
```

Kita juga dapat menghitung P dengan menggunakan rumus penyebaran.

```
> py&=factor(ratsimp(sa2*bisa)); $py
```

Nilainya sama dengan yang kita dapatkan dengan rumus trigonometri.

```
> sqrt(mxmeval("at (py, [a=3^2,b=4^2])"))
```

1.33333333333

Sudut Akor

Lihatlah situasi berikut ini.

```
> setPlotRange(1.2); ...
> color(1); plotCircle(circleWithCenter([0,0],1)); ...
> A:=[cos(1),sin(1)]; B:=[cos(2),sin(2)]; C:=[cos(6),sin(6)]; ...
> plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C"); ...
> color(3); plotSegment(A,B,"c"); plotSegment(A,C,"b"); plotSegment(C,B,"a"); ...
> color(1); O:=[0,0]; plotPoint(O,"O"); ...
> plotSegment(A,O); plotSegment(B,O); plotSegment(C,O,"r"); ...
> insimg;
```

Kita dapat menggunakan Maxima untuk menyelesaikan rumus penyebaran tiga untuk sudut-sudut di pusat O untuk r. Dengan demikian kita mendapatkan rumus untuk jari-jari kuadrat dari percircle dalam hal kuadran sisi-sisinya.

Kali ini, Maxima menghasilkan beberapa angka nol yang rumit, yang kita abaikan.

```
> &remvalue(a,b,c,r); // hapus nilai-nilai sebelumnya untuk perhitungan baru
> rabc &= rhs(solve(triplespread(spread(b,r,r),spread(a,r,r),spread(c,r,r)),r)[4]); $rab
```

Kita dapat menjadikannya sebuah fungsi Euler.

```
> function periradius(a,b,c) &= rabc;
```

Mari kita periksa hasilnya untuk poin A, B, C.

```
> a:=quadrance(B,C); b:=quadrance(A,C); c:=quadrance(A,B);
```

Radiusnya memang 1.

```
> periradius(a,b,c)
```

1

Faktanya adalah, bahwa penyebaran CBA hanya bergantung pada b dan c. Ini adalah teorema sudut akor.

```
> $spread(b,a,c)*rabc | ratsimp
```

Faktanya, penyebarannya adalah $b/(4r)$, dan kita melihat bahwa sudut chord b adalah setengah dari sudut tengah.

```
> $doublespread(b/(4*r))-spread(b,r,r) | ratsimp
```

Contoh 6: Jarak Minimal pada Bidang

Keterangan awal

Fungsi yang, pada sebuah titik M pada bidang, menetapkan jarak AM antara titik tetap A dan M, memiliki garis-garis tingkat yang cukup sederhana: lingkaran yang berpusat di A.

```
> &remvalue();
> A=[-1,-1];
> function d1(x,y):=sqrt((x-A[1])^2+(y-A[2])^2)
> fcontour("d1",xmin=-2,xmax=0,ymin=-2,ymax=0,hue=1, ...
> title="If you see ellipses, please set your window square"):
```

dan grafiknya juga cukup sederhana: bagian atas kerucut:

```
> plot3d("d1",xmin=-2,xmax=0,ymin=-2,ymax=0):
```

Tentu saja nilai minimum 0 diperoleh dalam A.

Dua titik

Sekarang kita lihat fungsi MA+MB di mana A dan B adalah dua titik (tetap). Ini adalah "fakta yang terkenal" bahwa kurva tingkat adalah elips, titik fokusnya adalah A dan B; kecuali AB minimum yang konstan pada segmen [AB]:

```
> B=[1,-1];
> function d2(x,y):=d1(x,y)+sqrt((x-B[1])^2+(y-B[2])^2)
> fcontour("d2",xmin=-2,xmax=2,ymin=-3,ymax=1,hue=1):
```

Grafiknya lebih menarik:

```
> plot3d("d2",xmin=-2,xmax=2,ymin=-3,ymax=1):
```

Pembatasan pada garis (AB) lebih terkenal:

```
> plot2d("abs(x+1)+abs(x-1)",xmin=-3,xmax=3):
```

Tiga poin

Sekarang, hal-hal menjadi kurang sederhana: Hal ini sedikit kurang dikenal bahwa $MA+MB+MC$ mencapai minimumnya pada satu titik di bidang, tetapi untuk menentukannya tidak sesederhana itu:

- 1) Jika salah satu sudut segitiga ABC lebih dari 120° (katakanlah di A), maka minimum dicapai pada titik ini (katakanlah AB+AC).

Contoh:

```
> C=[-4,1];
> function d3(x,y):=d2(x,y)+sqrt((x-C[1])^2+(y-C[2])^2)
> plot3d("d3",xmin=-5,xmax=3,ymin=-4,ymax=4);
> insimg;
> fcontour("d3",xmin=-4,xmax=1,ymin=-2,ymax=2,hue=1,title="The minimum is on A");
> P=(A_B_C_A)';
> plot2d(P[1],P[2],add=1,color=12);
> insimg;
```

- 2) Tetapi jika semua sudut segitiga ABC kurang dari 120° , minimumnya adalah pada titik F di bagian dalam segitiga, yang merupakan satu-satunya titik yang melihat sisi-sisi ABC dengan sudut yang sama (masing-masing 120°):

```
> C=[-0.5,1];
> plot3d("d3",xmin=-2,xmax=2,ymin=-2,ymax=2):
> fcontour("d3",xmin=-2,xmax=2,ymin=-2,ymax=2,hue=1,title="The Fermat point");
> P=(A_B_C_A)';
> plot2d(P[1],P[2],add=1,color=12);
> insimg;
```

Merupakan kegiatan yang menarik untuk merealisasikan gambar di atas dengan perangkat lunak geometri; sebagai contoh, saya tahu sebuah perangkat lunak yang ditulis dalam bahasa Java yang memiliki instruksi "garis kontur"...

Semua hal di atas telah ditemukan oleh seorang hakim Perancis bernama Pierre de Fermat; dia menulis surat kepada para ahli lain seperti pendeta Marin Mersenne dan Blaise Pascal yang bekerja di bagian pajak penghasilan. Jadi titik unik F sedemikian rupa sehingga $FA+FB+FC$ minimal, disebut titik Fermat dari segitiga. Namun tampaknya beberapa tahun sebelumnya, Torricelli dari Italia telah menemukan titik ini sebelum Fermat menemukannya! Bagaimanapun juga, tradisinya adalah mencatat titik F ini...

Empat poin

Langkah selanjutnya adalah menambahkan titik ke-4 D dan mencoba meminimumkan $MA+MB+MC+MD$; misalkan Anda adalah operator TV kabel dan ingin menemukan di bidang mana Anda harus meletakkan antena sehingga Anda dapat memberi makan empat desa dan menggunakan panjang kabel sesedikit mungkin!

```

> D=[1,1];
> function d4(x,y):=d3(x,y)+sqrt((x-D[1])^2+(y-D[2])^2)
> plot3d("d4",xmin=-1.5,xmax=1.5,ymin=-1.5,ymax=1.5):
> fcontour("d4",xmin=-1.5,xmax=1.5,ymin=-1.5,ymax=1.5,hue=1);
> P=(A_B_C_D)';
> plot2d(P[1],P[2],points=1,add=1,color=12);
> insimg;

```

Masih ada nilai minimum dan tidak ada simpul A, B, C, maupun D:

```

> function f(x):=d4(x[1],x[2])
> neldermin("f",[0.2,0.2])

```

[0.142858, 0.142857]

Tampaknya dalam kasus ini, koordinat titik optimal adalah rasional atau mendekati rasional...

Karena ABCD adalah sebuah bujur sangkar, maka kita berharap bahwa titik optimalnya adalah pusat dari ABCD:

```

> C=[-1,1];
> plot3d("d4",xmin=-1,xmax=1,ymin=-1,ymax=1):
> fcontour("d4",xmin=-1.5,xmax=1.5,ymin=-1.5,ymax=1.5,hue=1);
> P=(A_B_C_D)';
> plot2d(P[1],P[2],add=1,color=12,points=1);
> insimg;

```

Contoh 7: Bola Dandelin dengan Povray

Anda dapat menjalankan demonstrasi ini, jika Anda telah menginstal Povray, dan pvengine.exe pada path program.

Pertama, kita menghitung jari-jari bola.

Jika Anda melihat gambar di bawah ini, Anda dapat melihat bahwa kita membutuhkan dua lingkaran yang menyentuh dua garis yang membentuk kerucut, dan satu garis yang membentuk bidang yang memotong kerucut.

Kita menggunakan file geometri.e dari Euler untuk hal ini.

```
> load geometry;
```

Pertama, dua garis yang membentuk kerucut.

```
> g1 &= lineThrough([0,0],[1,a])
```

[- a, 1, 0]

```
> g2 &= lineThrough([0,0],[-1,a])
```

```
[ - a, - 1, 0]
```

Kemudian baris ketiga.

```
> g &= lineThrough([-1,0],[1,1])
```

```
[ - 1, 2, 1]
```

Kami merencanakan semuanya sejauh ini.

```
> setPlotRange(-1,1,0,2);
> color(black); plotLine(g(), "")
> a:=2; color(blue); plotLine(g1(),""), plotLine(g2(),""):
```

Sekarang, kita ambil titik umum pada sumbu y.

```
> P &= [0,u]
```

```
[ 0, u]
```

Hitung jarak ke g1.

```
> d1 &= distance(P,projectToLine(P,g1)); $d1
```

Hitung jarak ke g.

```
> d &= distance(P,projectToLine(P,g)); $d
```

Dan temukan pusat kedua lingkaran, yang jaraknya sama.

```
> sol &= solve(d1^2=d^2,u); $sol
```

Ada dua solusi.

Kami mengevaluasi solusi simbolis, dan menemukan kedua pusat, dan kedua jarak.

```
> u := sol()
```

```
[ 0.333333, 1]
```

```
> dd := d()  
  
[0.149071, 0.447214]
```

Plot lingkaran-lingkaran tersebut ke dalam gambar.

```
> color(red);  
> plotCircle(circleWithCenter([0,u[1]],dd[1]), "");  
> plotCircle(circleWithCenter([0,u[2]],dd[2]), "");  
> insimg;
```

Plot dengan Povray

Selanjutnya kita plot semuanya dengan Povray. Perhatikan bahwa Anda mengubah perintah apapun pada urutan perintah Povray berikut ini, dan jalankan kembali semua perintah dengan Shift-Return.

Pertama kita memuat fungsi povray.

```
> load povray;  
> defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe

Kami menyiapkan pemandangan dengan tepat.

```
> povstart(zoom=11,center=[0,0,0.5],height=10°,angle=140°);
```

Selanjutnya kita tuliskan kedua bola tersebut ke file Povray.

```
> writeln(povsphere([0,0,u[1]],dd[1],povlook(red)));  
  
u is not a variable!  
Error in:  
writeln(povsphere([0,0,u[1]],dd[1],povlook(red))); ...  
^
```

```
> writeln(povsphere([0,0,u[2]],dd[2],povlook(red)));
```

Dan kerucutnya, transparan.

```
> writeln(povcone([0,0,0],0,[0,0,a],1,povlook(lightgray,1)));
```

Kami menghasilkan bidang yang terbatas pada kerucut.

```

> gp=g();
> pc=povcone([0,0,0],0,[0,0,a],1,"");
> vp=[gp[1],0,gp[2]]; dp=gp[3];
> writeln(povplane(vp,dp,povlook(blue,0.5),pc));

```

Sekarang kita menghasilkan dua titik pada lingkaran, di mana bola menyentuh kerucut.

```

> function turnz(v) := return [-v[2],v[1],v[3]]
> P1=projectToLine([0,u[1]],g()); P1=turnz([P1[1],0,P1[2]]);
> writeln(povpoint(P1,povlook(yellow)));
> P2=projectToLine([0,u[2]],g()); P2=turnz([P2[1],0,P2[2]]);
> writeln(povpoint(P2,povlook(yellow)));

```

Kemudian, kita menghasilkan dua titik di mana bola-bola tersebut menyentuh bidang. Ini adalah fokus elips.

```

> P3=projectToLine([0,u[1]],g()); P3=[P3[1],0,P3[2]];
> writeln(povpoint(P3,povlook(yellow)));
> P4=projectToLine([0,u[2]],g()); P4=[P4[1],0,P4[2]];
> writeln(povpoint(P4,povlook(yellow)));

```

Selanjutnya kita menghitung perpotongan P1P2 dengan bidang.

```

> t1=scalp(vp,P1)-dp; t2=scalp(vp,P2)-dp; P5=P1+t1/(t1-t2)*(P2-P1);
> writeln(povpoint(P5,povlook(yellow)));

```

Kami menghubungkan titik-titik dengan segmen garis.

```

> writeln(povsegment(P1,P2,povlook(yellow)));
> writeln(povsegment(P5,P3,povlook(yellow)));
> writeln(povsegment(P5,P4,povlook(yellow)));

```

Sekarang, kita menghasilkan pita abu-abu, di mana bola-bola menyentuh kerucut.

```

> pcw=povcone([0,0,0],0,[0,0,a],1.01);
> pc1=povcylinder([0,0,P1[3]-defaultpointsize/2],[0,0,P1[3]+defaultpointsize/2],1);
> writeln(povintersection([pcw,pc1],povlook(gray)));
> pc2=povcylinder([0,0,P2[3]-defaultpointsize/2],[0,0,P2[3]+defaultpointsize/2],1);
> writeln(povintersection([pcw,pc2],povlook(gray)));

```

Mulai program Povray.

```

> povend();

```

Untuk mendapatkan Anaglyph ini, kita perlu memasukkan semuanya ke dalam fungsi scene. Fungsi ini akan digunakan dua kali nanti.

```
> function scene () ...  
  
    global a,u,dd,g,g1,defaultpointsize;  
    writeln(povsphere([0,0,u[1]],dd[1],povlook(red)));  
    writeln(povsphere([0,0,u[2]],dd[2],povlook(red)));  
    writeln(povcone([0,0,0],0,[0,0,a],1,povlook(lightgray,1)));  
    gp=g();  
    pc=povcone([0,0,0],0,[0,0,a],1,"");  
    vp=[gp[1],0,gp[2]]; dp=gp[3];  
    writeln(povplane(vp,dp,povlook(blue,0.5),pc));  
    P1=projectToLine([0,u[1]],g1()); P1=turnz([P1[1],0,P1[2]]);  
    writeln(povpoint(P1,povlook(yellow)));  
    P2=projectToLine([0,u[2]],g1()); P2=turnz([P2[1],0,P2[2]]);  
    writeln(povpoint(P2,povlook(yellow)));  
    P3=projectToLine([0,u[1]],g()); P3=[P3[1],0,P3[2]];  
    writeln(povpoint(P3,povlook(yellow)));  
    P4=projectToLine([0,u[2]],g()); P4=[P4[1],0,P4[2]];  
    writeln(povpoint(P4,povlook(yellow)));  
    t1=scalp(vp,P1)-dp; t2=scalp(vp,P2)-dp; P5=P1+t1/(t1-t2)*(P2-P1);  
    writeln(povpoint(P5,povlook(yellow)));  
    writeln(povsegment(P1,P2,povlook(yellow)));  
    writeln(povsegment(P5,P3,povlook(yellow)));  
    writeln(povsegment(P5,P4,povlook(yellow)));  
    pcw=povcone([0,0,0],0,[0,0,a],1.01);  
    pcl=povcylinder([0,0,P1[3]-defaultpointsize/2],[0,0,P1[3]+defaultpointsize/2],1);  
    writeln(povintersection([pcw,pcl],povlook(gray)));  
    pc2=povcylinder([0,0,P2[3]-defaultpointsize/2],[0,0,P2[3]+defaultpointsize/2],1);  
    writeln(povintersection([pcw,pc2],povlook(gray)));  
endfunction
```

Anda memerlukan kacamata merah/sian untuk mengapresiasi efek berikut ini.

```
> povanaglyph("scene",zoom=11,center=[0,0,0.5],height=10°,angle=140°);
```

Contoh 8: Geometri Bumi

Pada buku catatan ini, kita ingin melakukan beberapa komputasi bola. Fungsi-fungsi tersebut terdapat pada file "spherical.e" pada folder contoh. Kita perlu memuat file tersebut terlebih dahulu.

```
> load "spherical.e";
```

Untuk memasukkan posisi geografis, kita menggunakan vektor dengan dua koordinat dalam radian (utara dan timur, nilai negatif untuk selatan dan barat). Berikut ini adalah koordinat untuk Kampus FMIPA UNY.

```
> FMIPA=[rad(-7,-46.467),rad(110,23.05)]
```

```
[-0.13569, 1.92657]
```

Anda dapat mencetak posisi ini dengan sposprint (cetak posisi bola).

```
> sposprint(FMIPA) // posisi garis lintang dan garis bujur FMIPA UNY
```

S $7^{\circ}46.467'$ E $110^{\circ}23.050'$

Mari kita tambahkan dua kota lagi, Solo dan Semarang.

```
> Solo=[rad(-7,-34.333),rad(110,49.683)]; Semarang=[rad(-6,-59.05),rad(110,24.533)];  
> sposprint(Solo), sposprint(Semarang),
```

S $7^{\circ}34.333'$ E $110^{\circ}49.683'$

S $6^{\circ}59.050'$ E $110^{\circ}24.533'$

Pertama, kita menghitung vektor dari satu titik ke titik lainnya pada bola ideal. Vektor ini adalah [heading, jarak] dalam radian. Untuk menghitung jarak di bumi, kita kalikan dengan jari-jari bumi pada garis lintang 7° .

```
> br=svector(FMIPA,Solo); degprint(br[1]), br[2]*rearth( $7^{\circ}$ )->km // perkiraan jarak FMIPA-Solo
```

$65^{\circ}20'26.60''$
53.8945384608

Ini adalah perkiraan yang baik. Rutinitas berikut ini menggunakan perkiraan yang lebih baik lagi. Pada jarak yang begitu pendek, hasilnya hampir sama.

```
> esdist(FMIPA,Semarang)->" km", // perkiraan jarak FMIPA-Semarang
```

88.0114026318 km

Ada fungsi untuk judul, dengan mempertimbangkan bentuk elips bumi. Sekali lagi, kami mencetak dengan cara yang canggih.

```
> sdegprint(esdir(FMIPA,Solo))
```

65.34°

Sudut segitiga melebihi 180° pada bola.

```
> asum=sangle(Solo,FMIPA,Semarang)+sangle(FMIPA,Solo,Semarang)+sangle(FMIPA,Semarang,Solo)
```

$180^{\circ}0'10.77''$

Ini bisa digunakan untuk menghitung luas area segitiga. Catatan: Untuk segitiga kecil, cara ini tidak akurat karena kesalahan pengurangan dalam asum- π .

```
> (asum-pi)*rearth( $48^{\circ}$ )^2->" km^2", // perkiraan luas segitiga FMIPA-Solo-Semarang
```

2116.02948749 km 2

Ada sebuah fungsi untuk hal ini, yang menggunakan garis lintang rata-rata segitiga untuk menghitung radius bumi, dan menangani kesalahan pembulatan untuk segitiga yang sangat kecil.

```
> esarea(Solo,FMIPA,Semarang) -> " km^2", //perkiraan yang sama dengan fungsi esarea()
```

2123.64310526 km²

Kita juga dapat menambahkan vektor ke posisi. Sebuah vektor berisi arah dan jarak, keduanya dalam radian. Untuk mendapatkan sebuah vektor, kita menggunakan svector. Untuk menambahkan sebuah vektor ke sebuah posisi, kita menggunakan saddvector.

```
> v=svector(FMIPA,Solo); sposprint(saddvector(FMIPA,v)), sposprint(Solo),
```

S 7°34.333' E 110°49.683'
S 7°34.333' E 110°49.683'

Fungsi-fungsi ini mengasumsikan bola yang ideal. Hal yang sama di bumi.

```
> sposprint(esadd(esdir(FMIPA,esdist(FMIPA,Solo)),esdist(FMIPA,Solo))), sposprint(Solo),
```

S 7°34.333' E 110°49.683'
S 7°34.333' E 110°49.683'

Mari kita beralih ke contoh yang lebih besar, Tugu Jogja dan Monas Jakarta (menggunakan Google Earth untuk mencari koordinatnya).

```
> Tugu=[-7.7833°,110.3661°]; Monas=[-6.175°,106.811944°];  
> sposprint(Tugu), sposprint(Monas)
```

S 7°46.998' E 110°21.966'
S 6°10.500' E 106°48.717'

Menurut Google Earth, jaraknya adalah 429,66 km. Kami mendapatkan perkiraan yang bagus.

```
> esdist(Tugu,Monas) -> " km", // perkiraan jarak Tugu Jogja - Monas Jakarta
```

431.565659488 km

Judulnya sama dengan yang dihitung di Google Earth.

```
> degprint(esdir(Tugu,Monas))
```

294°17'2.85''

Namun demikian, kita tidak lagi mendapatkan posisi target yang tepat, jika kita menambahkan arah dan jarak ke posisi semula. Hal ini terjadi, karena kita tidak menghitung fungsi inversi secara tepat, tetapi mengambil perkiraan radius bumi di sepanjang jalur.

```
> sposprint(esadd(Tugu, esdir(Tugu, Monas), esdist(Tugu, Monas)))
```

S 6°10.500' E 106°48.717'

Namun demikian, kesalahannya tidak besar.

```
> sposprint(Monas),
```

S 6°10.500' E 106°48.717'

Tentu saja, kita tidak bisa berlayar dengan arah yang sama dari satu tujuan ke tujuan lainnya, jika kita ingin mengambil jalur terpendek. Bayangkan, Anda terbang ke arah NE mulai dari titik mana pun di bumi. Kemudian Anda akan berputar ke kutub utara. Lingkaran besar tidak mengikuti arah yang konstan!

Perhitungan berikut ini menunjukkan bahwa kita akan melenceng dari tujuan yang benar, jika kita menggunakan arah yang sama selama perjalanan.

```
> dist=esdist(Tugu, Monas); hd=esdir(Tugu, Monas);
```

Sekarang kita tambahkan 10 kali sepersepuluh dari jarak tersebut, dengan menggunakan arah menuju Monas, kita akan sampai di Tugu.

```
> p=Tugu; loop 1 to 10; p=esadd(p,hd,dist/10); end;
```

Hasilnya jauh berbeda.

```
> sposprint(p), skmprint(esdist(p, Monas))
```

S 6°11.250' E 106°48.372'

1.529km

Sebagai contoh lain, mari kita ambil dua titik di bumi pada garis lintang yang sama.

```
> P1=[30°, 10°]; P2=[30°, 50°];
```

Jalur terpendek dari P1 ke P2 bukanlah lingkaran lintang 30°, tetapi jalur yang lebih pendek yang dimulai 10° lebih jauh ke utara di P1.

```
> sdegprint(esdir(P1, P2))
```

79.69°

Namun, jika kita mengikuti pembacaan kompas ini, kita akan berputar ke kutub utara! Jadi, kita harus menyesuaikan arah kita di sepanjang jalan. Untuk tujuan kasar, kita sesuaikan pada 1/10 dari jarak total.

```
> p=P1; dist=esdist(P1,P2); ...
> loop 1 to 10; dir=esdir(p,P2); sdeprint(dir), p=esadd(p,dir,dist/10); end;
```

79.69°
81.67°
83.71°
85.78°
87.89°
90.00°
92.12°
94.22°
96.29°
98.33°

Jaraknya tidak tepat, karena kita akan menambahkan sedikit kesalahan, jika kita mengikuti judul yang sama terlalu lama.

```
> skmprint(esdist(p,P2))
```

0.203km

Kita akan mendapatkan perkiraan yang baik, jika kita menyesuaikan arah setiap 1/100 dari total jarak dari Tugu ke Monas.

```
> p=Tugu; dist=esdist(Tugu,Monas); ...
> loop 1 to 100; p=esadd(p,esdir(p,Monas),dist/100); end;
> skmprint(esdist(p,Monas))
```

0.000km

Untuk keperluan navigasi, kita bisa mendapatkan urutan posisi GPS di sepanjang Bundaran Hotel Indonesia menuju Monas dengan fungsi navigate.

```
> load spherical; v=navigate(Tugu,Monas,10); ...
> loop 1 to rows(v); sposprint(v[#]), end;
```

S 7°46.998' E 110°21.966'
S 7°37.422' E 110°0.573'
S 7°27.829' E 109°39.196'
S 7°18.219' E 109°17.834'
S 7°8.592' E 108°56.488'
S 6°58.948' E 108°35.157'
S 6°49.289' E 108°13.841'
S 6°39.614' E 107°52.539'
S 6°29.924' E 107°31.251'
S 6°20.219' E 107°9.977'
S 6°10.500' E 106°48.717'

Kami menulis sebuah fungsi, yang memplot bumi, dua posisi, dan posisi di antaranya.

```
> function testplot ...
```

```
useglobal;
plotearth;
plotpos(Tugu,"Tugu Jogja"); plotpos(Monas,"Tugu Monas");
plotposline(v);
endfunction
```

Sekarang plot semuanya.

```
> plot3d("testplot",angle=25, height=6,>own,>user,zoom=4):
```

Atau gunakan plot3d untuk mendapatkan tampilan anaglyph. Ini terlihat sangat bagus dengan kacamata merah/cyan.

```
> plot3d("testplot",angle=25,height=6,distance=5,own=1,anaglyph=1,zoom=4):
```

Latihan

1. Gambarlah segi-n beraturan jika diketahui titik pusat O, n, dan jarak titik pusat ke titik-titik sudut segi-n tersebut (jari-jari lingkaran luar segi-n), r.

Petunjuk:

- Besar sudut pusat yang menghadap masing-masing sisi segi-n adalah $(360/n)$.
- Titik-titik sudut segi-n merupakan perpotongan lingkaran luar segi-n dan garis-garis yang melalui pusat dan saling membentuk sudut sebesar kelipatan $(360/n)$.
- Untuk n ganjil, pilih salah satu titik sudut adalah di atas.
- Untuk n genap, pilih 2 titik di kanan dan kiri lurus dengan titik pusat.
- Anda dapat menggambar segi-3, 4, 5, 6, 7, dst beraturan.

Penyelesaian:

```
> load geometry
```

Numerical and symbolic geometry.

```
> setPlotRange(-3.5,3.5,-3.5,3.5);
> A=[-2,-2]; plotPoint(A,"A");
> B=[2,-2]; plotPoint(B,"B");
> C=[0,3]; plotPoint(C,"C");
> plotSegment(A,B,"c");
> plotSegment(B,C,"a");
> plotSegment(A,C,"b");
> aspect(1):
> c=circleThrough(A,B,C);
> R=getCircleRadius(c);
> O=getCircleCenter(c);
> plotPoint(O,"O");
```

```

> l=angleBisector(A,C,B);
> color(2); plotLine(l); color(1);
> plotCircle(c,"Lingkaran luar segitiga ABC");

```

2. Gambarlah suatu parabola yang melalui 3 titik yang diketahui.

Petunjuk:

- Misalkan persamaan parabolanya $y = ax^2 + bx + c$.
- Subsitusikan koordinat titik-titik yang diketahui ke persamaan tersebut.
- Selesaikan SPL yang terbentuk untuk mendapatkan nilai-nilai a, b, c .

Penyelesaian:

```

> load geometry;
> setPlotRange(5); P=[2,0]; Q=[4,0]; R=[0,-4];
> plotPoint(P,"P"); plotPoint(Q,"Q"); plotPoint(R,"R");
> sol &= solve([a+b=-c,16*a+4*b=-c,c=-4],[a,b,c])

```

$\begin{bmatrix} [a = -1, b = 5, c = -4] \end{bmatrix}$

Sehingga didapatkan nilai $a = -1$, $b = 5$ dan $c = -4$

```

> function y(x) &= -x^2+5*x-4; $y(x)
> plot2d("y(x)",-5,5,-5,5):

```

3. Gambarlah suatu segi-4 yang diketahui keempat titik sudutnya, misalnya A, B, C, D.

- Tentukan apakah segi-4 tersebut merupakan segi-4 garis singgung (sisinya-sisinya merupakan garis singgung lingkaran yang sama yakni lingkaran dalam segi-4 tersebut).
- Suatu segi-4 merupakan segi-4 garis singgung apabila keempat garis bagi sudutnya bertemu di satu titik.
- Jika segi-4 tersebut merupakan segi-4 garis singgung, gambar lingkaran dalamnya.
- Tunjukkan bahwa syarat suatu segi-4 merupakan segi-4 garis singgung apabila hasil kali panjang sisi-sisi yang berhadapan sama.

Penyelesaian:

```
> load geometry
```

Numerical and symbolic geometry.

```
> setPlotRange(-4.5,4.5,-4.5,4.5);  
> A=[-3,-3]; plotPoint(A,"A");  
> B=[3,-3]; plotPoint(B,"B");  
> C=[3,3]; plotPoint(C,"C");  
> D=[-3,3]; plotPoint(D,"D");  
> plotSegment(A,B,"");  
> plotSegment(B,C,"");  
> plotSegment(C,D,"");  
> plotSegment(A,D,"");  
> aspect(1):  
> l=angleBisector(A,B,C);  
> m=angleBisector(B,C,D);  
> P=lineIntersection(l,m);  
> color(5); plotLine(l); plotLine(m); color(1);  
> plotPoint(P,"P");
```

Dari gambar diatas terlihat bahwa keempat garis bagi sudutnya bertemu di satu titik yaitu titik P.

```
> r=norm(P-projectToLine(P,lineThrough(A,B)));  
> plotCircle(circleWithCenter(P,r),"Lingkaran dalam segiempat ABCD"):
```

Dari gambar diatas, terlihat bahwa sisi-sisinya merupakan garis singgung lingkaran yang sama yaitu lingkaran dalam segiempat.

Akan ditunjukkan bahwa hasil kali panjang sisi-sisi yang berhadapan sama.

```
> AB=norm(A-B) //panjang sisi AB
```

6

```
> CD=norm(C-D) //panjang sisi CD
```

6

```
> AD=norm(A-D) //panjang sisi AD
```

6

```
> BC=norm(B-C) //panjang sisi BC
```

6

```
> AB.CD
```

36

```
> AD.BC
```

36

Terbukti bahwa hasil kali panjang sisi-sisi yang berhadapan sama yaitu 36. Jadi dapat dipastikan bahwa segiempat tersebut merupakan segiempat garis singgung.

4. Gambarlah suatu ellips jika diketahui kedua titik fokusnya, misalnya P dan Q. Ingat ellips dengan fokus P dan Q adalah tempat kedudukan titik-titik yang jumlah jarak ke P dan ke Q selalu sama (konstan).

Penyelesaian :

Diketahui kedua titik fokus $P = [-1, -1]$ dan $Q = [1, -1]$

```
> P=[-1,-1]; Q=[1,-1];
> function d1(x,y):=sqrt((x-P[1])^2+(y-P[2])^2)
> Q=[1,-1]; function d2(x,y):=sqrt((x-P[1])^2+(y-P[2])^2)+sqrt((x-Q[1])^2+(y-Q[2])^2)
> fcontour("d2",xmin=-2,xmax=2,ymin=-3,ymax=1,hue=1):
```

Grafik yang lebih menarik

```
> plot3d("d2",xmin=-2,xmax=2,ymin=-3,ymax=1):
```

Batasan ke garis PQ

```
> plot2d("abs(x+1)+abs(x-1)",xmin=-3,xmax=3):
```

5. Gambarlah suatu hiperbola jika diketahui kedua titik fokusnya, misalnya P dan Q. Ingat ellips dengan fokus P dan Q adalah tempat kedudukan titik-titik yang selisih jarak ke P dan ke Q selalu sama (konstan).

Penyelesaian :

```
> P=[-1,-1]; Q=[1,-1];
> function d1(x,y):=sqrt((x-p[1])^2+(y-p[2])^2)
> Q=[1,-1]; function d2(x,y):=sqrt((x-P[1])^2+(y-P[2])^2)+sqrt((x+Q[1])^2+(y+Q[2])^2)
> fcontour("d2",xmin=-2,xmax=2,ymin=-3,ymax=1,hue=1):
```

Grafik yang lebih menarik

```
> plot3d("d2",xmin=-2,xmax=2,ymin=-3,ymax=1):
```

BAB 8

PENGGUNAAN SOFTWARE EMT UNTUK STATISTIKA

EMT untuk Statistika

In this notebook, we demonstrate the main statistical plots, tests and distributions in Euler.

Let us start with some descriptive statistics. This is not an introduction to statistics. So you might need some background to understand the details.

Assume the following measurements. We wish to compute the mean value and the measured standard deviation.

```
>M=[1000,1004,998,997,1002,1001,998,1004,998,997]; ...
>median(M), mean(M), dev(M),
```

```
999
999.9
2.72641400622
```

We can plot the box-and-whiskers plot for the data. In our case there are no outliers.

```
>aspect(1.75); boxplot(M);
```

We compute the probability that a value is bigger than 1005, assuming the measured values from a normal distribution.

All functions for distributions in Euler end with ...dis and compute the cumulative probability distribution (CPF).

$$\text{normaldis}(x,m,d) = \int_{-\infty}^x \frac{1}{d\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{t-m}{d})^2} dt.$$

We print the result in % with 2 digits accuracy using the print function.

```
>print((1-normaldis(1005,mean(M),dev(M)))*100,2,unit="%")
```

3.07 %

For the next example, we assume the following numbers of men in given a size ranges.

```
>r=155.5:4:187.5; v=[22,71,136,169,139,71,32,8];
```

Here is a plot of the distribution.

```
>plot2d(r,v,a=150,b=200,c=0,d=190,bar=1,style="/"):
```

We can put such raw data into a table.

Tables are a method to store statistical data. Our table should contain three columns: Start of range, end of range, number of men in the range.

Tables can be printed with headers. We use a vector of strings to set the headers.

```
>T:=r[1:8]' | r[2:9]' | v'; writetable(T,labc=["BB","BA","Frek"])
```

BB	BA	Frek
155.5	159.5	22
159.5	163.5	71
163.5	167.5	136
167.5	171.5	169
171.5	175.5	139
175.5	179.5	71
179.5	183.5	32
183.5	187.5	8

If we need the mean value and other statistics of the sizes, we need to compute the midpoint of the ranges. We can use the first two columns of our table for this.

The symbol "|" is used to separate column, the function "writetable" is used to write the table, with options "labc" is to specify column headers.

```
>(T[,1]+T[,2])/2 // the midpoint of each interval
```

157.5
161.5
165.5
169.5
173.5
177.5
181.5
185.5

But it is easier, to fold the ranges with the vector [1/2,1/2].

```
>M=fold(r,[0.5,0.5])
```

```
[157.5, 161.5, 165.5, 169.5, 173.5, 177.5, 181.5, 185.5]
```

Now we can compute the mean and deviation of the sample with the given frequencies.

```
>{m,d}=meandev(M,v); m, d,
```

```
169.901234568  
5.98912964449
```

Let us add the normal distribution of the values to the above bar plot. The formula for normal distribution with mean m and standard deviation d is:

$$y = \frac{1}{d\sqrt{2\pi}} e^{\frac{-(x-m)^2}{2d^2}}.$$

Because its values are between 0 and 1, to plot it on the bar plot it must be multiplied by 4 times the total number of data.

```
>plot2d("qnormal(x,m,d)*sum(v)*4", ...  
> xmin=min(r),xmax=max(r),thickness=3,add=1):
```

Tables

In the directory of this notebook you find a file with a table. The data represent the results of a survey. Here are the first four lines of the file. The data are from an German online book "Einführung in die Statistik mit R" by A. Handl.

```
>printfile("table.dat",4);
```

```
Person Sex Age Titanic Evaluation Tip Problem  
1 m 30 n . 1.80 n  
2 f 23 y g 1.80 n  
3 f 26 y g 1.80 y
```

The table contains 7 columns of numbers or tokens (strings). We want read the table from the file. First, we use our own translation for the tokens.

For this, we define sets of tokens. The function strtokens() gets a string vector of tokens from a given string.

```
>mf:=[ "m", "f" ]; yn:=[ "y", "n" ]; ev:=strtokens("g vg m b vb");
```

Now we read the table with these translations.

The arguments tok2, tok4 etc. are the translations of the columns of the table. These arguments are not in the parameter list of readtable(), so you need to provide them with ":=".

```
>{MT,hd}=readtable("table.dat",tok2:=mf,tok4:=yn,tok5:=ev,tok7:=yn);  
>load over statistics;
```

For printing, we need to specify the same token sets. We print the first four lines only.

```
>writetable(MT[1:10],labc=hd,wc=5,tok2:=mf,tok4:=yn,tok5:=ev,tok7:=yn);
```

Person	Sex	Age	Titanic	Evaluation	Tip	Problem
1	m	30	n	.	1.8	n
2	f	23	y	g	1.8	n
3	f	26	y	g	1.8	y
4	m	33	n	.	2.8	n
5	m	37	n	.	1.8	n
6	m	28	y	g	2.8	y
7	f	31	y	vg	2.8	n
8	m	23	n	.	0.8	n
9	f	24	y	vg	1.8	y
10	m	26	n	.	1.8	n

The dots "." represent values, which are not available.

If we do not want to specify the tokens for the translation in advance, we only need to specify, which columns contain tokens and not numbers.

```
>ctok=[2,4,5,7]; {MT,hd,tok}=readtable("table.dat",ctok=ctok);
```

The function readtable() now returns a set of tokens.

```
>tok
```

```
m  
n  
f  
y  
g  
vg
```

The table contains the entries from the file with tokens translated to numbers.

The special string NA="." is interpreted as "Not Available", and gets NAN (not a number) in the table. This translation can be changed with the parameters NA, and NAvl.

```
>MT[1]
```

```
[1, 1, 30, 2, NAN, 1.8, 2]
```

Here is the content of the table with untranslated numbers.

```
>writetable(MT,wc=5)
```

1	1	30	2	.	1.8	2
2	3	23	4	5	1.8	2
3	3	26	4	5	1.8	4
4	1	33	2	.	2.8	2

5	1	37	2	.	1.8	2
6	1	28	4	5	2.8	4
7	3	31	4	6	2.8	2
8	1	23	2	.	0.8	2
9	3	24	4	6	1.8	4
10	1	26	2	.	1.8	2
11	3	23	4	6	1.8	4
12	1	32	4	5	1.8	2
13	1	29	4	6	1.8	4
14	3	25	4	5	1.8	4
15	3	31	4	5	0.8	2
16	1	26	4	5	2.8	2
17	1	37	2	.	3.8	2
18	1	38	4	5	.	2
19	3	29	2	.	3.8	2
20	3	28	4	6	1.8	2
21	3	28	4	1	2.8	4
22	3	28	4	6	1.8	4
23	3	38	4	5	2.8	2
24	3	27	4	1	1.8	4
25	1	27	2	.	2.8	4

For convenience, you can put the output of `readtable()` into a list.

```
>Table={{readtable("table.dat",ctok=ctok)}},
```

Using the same token columns and the tokens read from the file, we can print the table. We can either specify `ctok`, `tok`, etc. or use the list `Table`.

```
>writetable(Table,ctok=ctok,wc=5);
```

Person	Sex	Age	Titanic	Evaluation	Tip	Problem
1	m	30	n	.	1.8	n
2	f	23	y	g	1.8	n
3	f	26	y	g	1.8	y
4	m	33	n	.	2.8	n
5	m	37	n	.	1.8	n
6	m	28	y	g	2.8	y
7	f	31	y	vg	2.8	n
8	m	23	n	.	0.8	n
9	f	24	y	vg	1.8	y
10	m	26	n	.	1.8	n
11	f	23	y	vg	1.8	y
12	m	32	y	g	1.8	n
13	m	29	y	vg	1.8	y
14	f	25	y	g	1.8	y
15	f	31	y	g	0.8	n
16	m	26	y	g	2.8	n
17	m	37	n	.	3.8	n
18	m	38	y	g	.	n
19	f	29	n	.	3.8	n
20	f	28	y	vg	1.8	n
21	f	28	y	m	2.8	y

```

22     f    28      y      vg   1.8      y
23     f    38      y      g    2.8      n
24     f    27      y      m    1.8      y
25     m    27      n      .    2.8      y

```

The function `tablecol()` returns the values of columns of the table, skipping any rows with NAN values("." in the file), and the indices of the columns, which contain these values.

```
>{c,i}=tablecol(MT,[5,6]);
```

We can use this to extract columns from the table for a new table.

```
>j=[1,5,6]; writetable(MT[i,j],labc=hd[j],ctok=[2],tok=tok)
```

Person	Evaluation	Tip
2	g	1.8
3	g	1.8
6	g	2.8
7	vg	2.8
9	vg	1.8
11	vg	1.8
12	g	1.8
13	vg	1.8
14	g	1.8
15	g	0.8
16	g	2.8
20	vg	1.8
21	m	2.8
22	vg	1.8
23	g	2.8
24	m	1.8

Of course, we need to extract the table itself from the list `Table` in this case.

```
>MT=Table[1];
```

Of course, we can also use it to determine the mean value of a column or any other statistical value.

```
>mean(tablecol(MT,6))
```

2.175

The `getstatistics()` function returns the elements in a vector, and their counts. We apply it to the "m" and "f" values in the second column of our table.

```
>{xu,count}=getstatistics(tablecol(MT,2)); xu, count,
```

```
[1, 3]
[12, 13]
```

We can print the result in a new table.

```
>writetable(count', labr=tok[xu])
```

m	12
f	13

The function `selectable()` returns a new table with the values in one column selected from a vector of indices. First we look up the indices of two of our values in the token table.

```
>v:=indexof(tok, ["g", "vg"])
```

[5, 6]

Now we can select the rows of the table, which have any of the values in v in their 5-th row.

```
>MT1:=MT[selectrows(MT, 5, v)]; i:=sortedrows(MT1, 5);
```

Now we can print the table, with extracted and sorted values in the 5-th column.

```
>writetable(MT1[i], labc=hd, ctok=ctok, tok=tok, wc=7);
```

Person	Sex	Age	Titanic	Evaluation	Tip	Problem
2	f	23	y	g	1.8	n
3	f	26	y	g	1.8	y
6	m	28	y	g	2.8	y
18	m	38	y	g	.	n
16	m	26	y	g	2.8	n
15	f	31	y	g	0.8	n
12	m	32	y	g	1.8	n
23	f	38	y	g	2.8	n
14	f	25	y	g	1.8	y
9	f	24	y	vg	1.8	y
7	f	31	y	vg	2.8	n
20	f	28	y	vg	1.8	n
22	f	28	y	vg	1.8	y
13	m	29	y	vg	1.8	y
11	f	23	y	vg	1.8	y

For the next statistic, we want to relate two columns of the table. So we extract column 2 and 4 and sort the table.

```
>i=sortedrows(MT, [2, 4]); ...
> writetable(tablecol(MT[i], [2, 4])', ctok=[1, 2], tok=tok)
```

m	n
m	n
m	n
m	n

m	n
m	n
m	n
m	y
m	y
m	y
m	y
m	y
f	n
f	y
f	y
f	y
f	y
f	y
f	y
f	y
f	y
f	y
f	y
f	y
f	y
f	y

With `getstatistics()`, we can also relate the counts in two columns of the table to each other.

```
>MT24=tablecol(MT,[2,4]); ...
>{xu1,xu2,count}=getstatistics(MT24[1],MT24[2]); ...
>writetable(count,labr=tok[xu1],labc=tok[xu2])
```

	n	y
m	7	5
f	1	12

A table can be written to a file.

```
>filename="test.dat"; ...
>writetable(count,labr=tok[xu1],labc=tok[xu2],file=filename);
```

Then we can read the table from the file.

```
>{MT2,hd,tok2,hdr}=readtable(filename,>clabs,>rlabs); ...
>writetable(MT2,labr=hdr,labc=hd)
```

	n	y
m	7	5
f	1	12

And delete the file.

```
>fileremove(filename);
```

Distributions

With `plot2d`, there is a very easy method to plot a distribution of experimental data.

```
>p=normal(1,1000); //1000 random normal-distributed sample p  
>plot2d(p,distribution=20,style="\\\"/\\\""); // plot the random sample p  
>plot2d("qnormal(x,0,1)",add=1); // add the standard normal distribution plot
```

Please note the different between the bar plot (sample) and the normal curve (the real distribution) . Reenter the three commands to see another sampling result.

Here is a comparison of 10 simulations of 1000 normal distributed values using a so-called box plot. This plot shows the median, the 25% and 75% quartiles, the minimal and maximal values, and the outliers.

```
>p=normal(10,1000); boxplot(p):
```

To generate random integers, Euler has `intrandom`. Let us simulate dice throws and plot the distribution.

We use the `getmultiplicities(v,x)` function, which counts how often the elements of `v` appear in `x`. Then we plot the the result using `columnsplot()`.

```
>k=intrandom(1,6000,6); ...  
>columnsplot(getmultiplicities(1:6,k)); ...  
>ygrid(1000,color=red):
```

While `intrandom(n,m,k)` returns uniformly distributed integers from 1 to k, it is possible to use any other given distribution of integers with `randpint()`.

In the following example, the probabilities for 1,2,3 are 0.4,0.1,0.5 respectively.

```
>randpint(1,1000,[0.4,0.1,0.5]); getmultiplicities(1:3,%)
```

```
[402, 90, 508]
```

Euler can produce random values from more distributions. Have a look into the reference.

E.g., we try the exponential distribution. A continuous random variable X is said to have an exponential distribution, if its PDF is given by

$$f_X(x) = \lambda e^{-\lambda x}, \quad x > 0, \quad \lambda > 0,$$

with parameter

$$\lambda = \frac{1}{\mu}, \quad \mu \text{ is the mean, and denoted by } X \sim \text{Exponential}(\lambda).$$

```
>plot2d(randexponential(1,1000,2),>distribution):
```

For many distributions, Euler can compute the distribution function and the inverse.

```
>plot2d("normaldis",-4,4):
```

The following is one way to plot a quantile.

```
>plot2d("qnormal(x,1,1.5)",-4,6); ...  
>plot2d("qnormal(x,1,1.5)",a=2,b=5,>add,>filled):
```

$$\text{normaldis}(x,m,d) = \int_{-\infty}^x \frac{1}{d\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{t-m}{d})^2} dt.$$

The probability to be in the green area is the following.

```
>normaldis(5,1,1.5)-normaldis(2,1,1.5)
```

0.248662156979

This can be computed numerically with the following integral.

$$\int_2^5 \frac{1}{1.5\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-1}{1.5})^2} dx.$$

```
>gauss("qnormal(x,1,1.5)",2,5)
```

0.248662156979

Let us compare the binomial distribution with the normal distribution of same mean and deviation. The function `invbindis()` solves a linear interpolation between integer values.

```
>invbindis(0.95,1000,0.5), invnormaldis(0.95,500,0.5*sqrt(1000))
```

525.516721219
526.007419394

The function `qdis()` is the density of the chi-square distribution. As usual, Euler maps vectors to this function. Thus we get a plot of all chi-square distributions with degrees 5 to 30 easily in the following way.

```
>plot2d("qchidis(x,(5:5:50)'),0,50):
```

Euler has accurate functions to evaluate distributions. Let us check `chidis()` with an integral.

The naming tries to be consistent. E.g.,

- the chi-square distribution is `chidis()`,
- the inverse function is `invchidis()`,
- the density is `qchidis()`.

The complements of the distribution (upper tail) is `chicdis()`.

```
>chidis(1.5,2), integrate("qchidis(x,2)",0,1.5)
```

```
0.527633447259  
0.527633447259
```

Discrete Distributions

To define your own discrete distribution, you can use the following method.

First we set the distribution function.

```
>wd = 0 | ((1:6)+[-0.01,0.01,0,0,0,0])/6
```

```
[0, 0.165, 0.335, 0.5, 0.666667, 0.833333, 1]
```

The meaning is that with probability $wd[i+1]-wd[i]$ we produce the random value i .

This is almost a uniform distribution. Let us define a random number generator for this. The `find(v,x)` function finds the value x in the vector v . It works for vectors x too.

```
>function wrongdice (n,m) := find(wd,random(n,m))
```

The error is so subtle that we see it only with very many iterations.

```
>columnsplot (getmultiplicities(1:6,wrongdice(1,1000000))):
```

Here is a simple function to check for uniform distribution of the values $1 \dots K$ in v . We accept the result, if for all frequencies

$$\left| f_i - \frac{1}{K} \right| < \frac{\delta}{\sqrt{n}}.$$

```
>function checkrandom (v, delta=1) ...
```

```
K=max(v); n=cols(v);  
fr=getfrequencies(v,1:K);  
return max(fr/n-1/K)<delta/sqrt(n);  
endfunction
```

Indeed the function rejects the uniform distribution.

```
>checkrandom(wrongdice(1,1000000))
```

```
0
```

And it accepts the built-in random generator.

```
>checkrandom(intrandom(1, 1000000, 6))
```

1

We can compute the binomial distribution. First there is binomials() which returns the probability of i or less hits out of n trials.

```
>bindis(410, 1000, 0.4)
```

0.751401349654

The inverse Beta function is used to compute a Clopper-Pearson confidence interval for the parameter p. The default level is alpha.

The meaning of this interval is that if p is outside the interval, the observed result of 410 in 1000 is rare.

```
>clopperpearson(410, 1000)
```

[0.37932, 0.441212]

The following commands are the direct way to get the above result. But for large n, the direct summation is not accurate and slow.

```
>p=0.4; i=0:410; n=1000; sum(bin(n,i)*p^i*(1-p)^(n-i))
```

0.751401349655

By the way, invbinsum() computes the inverse of binomials().

```
>invbindis(0.75, 1000, 0.4)
```

409.932733047

In Bridge, we assume 5 outstanding cards (out of 52) in two hands (26 cards). Let us compute the probability of a distribution worse than 3:2 (e.g. 0:5, 1:4, 4:1 or 5:0).

```
>2*hypergeomsum(1, 5, 13, 26)
```

0.321739130435

There is also a simulation of multinomial distributions.

```
>randmultinomial(10, 1000, [0.4, 0.1, 0.5])
```

366	117	517
400	109	491
395	98	507
395	103	502
427	105	468
414	114	472
398	102	500
387	102	511
387	99	514
423	98	479

Plotting Data

To plot data, we try the results of the German elections since 1990, measured in seats.

```
>BW := [ ...
>1990, 662, 319, 239, 79, 8, 17; ...
>1994, 672, 294, 252, 47, 49, 30; ...
>1998, 669, 245, 298, 43, 47, 36; ...
>2002, 603, 248, 251, 47, 55, 2; ...
>2005, 614, 226, 222, 61, 51, 54; ...
>2009, 622, 239, 146, 93, 68, 76; ...
>2013, 631, 311, 193, 0, 63, 64];
```

For the parties, we use a string of names.

```
>P := ["CDU/CSU", "SPD", "FDP", "Gr", "Li"];
```

Let us print the percentages nicely.

First we extract the necessary columns. Columns 3 to 7 are the seats of each party, and column 2 is the total number of seats. column 1 is the year of the election.

```
>BT := BW[, 3:7]; BT := BT / sum(BT); YT := BW[, 1]';
```

Then we print the statistics in table form. We use the names as column headers, and the years as headers for the rows. The default width for the columns is `wc=10`, but we prefer a denser output. The columns will be expanded for the labels of the columns, if necessary.

```
>writetable(BT * 100, wc=6, dc=0, >fixed, labc=P, labr=YT)
```

	CDU/CSU	SPD	FDP	Gr	Li
1990	48	36	12	1	3
1994	44	38	7	7	4
1998	37	45	6	7	5
2002	41	42	8	9	0
2005	37	36	10	8	9
2009	38	23	15	11	12
2013	49	31	0	10	10

The following matrix multiplication extracts the sum of the percentages of the two big parties showing that the small parties have gained footage in the parliament until 2009.

```
>BT1:=(BT.[1;1;0;0;0])'*100
```

```
[84.29, 81.25, 81.1659, 82.7529, 72.9642, 61.8971, 79.8732]
```

There is also a simple statistical plot. We use it to display lines and points simultaneously. The alternative is to call `plot2d` twice with `>add`.

```
>statplot(YT,BT1,"b"):
```

Define some colors for each party.

```
>CP:=[rgb(0.5,0.5,0.5),red,yellow,green,rgb(0.8,0,0)];
```

Now we can plot the results of the 2009 election and the changes into one plot using `figure`. We can add a vector of columns to each plot.

```
>figure(2,1); ...
>figure(1); columnsplot(BW[6,3:7],P,color=CP); ...
>figure(2); columnsplot(BW[6,3:7]-BW[5,3:7],P,color=CP); ...
>figure(0):
```

Data plots combine rows of statistical data in one plot.

```
>J:=BW[,1]'; DP:=BW[,3:7]'; ...
>dataplot(YT,BT',color=CP); ...
>labelbox(P,colors=CP,styles="[]",>points,w=0.2,x=0.3,y=0.4):
```

A 3D columns plot shows rows of statistical data in form of columns. We provide labels for the rows and the columns. `angle` is the viewing angle.

```
>columnsplot3d(BT,scols=P,srows=YT, ...
> angle=30°,ccols=CP):
```

Another representation is the mosaic plot. Note that the columns of the plot represent the columns of the matrix here. Because of the length of the label CDU/CSU, we take a smaller window than usual.

```
>shrinkwindow(>smaller); ...
>mosaicplot(BT',srows=YT,scols=P,color=CP,style="#"); ...
>shrinkwindow():
```

We can also do a pie chart. Since black and yellow form a coalition, we reorder the elements.

```
>i=[1,3,5,4,2]; piechart(BW[6,3:7][i],color=CP[i],lab=P[i]):
```

Here is another kind of plot.

```
>starplot(normal(1,10)+4,lab=1:10,>rays):
```

Some plots in plot2d are good for statics. Here is an impulse plot of random data, uniformly distributed in [0,1].

```
>plot2d(makeimpulse(1:10,random(1,10)),>bar):
```

But for exponentially distributed data, we may need a logarithmic plot.

```
>logimpulseplot(1:10,-log(random(1,10))*10):
```

The function columnsplot() is easier to use, since it needs just a vector of values. Moreover, it can set its labels to anything we want, we demonstrated this already in this tutorial.

Here is another application, where we count characters in a sentence and plot a statistics.

```
>v=strtochar("the quick brown fox jumps over the lazy dog"); ...
>w=ascii("a"):ascii("z"); x=getmultiplicities(w,v); ...
>cw=[]; for k=w; cw=cw|char(k); end; ...
>columnsplot(x,lab=cw,width=0.05):
```

It is also possible to manually set axes.

```
>n=10; p=0.4; i=0:n; x=bin(n,i)*p^i*(1-p)^(n-i); ...
>columnsplot(x,lab=i,width=0.05,<frame,<grid); ...
>yaxis(0,0:0.1:1,style="->",>left); xaxis(0,style="."); ...
>label("p",0,0.25), label("i",11,0); ...
>textbox(["Binomial distribution","with p=0.4"]):
```

The following is a way to plot the frequencies of numbers in a vector.

We create a vector of integer random numbers 1 to 6.

```
>v:=intrandom(1,10,10)
```

```
[2, 3, 6, 4, 8, 8, 10, 10, 5, 10]
```

Then extract the unique numbers in v.

```
>vu:=unique(v)
```

```
[2, 3, 4, 5, 6, 8, 10]
```

And plot the frequencies in a columns plot.

```
>columnsplot(getmultiplicities(vu,v),lab=vu,style="/"):
```

We want to demonstrate functions for the empirical distribution of values.

```
>x=normal(1,20);
```

The function empdist(x,vs) needs a sorted array of values. So we have to sort x before we can use it.

```
>xs=sort(x);
```

Then we plot the empirical distribution and some density bars into one plot. Instead of a bar plot for the distribution we use a sawtooth plot this time.

```
>figure(2,1); ...
>figure(1); plot2d("empdist",-4,4;xs); ...
>figure(2); plot2d(histo(x,v=-4:0.2:4,<bar)); ...
>figure(0):
```

A scatter plot is easy to do in Euler with the usual point plot. The following graph shows that the X and X+Y are clearly positively correlated.

```
>x=normal(1,100); plot2d(x,x+rotright(x),>points,style=".."):
```

Often, we wish to compare two samples of different distributions. This can be done with a quantile-quantile-plot.

For a test, we try the student-t distribution and exponential distribution.

```
>x=randt(1,1000,5); y=randnormal(1,1000,mean(x),dev(x)); ...
>plot2d("x",r=6,style="--",yl="normal",xl="student-t",>vertical); ...
>plot2d(sort(x),sort(y),>points,color=red,style="x",>add):
```

The plot clearly shows that the normal distributed values tend to be smaller at the extreme ends.

If we have two distributions of different size, we can expand the smaller one or shrink the larger one. The following function is good for both. It takes the median values with percentages between 0 and 1.

```
>function medianexpand (x,n) := median(x,p=linspace(0,1,n-1));
```

Let us compare two equal distributions.

```
>x=random(1000); y=random(400); ...
>plot2d("x",0,1,style="--"); ...
>plot2d(sort(medianexpand(x,400)),sort(y),>points,color=red,style="x",>add):
```

Regression and Correlation

Linear regression can be done with the functions polyfit() or various fit functions.

For a start we find the regression line for univariate data with polyfit(x,y,1).

```
>x=1:10; y=[2,3,1,5,6,3,7,8,9,8]; writetable(x' | y',labc=["x","y"])
```

x	y
1	2
2	3
3	1
4	5
5	6
6	3
7	7
8	8
9	9
10	8

We want to compare non-weighted and weighted fits. First the coefficients of the linear fit.

```
>p=polyfit(x,y,1)
```

```
[0.733333, 0.812121]
```

Now the coefficients with a weight that emphasizes the last values.

```
>w &= "exp(-(x-10)^2/10)"; pw=polyfit(x,y,1,w=w(x))
```

```
[4.71566, 0.38319]
```

We put everything into one plot for the points and the regression lines, and for the weights used.

```
>figure(2,1); ...
>figure(1); statplot(x,y,"b",xl="Regression"); ...
> plot2d("evalpoly(x,p)",>add,color=blue,style="--"); ...
> plot2d("evalpoly(x,pw)",5,10,>add,color=red,style="--"); ...
>figure(2); plot2d(w,1,10,>filled,style="/",fillcolor=red,xl=w); ...
>figure(0):
```

For another example we read a survey of students, their ages, the ages of their parents and the number of siblings from a file.

This table contains "m" and "f" in the second column. We use the variable tok2 to set the proper translations instead of letting readtable() collect the translations.

```
>{MS,hd}:=readtable("table1.dat",tok2:=[ "m", "f" ]); ...  
>writetable(MS,labc=hd,tok2:=[ "m", "f" ]);
```

Person	Sex	Age	Mother	Father	Siblings
1	m	29	58	61	1
2	f	26	53	54	2
3	m	24	49	55	1
4	f	25	56	63	3
5	f	25	49	53	0
6	f	23	55	55	2
7	m	23	48	54	2
8	m	27	56	58	1
9	m	25	57	59	1
10	m	24	50	54	1
11	f	26	61	65	1
12	m	24	50	52	1
13	m	29	54	56	1
14	m	28	48	51	2
15	f	23	52	52	1
16	m	24	45	57	1
17	f	24	59	63	0
18	f	23	52	55	1
19	m	24	54	61	2
20	f	23	54	55	1

How do the ages depend on each other? A first impression comes from a pairwise scatterplot.

```
>scatterplots(tablecol(MS,3:5),hd[3:5]):
```

It is clear that the age of the father and mother depend on each other. Let us determine and plot the regression line.

```
>cs:=MS[,4:5]'; ps:=polyfit(cs[1],cs[2],1)
```

```
[17.3789, 0.740964]
```

This is obviously the wrong model. The regression line would be $s=17+0.74t$, where t is the age of the mother and s the age of the father. The age difference may depend a little bit on the age, but not that much.

Rather, we suspect a function like $s=a+t$. Then a is the mean of the $s-t$. It is the average age difference between fathers and mothers.

```
>da:=mean(cs[2]-cs[1])
```

Let us plot this into one scatter plot.

```
>plot2d(cs[1],cs[2],>points); ...
>plot2d("evalpoly(x,ps)",color=red,style=".",>add); ...
>plot2d("x+da",color=blue,>add);
```

Here is a box plot of the two ages. This only shows, that the ages are different.

```
>boxplot(cs, ["mothers","fathers"]):
```

It is interesting that the difference in medians is not as large as the difference in means.

```
>median(cs[2])-median(cs[1])
```

1.5

The correlation coefficient suggests a positive correlation.

```
>correl(cs[1],cs[2])
```

0.7588307236

The correlation of the ranks is a measure for the same order in both vectors. It is also quite positive.

```
>rankcorrel(cs[1],cs[2])
```

0.758925292358

Creating new Functions

Of course, the EMT language can be used to program new functions. E.g., we define the skewness function.

$$\text{sk}(x) = \frac{\sqrt{n} \sum_i (x_i - m)^3}{(\sum_i (x_i - m)^2)^{3/2}}$$

where m is the mean of x .

```
>function skew (x:vector) ...
m=mean(x);
return sqrt(cols(x))*sum((x-m)^3)/(sum((x-m)^2))^(3/2);
endfunction
```

As you see, we can easily use the matrix language to get a very short and efficient implementation. Let us try this function.

```
>data=normal(20); skew(normal(10))
```

```
-0.442260891308
```

Here is another function, called the Pearson skewness coefficient.

```
>function skew1 (x) := 3*(mean(x)-median(x))/dev(x)
>skew1(data)
```

```
-0.0512737797453
```

Monte Carlo Simulation

Euler can be used to simulate random events. We have already seen simple examples above. Here is another one, which simulates 1000 times 3 dice throws, and asks for the distribution of the sums.

```
>ds:=sum(intrandom(1000,3,6)'); fs=getmultiplicities(3:18,ds)
```

```
[2, 13, 33, 44, 61, 109, 126, 125, 118, 114, 92, 76, 41, 36, 9, 1]
```

We can plot this now.

```
>columnsplot(fs,lab=3:18):
```

To determine the expected distribution is not so easy. We use an advanced recursion for this.

The following function counts the number of ways the number k can be represented as the sum of n numbers in the range of 1 to m. It works recursively in an obvious way.

```
>function map countways (k; n, m) ...
if n==1 then return k>=1 && k<=m
else
  sum=0;
  loop 1 to m; sum=sum+countways(k-#,n-1,m); end;
  return sum;
end;
endfunction
```

Here is the result for three throws of dices.

```
>countways(5:25,5,5)
```

```
[1, 5, 15, 35, 70, 121, 185, 255, 320, 365, 381, 365, 320, 255, 185, 121,
```

```
>cw=countways(3:18,3,6)
```

```
[1, 3, 6, 10, 15, 21, 25, 27, 27, 25, 21, 15, 10, 6, 3, 1]
```

We add the expected values to the plot.

```
>plot2d(cw/6^3*1000,>add); plot2d(cw/6^3*1000,>points,>add):
```

For another simulation, the deviation of the mean value of n 0-1-normal distributed random variables is $1/\sqrt{n}$.

```
>longformat; 1/sqrt(10)
```

```
0.316227766017
```

Let us check this with a simulation. We produce 10000 times 10 random vectors.

```
>M=normal(10000,10); dev(mean(M)')
```

```
0.312808356044
```

```
>plot2d(mean(M)',>distribution):
```

The median of 10 0-1-normal distributed random numbers has a larger deviation.

```
>dev(median(M)')
```

```
0.365189017877
```

Since we can easily generate random walks, we can simulate the Wiener process. We take 1000 steps of 1000 processes. We then plot the standard deviation and the mean of the n -th step of these processes together with the expected values in red.

```
>n=1000; m=1000; M=cumsum(normal(n,m)/sqrt(m)); ...
>t=(1:n)/n; figure(2,1); ...
>figure(1); plot2d(t,mean(M)'), plot2d(t,0,color=red,>add); ...
>figure(2); plot2d(t,dev(M)'), plot2d(t,sqrt(t),color=red,>add); ...
>figure(0):
```

Tests

Tests are an important tool in statistics. In Euler, many tests are implemented. All of these tests returns the error that we accept if we reject the zero hypothesis.

For an example, we test dice throws for uniform distribution. At 600 throws, we got the following values, which we plug into the chi-square test.

```
>chitest([90,103,114,101,103,89],dup(100,6)')
```

```
0.498830517952
```

The chi-square test also has a mode, which uses a Monte Carlo simulation to test the statistics. The result should be almost the same. The parameter `>p` interprets the y-vector as a vector of probabilities.

```
>chitest([90,103,114,101,103,89],dup(1/6,6)',>p,>montecarlo)
```

0.529

This error is much too large. So we cannot reject uniform distribution. This does not prove that our dice was fair. But we cannot reject our hypothesis.

Next we generate 1000 dice throws using the random number generator, and do the same test.

```
>n=1000; t=random([1,n*6]); chitest(count(t*6,6),dup(n,6)')
```

0.471366391089

Let us test for the mean value 100 with the t-test.

```
>s=200+normal([1,100])*10; ...
>ttest(mean(s),dev(s),100,200)
```

0.226642848601

The function `ttest()` needs the mean value, the deviation, the number of data, and the mean value to test for.

Now let us check two measurements for the same mean. We reject the hypothesis that they have the same mean, if the result is <0.05.

```
>tcomparedata(normal(1,10),normal(1,10))
```

0.122700802664

If we add a bias to one distribution, we get more rejections. Repeat this simulation several times to see the effect.

```
>tcomparedata(normal(1,10),normal(1,10)+2)
```

6.05989394731e-06

In the next example, we generate 20 random dice throws 100 times and count the ones in it. There must be $20/6=3.3$ ones on average.

```
>R=random(100,20); R=sum(R*6<=1)'; mean(R)
```

3.39

We now compare the number of ones with the binomial distribution. First we plot the distribution of ones.

```
>plot2d(R,distribution=max(R)+1,even=1,style="\\"/"):
>t=count(R,21);
```

Then we compute the expected values.

```
>n=0:20; b=bin(20,n)*(1/6)^n*(5/6)^(20-n)*100;
```

We have to collect several numbers to get categories, which are big enough.

```
>t1=sum(t[1:2])|t[3:7]|sum(t[8:21]); ...
>b1=sum(b[1:2])|b[3:7]|sum(b[8:21]);
```

The chi-square test rejects the hypothesis that our distribution is a binomial distribution, if its result is <0.05.

```
>chisqtest(t1,b1)
```

0.216232784449

The following example contains results of two groups of persons (male and female, say) voting for one out of six parties.

```
>A=[23,37,43,52,64,74;27,39,41,49,63,76]; ...
> writetable(A,wc=6,labr=["m","f"],labc=1:6)
```

	1	2	3	4	5	6
m	23	37	43	52	64	74
f	27	39	41	49	63	76

We wish to test for independence of the votes from the sex. The χ^2 table test does this. The result is way to large to reject independence. So we cannot say, if the voting depends on the sex from these data.

```
>tabletest(A)
```

0.990701632326

The following is the expected table, if we assume the observed frequencies of voting.

```
>writetable(expectedtable(A),wc=6,dc=1,labr=["m","f"],labc=1:6)
```

	1	2	3	4	5	6
m	24.9	37.9	41.9	50.3	63.3	74.7
f	25.1	38.1	42.1	50.7	63.7	75.3

We can compute the corrected contingency coefficient. Since it very close to 0, we conclude that the voting does not depend on the sex.

```
>contingency(A)
```

0.0427225484717

Next we use a variance analysis (F-test) to test three samples of normally distributed data for same mean value. The method is called ANOVA (analysis of variance). In Euler, the function varanalysis() is used.

```
>x1=[109,111,98,119,91,118,109,99,115,109,94]; mean(x1),
```

106.545454545

```
>x2=[120,124,115,139,114,110,113,120,117]; mean(x2),
```

119.111111111

```
>x3=[120,112,115,110,105,134,105,130,121,111]; mean(x3)
```

116.3

```
>varanalysis(x1,x2,x3)
```

0.0138048221371

This means, we reject the hypothesis of same mean value. We do this with an error probability of 1.3%.

There is also the median test, which rejects data samples with different mean distribution testing the median of the united sample.

```
>a=[56,66,68,49,61,53,45,58,54];
>b=[72,81,51,73,69,78,59,67,65,71,68,71];
>mediantest(a,b)
```

0.0241724220052

Another test on equality is the rank test. It is much sharper than the median test.

```
>ranktest(a,b)
```

0.00199969612469

In the following example, both distributions have the same mean.

```
>ranktest(random(1,100),random(1,50)*3-1)
```

0.153721132329

Let us now try to simulate two treatments a and b applied to different persons.

```
>a=[8.0,7.4,5.9,9.4,8.6,8.2,7.6,8.1,6.2,8.9];  
>b=[6.8,7.1,6.8,8.3,7.9,7.2,7.4,6.8,6.8,8.1];
```

The signum test decides, if a is better than b.

```
>signtest(a,b)
```

0.0546875

This is too much of an error. We cannot reject that a is as good as b.

The Wilcoxon test is sharper than this test, but relies on the quantitative value of the differences.

```
>wilcoxon(a,b)
```

0.0296680599405

Let us try two more tests using generated series.

```
>wilcoxon(normal(1,20),normal(1,20)-1)
```

0.000445901348481

```
>wilcoxon(normal(1,20),normal(1,20))
```

0.814869364774

Random Numbers

The following is a test for the random number generator. Euler uses a very good generator, so we need not expect any problems.

First we generate ten millions of random numbers in [0,1].

```
>n:=10000000; r:=random(1,n);
```

Next we count the distances between two numbers less than 0.05.

```
>a:=0.05; d:=differences(nonzeros(r<a));
```

Finally, we plot the number of times, each distance occurred, and compare with the expected value.

```
>m=getmultiplicities(1:100,d); plot2d(m); ...  
> plot2d("n*(1-a)^(x-1)*a^2",color=red,>add):
```

Clear the data.

```
>remvalue n;
```

Introduction for Users of the R Project

Clearly, EMT is not competing with R as a statistical package. However, there are many statistical procedures and functions available in EMT too. So EMT may satisfy the basic needs. After all, EMT comes with numerical packages and a computer algebra system.

This notebook is for you if you are familiar with R, but need to know the differences of the syntax of EMT and R. We try to give an overview of obvious and less obvious things you need to know.

Moreover, we look at ways to exchange data between the two systems.

Note that this is a work in progress.

Basic Syntax

The first thing you learn in R is to make a vector. In EMT, the main difference is that the : operator can take a step size. Moreover it has a low binding power.

```
>n=10; 0:n/20:n-1
```

```
[0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6, 6.5, 7, 7.5, 8, 8.5]
```

The c() function does not exist. It is possible to use vectors to concatenate things.

The following example is, like many others, from the "Introduction to R" that comes with the R project. If you read this PDF, you will find that I follow its path in this tutorial.

```
>x=[10.4, 5.6, 3.1, 6.4, 21.7]; [x,0,x]
```

```
[10.4, 5.6, 3.1, 6.4, 21.7, 0, 10.4, 5.6, 3.1, 6.4, 21.7]
```

The colon operator with step size of EMT is replaced by the function seq() in R. We can write this function in EMT.

```
>function seq(a,b,c) := a:b:c; ...  
>seq(0,-0.1,-1)
```

```
[0, -0.1, -0.2, -0.3, -0.4, -0.5, -0.6, -0.7, -0.8, -0.9, -1]
```

The function rep() of R is not present in EMT. For vector input, it could be written as follows.

```
>function rep(x:vector,n:index) := flatten(dup(x,n)); ...  
>rep(x,2)
```

```
[10.4, 5.6, 3.1, 6.4, 21.7, 10.4, 5.6, 3.1, 6.4, 21.7]
```

Note that "=" or ":=" is used for assignments. The "->" operator is used for units in EMT.

```
>125km -> " miles"
```

```
77.6713990297 miles
```

The "`<-`" operator for assignment is misleading anyway, and not a good idea of R. The following will compare `a` and `-4` in EMT.

$$>a=2; \quad a < -4$$

0

In R, "a<-4<3" works, but "a<-4<-3" does not. I had similar ambiguities in EMT too, but tried to eliminate them by and by.

EMT and R have vectors of boolean type. But in EMT, the numbers 0 and 1 are used to represent false and true. In R, the values true and false can nevertheless be used in ordinary arithmetic just like in EMT.

$$x > 5, \quad \% * x$$

$$\begin{bmatrix} 0, & 0, & 1, & 0, & 0 \\ 0, & 0, & 3.1, & 0, & 0 \end{bmatrix}$$

EMT throws errors or yields NAN depending on the flag "errors".

```
>errors off; 0/0, isNaN(sqrt(-1)), errors on;
```

NAN
1

Strings are the same in R and EMT. Both are in the current locale, not in Unicode.

In R there are packages for Unicode. In EMT, a string can be Unicode string. A unicode string can be translated to the local encoding and vice versa. Moreover, u"..." can contain HTML entities.

>u" ; Ren  Grothmann"

© René Grothmann

The following may or may not display correctly on your system as A with dot and dash above it. It depends on the font you are using.

```
>chartoutf([480])
```

The string concatenation is done with "+" or "|". It can include numbers, which will print in the current format.

```
>"pi = "+pi
```

pi = 3.14159265359

Indexing

Most of the time, this will work as in R.

But EMT will interpret negative indices from the back of the vector, while R interprets $x[n]$ as x without the n -th elements.

```
>x, x[1:3], x[-2]
```

```
[10.4, 5.6, 3.1, 6.4, 21.7]
[10.4, 5.6, 3.1]
6.4
```

The behavior of R can be achieved in EMT with drop().

```
>drop(x, 2)
```

```
[10.4, 3.1, 6.4, 21.7]
```

Logical vectors are not treated differently as index in EMT, in contrast to R. You need to extract the non-zero elements first in EMT.

```
>x, x>5, x[nonzeros(x>5)]
```

```
[10.4, 5.6, 3.1, 6.4, 21.7]
[1, 1, 0, 1, 1]
[10.4, 5.6, 6.4, 21.7]
```

Just as in R, the index vector can contain repetitions.

```
>x[[1,2,2,1]]
```

```
[10.4, 5.6, 5.6, 10.4]
```

But names for indices are not possible in EMT. For a statistical package, this may often be necessary to ease access to elements of vectors.

To mimic this behavior, we can define a function as the following.

```
>function sel (v,i,s) := v[indexof(s,i)]; ...
>s=["first","second","third","fourth"]; sel(x,[ "first","third"],s)
```

```
[10.4, 3.1]
```

Data Types

EMT has more fixed data types than R. Obviously, in R there exist growing vectors. You can set an empty numerical vector v and assign a value to the element $v[17]$. This is not possible in EMT.

The following is a bit inefficient.

```
>v=[]; for i=1 to 10000; v=v|i; end;
```

EMT will now construct a vector with v and i appended on the stack and copy that vector back to the global variable v.

The more efficient pre-defines the vector.

```
>v=zeros(10000); for i=1 to 10000; v[i]=i; end;
```

To change date types in EMT, you can use functions like complex().

```
>complex(1:4)
```

```
[ 1+0i , 2+0i , 3+0i , 4+0i ]
```

Conversions to strings is possible for elementary data types only. The current format is used for simple string concatenation. But there are functions like print() or frac().

For vectors, you can easily write your own function.

```
>function tostr (v) ...
```

```
s="[";  
loop 1 to length(v);  
  s=s+print(v[#],2,0);  
  if #<length(v) then s=s+","; endif;  
end;  
return s+"]";  
endfunction
```

```
>tostr(linspace(0,1,10))
```

```
[0.00,0.10,0.20,0.30,0.40,0.50,0.60,0.70,0.80,0.90,1.00]
```

For communication with Maxima, there exists a function convertm xm(), which can also be used to format a vector for output.

```
>convertm xm(1:10)
```

```
[1,2,3,4,5,6,7,8,9,10]
```

For Latex the tex command can be used to obtain the Latex command.

```
>tex(&[1,2,3])
```

```
\left[ 1 , 2 , 3 \right]
```

Factors and Tables

In the introduction to R there is an example with so called factors.

The following is a list of the territories of 30 states.

```
>austates = ["tas", "sa", "qld", "nsw", "nsw", "nt", "wa", "wa", ...
>"qld", "vic", "nsw", "vic", "qld", "qld", "sa", "tas", ...
>"sa", "nt", "wa", "vic", "qld", "nsw", "nsw", "wa", ...
>"sa", "act", "nsw", "vic", "vic", "act"];
```

Assume, we have corresponding incomes in each state.

```
>incomes = [60, 49, 40, 61, 64, 60, 59, 54, 62, 69, 70, 42, 56, ...
>61, 61, 58, 51, 48, 65, 49, 49, 41, 48, 52, 46, ...
>59, 46, 58, 43];
```

Now, we want to compute the mean of incomes in the territories. Being a statistical program, R has factor() and tapply() for this.

EMT can make this by finding the index of territories in the unique list of territories.

```
>auterr=sort(unique(austates)); f=indexofsorted(auterr,austates)
```

```
[6, 5, 4, 2, 2, 3, 8, 8, 4, 7, 2, 7, 4, 4, 5, 6, 5, 3, 8, 7, 4, 2,
5, 1, 2, 7, 7, 1]
```

At that point, we can write our own loop function to do things for one factor only.

Or we can mimic the tapply() function in the following way.

```
>function map_tappl (i; f$call, cat, x) ...
```

```
u=sort(unique(cat));
f=indexof(u,cat);
return f$(x[nonzeros(f==indexof(u,i))]);
endfunction
```

It is a bit inefficient, since it computes The unique territories for each i, but it works.

```
>tappl(auterr,"mean",austates,incomes)
```

```
[44.5, 57.3333333333, 55.5, 53.6, 55, 60.5, 56, 52.25]
```

Note that it works for each vector of territories.

```
>tappl(["act","nsw"],"mean",austates,incomes)
```

```
[44.5, 57.3333333333]
```

Now, the statistical package of EMT defines tables just as in R. The functions readtable() and writetable() can be used for input and output.

So we can print the average state income in the territories in a friendly way.

```
>writetable(tappl(auterr,"mean",austates,incomes),labc=auterr,wc=7)
```

act	nsw	nt	qld	sa	tas	vic	wa
44.5	57.33	55.5	53.6	55	60.5	56	52.25

We can also try to mimic the behavior of R completely.

The factors should clearly be kept in a collection with the types and the categories (states and territories in our example). For EMT, we add the pre-computed indices.

```
>function makef (t) ...
```

```
## Factor data
## Returns a collection with data t, unique data, indices.
## See: tapply
u=sort(unique(t));
return ({t,u,indexofsorted(u,t)}};
endfunction
```

```
>statef=makef(austates);
```

Now the third element of the collection will contain the indices.

```
>statef[3]
```

```
[6, 5, 4, 2, 2, 3, 8, 8, 4, 7, 2, 7, 4, 4, 5, 6, 5, 3, 8, 7, 4, 2,
5, 1, 2, 7, 7, 1]
```

Now we can mimic tapply() in the following way. It will return a table as a collection of table data and column headings.

```
>function tapply (t:vector,tf,f$:call) ...
```

```
## Makes a table of data and factors
## tf : output of makef()
## See: makef
uf=tf[2]; f=tf[3]; x=zeros(length(uf));
for i=1 to length(uf);
  ind=nonzeros(f==i);
  if length(ind)==0 then x[i]=NAN;
  else x[i]=f$(t[ind]);
  endif;
end;
return {{x,uf}};
endfunction
```

We did not add much type checking here. The only precaution concerns categories (factors) with no data. But one should check for the correct length of t and for the correctness of the collection tf.

This table can be printed as a table with writetable().

```
>writetable(tapply(incomes,statef,"mean"),wc=7)
```

act	nsw	nt	qld	sa	tas	vic	wa
44.5	57.33	55.5	53.6	55	60.5	56	52.25

Arrays

EMT has only two dimensions for arrays. The data type is called a matrix. It would be easy to write functions for higher dimensions or a C library for this, however.

R has more than two dimensions. In R the array is a vector with a dimension field.

In EMT, a vector is a matrix with one row. It can be made into a matrix with redim().

```
>shortformat; X=redim(1:20,4,5)
```

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20

Extraction of rows and columns, or sub-matrices, is much like in R.

```
>X[,2:3]
```

2	3
7	8
12	13
17	18

However, in R it is possible to set a list of specific indices of the vector to a value. The same is possible in EMT only with a loop.

```
>function setmatrixvalue (M, i, j, v) ...  
  
loop 1 to max(length(i),length(j),length(v))  
    M[i{#},j{#}] = v{#};  
end;  
endfunction
```

We demonstrate this to show that matrices are passed by reference in EMT. If you do not want to change the original matrix M, you need to copy it in the function.

```
>setmatrixvalue(X,1:3,3:-1:1,0); X,
```

1	2	0	4	5
6	0	8	9	10
0	12	13	14	15
16	17	18	19	20

The outer product in EMT can only be done between vectors. It is automatic due to the matrix language. One vector needs to be a column vector and the other a row vector.

```
>(1:5)*(1:5)'
```

1	2	3	4	5
2	4	6	8	10
3	6	9	12	15
4	8	12	16	20
5	10	15	20	25

In the introduction PDF for R there is an example, which computes the distribution of ab-cd for a,b,c,d chosen from 0 to n randomly. The solution in R is form a 4-dimensional matrix and run table() over it.

Of course, this can be achieved with a loop. But loops are not effective in EMT or R. In EMT, we could write the loop in C and that would be the quickest solution.

But we want to mimic the behavior of R. For this, we need to flatten the multiplications ab and make a matrix of ab-cd.

```
>a=0:6; b=a'; p=flatten(a*b); q=flatten(p-p'); ...
>u=sort(unique(q)); f=getmultiplicities(u,q); ...
>statplot(u,f,"h"):
```

Besides the exact multiplicities, EMT can compute frequencies in vectors.

```
>getfrequencies(q,-50:10:50)
```

```
[0, 23, 132, 316, 602, 801, 333, 141, 53, 0]
```

The most easy way to plot this as a distribution is the following.

```
>plot2d(q,distribution=11):
```

But it is also possible to pre-compute the count in chosen intervals beforehand. Of course, the following uses getfrequencies() internally.

Since the histo() function returns frequencies, we need to scale these so that the integral under the bar graph is 1.

```
>{x,y}=histo(q,v=-55:10:55); y=y/sum(y)/differences(x); ...
>plot2d(x,y,>bar,style="/"):
```

Lists

EMT has two sorts of lists. One is a global list which is mutable, and the other is a list type which is immutable. We do not care about global lists here.

The immutable list type is called a collection in EMT. It behaves like a structure in C, but the elements are just numbered and not named.

```
>L={"Fred","Flintstone",40,[1990,1992]}{}
```

```
Fred  
Flintstone  
40  
[1990, 1992]
```

Currently the elements do not have names, though names can be set for special purposes. They are accessed by numbers.

```
>(L[4])[2]
```

```
1992
```

File Input and Output (Reading and Writing Data)

You will often want to import a matrix of data from other sources to EMT. This tutorial tells you about the many ways to achieve this. Simple functions are writematrix() and readmatrix().

Let us demonstrate how to read and write a vector of reals to a file.

```
>a=random(1,100); mean(a), dev(a),
```

```
0.55269  
0.29344
```

To write the data to a file, we use the function writematrix().

Since this introduction is most likely in a directory, where the user has no write access, we write the data to the user home directory. For own notebooks, this is not necessary, since the data file will be written into the same directory.

```
>filename="test.dat";
```

Now we write the column vector a' to the file. This yields one number in each line of the file.

```
>writematrix(a',filename);
```

To read the data, we use readmatrix().

```
>a=readmatrix(filename)';
```

And remove the file.

```
>fileremove(filename);  
>mean(a), dev(a),
```

```
0.55269  
0.29344
```

The functions `writematrix()` or `writetable()` can be configured for other languages.

E.g., if you have an Indonesian system (decimal point with comma), your Excel needs values with decimal commas separated by semicolons in a csv (the default is comma separated values) file. The following file "test.csv" should appear on your current folder.

```
>filename="test.csv"; ...
>writematrix(random(5,3),file=filename,separator=",");
```

You can now open this file with an Indonesian Excel directly.

```
>fileremove(filename);
```

Sometimes we have strings with tokens like the following.

```
>s1:="f m m f m m m f f f m m f"; ...
>s2:="f f f m m f f";
```

To tokenize this, we define a vector of tokens.

```
>tok:=[ "f", "m" ]
```

f
m

Then we can count the number of times each token appears in the string, and put the result into a table.

```
>M:=getmultiplicities(tok,strtokens(s1))_ ...
>  getmultiplicities(tok,strtokens(s2));
```

Write the table with the token headers.

```
>writetable(M,labc=tok,labr=1:2,wc=8)
```

	f	m
1	6	7
2	5	2

For statics, EMT can read and write tables.

```
>file="test.dat"; open(file,"w"); ...
>writeln("A,B,C"); writematrix(random(3,3)); ...
>close();
```

The file looks like this.

```
>printfile(file)
```

```
A,B,C  
0.870270022178512,0.02481087162574944,0.466040156601655  
0.3026325231063014,0.6220889584522515,0.9367082489748491  
0.7466896392724567,0.7368844620115959,0.3419781863075429
```

The function `readtable()` in its simplest form can read this and return a collection of values and heading lines.

```
>L=readtable(file,>list);
```

This collection can be printed with `writetable()` to the notebook, or to a file.

```
>writetable(L,wc=10,dc=5)
```

A	B	C
0.87027	0.02481	0.46604
0.30263	0.62209	0.93671
0.74669	0.73688	0.34198

The matrix of values is the first element of L. Note that `mean()` in EMT computes the mean values of the rows of a matrix.

```
>mean(L[1])
```

```
0.45371  
0.62048  
0.60852
```

CSV Files

First, let us write a matrix into a file. For the output, we generate a file in the current working directory.

```
>file="test.csv"; ...  
>M=random(3,3); writematrix(M,file);
```

Here is the content of this file.

```
>printfile(file)
```

```
0.8644485191805866,0.823403481389068,0.790241662026635  
0.5906186300341895,0.1676904904906134,0.6263113968244143  
0.2579517401409792,0.8455384857350826,0.721718522415531
```

This CVS can be opened on English systems into Excel by a double click. If you get such a file on a German system, you need to import the data into Excel taking care of the decimal dot.

But the decimal dot is the default format for EMT too. You can read a matrix from a file with readmatrix().

```
>readmatrix(file)
```

```
0.86445    0.8234    0.79024  
0.59062    0.16769   0.62631  
0.25795    0.84554   0.72172
```

It is possible to write several matrices to one file. The open() command can open a file for writing with the "w" parameter. The default is "r" for reading.

```
>open(file,"w"); writematrix(M); writematrix(M'); close();
```

The matrices are separated by a blank line. To read the matrices, open the file and call readmatrix() several times.

```
>open(file); A=readmatrix(); B=readmatrix(); A==B, close();
```

```
1      0      0  
0      1      0  
0      0      1
```

In Excel or similar spreadsheets, you can export a matrix as CSV (comma separated values). In Excel 2007, use "save as" and "other formats", then select "CSV". Make sure, the current table contains only data you wish to export.

Here is an example.

```
>printfile("excel-data.csv")
```

```
0;1000;1000  
1;1051,271096;1072,508181  
2;1105,170918;1150,273799  
3;1161,834243;1233,67806  
4;1221,402758;1323,129812  
5;1284,025417;1419,067549  
6;1349,858808;1521,961556  
7;1419,067549;1632,31622  
8;1491,824698;1750,6725  
9;1568,312185;1877,610579  
10;1648,721271;2013,752707
```

As you can see, my German system has used a semicolon as separator and a decimal comma. You can change this in the system settings or in Excel, but it is not necessary for reading the matrix into EMT.

The easiest way to read this into Euler is readmatrix(). All commas are replaced by dots with the parameter >comma. For English CSV, simply omit this parameter.

```
>M=readmatrix("excel-data.csv",>comma)
```

0	1000	1000
1	1051.3	1072.5
2	1105.2	1150.3
3	1161.8	1233.7
4	1221.4	1323.1
5	1284	1419.1
6	1349.9	1522
7	1419.1	1632.3
8	1491.8	1750.7
9	1568.3	1877.6
10	1648.7	2013.8

Let us plot this.

```
>plot2d(M' [1],M' [2:3],>points,color=[red,green]'):
```

There are more elementary ways to read data from a file. You can open the file and read the numbers line by line. The function getvectorline() will read numbers from a line of data. By default, it expects a decimal dot. But it can also use a decimal comma, if you call setdecimaldot(",") before you use this function.

The following function is an example for this. It will stop at the end of the file or an empty line.

```
>function myload (file) ...
```

```
open(file);
M=[];
repeat
    until eof();
    v=getvectorline(3);
    if length(v)>0 then M=M_v; else break; endif;
end;
return M;
close(file);
endfunction
```

```
>myload(file)
```

0.86445	0.8234	0.79024
0.59062	0.16769	0.62631
0.25795	0.84554	0.72172

It would also be possible to read all numbers in that file with getvector().

```
>open(file); v=getvector(10000); close(); redim(v[1:9],3,3)
```

0.86445	0.8234	0.79024
0.59062	0.16769	0.62631
0.25795	0.84554	0.72172

Thus it is very easy to save a vector of values, one value in each line and read back this vector.

```
>v=random(1000); mean(v)
```

0.51267

```
>writematrix(v',file); mean(readmatrix(file)')
```

0.51267

Using Tables

Tables can be used to read or write numerical data. For an example, we write a table with row and column headers to a file.

```
>file="test.tab"; M=random(3,3); ...
>open(file,"w"); ...
>writetable(M,separator=",",labc=["one","two","three"]); ...
>close(); ...
>printfile(file)
```

one, two, three		
0.91,	0.19,	0.01
0.9,	0.05,	0.94
0.16,	0.81,	0.37

This can be imported into Excel.

To read the file in EMT, we use `readtable()`.

```
>{M,headings}=readtable(file,>clabs); ...
>writetable(M,labc=headings)
```

one	two	three
0.91	0.19	0.01
0.9	0.05	0.94
0.16	0.81	0.37

Analyzing a Line

You could even evaluate each line by hand. Suppose, we have a line of the following format.

```
>line="2020-11-03,Tue,1'114.05"
```

2020-11-03,Tue,1'114.05

First we can tokenize the line.

```
>vt=strtoks(line)
```

2020-11-03

Tue

1'114.05

Then we can evaluate each element of the line using appropriate evaluations.

```
>day(vt[1]), ...
>indexof(["mon","tue","wed","thu","fri","sat","sun"],tolower(vt[2])), ...
>strrepl(vt[3], "'", "")()
```

7.3816e+05

2

1114

Using regular expressions, it is possible to extract almost any information from a line of data.

Assume we have the following line an HTML document.

```
>line="<tr><td>1145.45</td><td>5.6</td><td>-4.5</td><tr>"
```

```
<tr><td>1145.45</td><td>5.6</td><td>-4.5</td><tr>
```

To extract this, we use a regular expression, which searches for

- a closing bracket >,
- any string not containing brackets with a sub-match "(...)",
- an opening and a closing bracket using the shortest solution,
- again any string not containing brackets,
- and an opening bracket <.

Regular expressions are somewhat difficult to learn but very powerful.

```
>{pos,s,vt}=strxfind(line,>([<>]+)<.+?>([<>]+)<" );
```

The result is the position of the match, the matched string, and a vector of strings for sub-matches.

```
>for k=1:length(vt); vt[k](), end;
```

1145.5

5.6

Here is a function, which reads all numerical items between <td> and </td>.

```
>function readtd (line) ...
v=[]; cp=0;
repeat
{pos,s,vt}=strxfind(line,"<td.*?>(.+?)</td>",cp);
until pos==0;
if length(vt)>0 then v=v|vt[1]; endif;
cp=pos+strlen(s);
end;
return v;
endfunction
```

```
>readtd(line+"<td>non-numerical</td>")
```

```
1145.45
5.6
-4.5
non-numerical
```

Reading from the Web

A web site or a file with an URL can be opened in EMT and can be read line by line.

In the example, we read the current version from the EMT site. We use regular expression to scan for "Version ..." in a heading.

```
>function readversion () ...
urlopen("http://www.euler-math-toolbox.de/Programs/Changes.html");
repeat
until urleof();
s=urlgetline();
k=strfind(s,"Version ",1);
if k>0 then substring(s,k,strfind(s,<,k)-1), break; endif;
end;
urlclose();
endfunction
```

```
>readversion
```

```
Version 2021-04-30
```

Input and Output of Variables

You can write a variable in the form of an Euler definition to a file or to the command line.

```
>writevar(pi, "mypi");
```

```
mypi = 3.141592653589793;
```

For a test, we generate an Euler file in the work directory of EMT.

```
>file="test.e"; ...
>writevar(random(2,2), "M", file); ...
>printfile(file, 3)
```

```
M = [ ..
0.106576066957305, 0.3792882958611032;
0.7685505432667192, 0.8279275951785248];
```

We can now load the file. It will define the matrix M.

```
>load(file); show M,
```

```
M =
0.10658   0.37929
0.76855   0.82793
```

By the way, if writevar() is used on a variable, it will print the variable definition with the name of this variable.

```
>writevar(M); writevar(inch$)
```

```
M = [ ..
0.106576066957305, 0.3792882958611032;
0.7685505432667192, 0.8279275951785248];
inch$ = 0.0254;
```

We can also open a new file or append to an existing file. In the example we append to the previously generated file.

```
>open(file, "a"); ...
>writevar(random(2,2), "M1"); ...
>writevar(random(3,1), "M2"); ...
>close();
>load(file); show M1; show M2;
```

```
M1 =
0.87558   0.40272
0.010018   0.99217
M2 =
0.66841
0.91819
0.3323
```

To remove any files use fileremove().

```
>fileremove(file);
```

A row vector in a file does not need commas, if each number is in a new line. Let us generate such a file, writing every line one by one with writeln().

```
>open(file,"w"); writeln("M = ["); ...
>for i=1 to 5; writeln(""+random()); end; ...
>writeln("]"); close(); ...
>printfile(file)
```

```
M = [
0.699685893708
0.56495656726
0.566012751828
0.913964584324
0.75449923013
];
```

```
>load(file); M
```

```
[0.69969, 0.56496, 0.56601, 0.91396, 0.7545]
```