

Tugas Overleaf and Markdown

Emlys Jenny Ester Sirait emlys-jenny.2023

November 2024

1 Pendahuluan

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.

```
plot2d("sin(x)exp(-x^2)", -pi, pi, title = "Test", xl = "x", yl = "y") :
```

* 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 Tutorias and Example", 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 mengklik kanan teks atau tekan menu Edit -gt;

Edit Comment atau tekan [F5], dan juga baris perintah EMT yang ditandai dengan "gt;" 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* : $\text{integrate}(x^3, x) = \text{integrate}(x^3, x) + C$ - *http* : <http://www.euler-math-toolbox.de> - *See* : <http://www.google.de> | *Google* - *image* : [hati.png](#) - - -

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

Judul

Sub-Judul

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

maxima: $\text{integrate}(x^3, x) = \text{integrate}(x^3, x) + C$

<http://www.euler-math-toolbox.de> | <http://www.euler-math-toolbox.de>

<http://www.google.de> | [Google](#)

[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 //

S=[2,6,0,5;2,0,1,5]

2 6 0 5 2 0 1 5

V=[1,2,3;4,5,6]

1 2 3 4 5 6

A=[2,9,4;0,2,5]

2 9 4 0 2 5

T=[26,05,15;22,09,04]

26 5 15 22 9 4

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; pir2, 2pir
4.90873852123 7.85398163397
sin(180°), cos(pi), log(sqrt(E))
0 -1 0.5
```

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 -.

- * *^* lebih kuat dari *, sehingga r^2 merupakan rumus luas lingkaran.

- * Jika perlu, Anda harus menambahkan tanda kurung, seperti pada $2^{(2^3)}$.

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 °. 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 -gt; Reference)

Untuk membuat rangkaian komputasi lebih mudah, Anda dapat merujuk ke hasil sebelumnya dengan "perhitungan dalam baris perintah yang sama.

```
(sqrt(5)+1)/2,
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. * —

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 easy units.*

```
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 -gt;.

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 mengatrtampilan 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 12001.0310, longE, longestpi
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 "defformat" atau "reset".

```

longestformat; pi, defformat; 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  $ax^2 + bx + c = 0$ 
secaramanual... D =
sqrt(b2/(a24) - c/a); ... - b/(2a) + D, ... - b/(2a) - 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
r 1.25 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
km1000cm 0.01 mm0.001nm 1853.24496 gram0.001m 1 hquantum6.62606957e-
34atm 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 "inrandom" berikut ini!

```
inrandom(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,
0.095153, 0.595017]
```

```
normal(10)
```

```
[-0.495418, 1.6463, -0.390056, -1.98151, 3.44132, 0.308178, -0.733427, -0.526167,
1.10018, 0.108453]
```

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(w2)
```

```
0.35
```

Bilangan Kompleks

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

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 imajiner 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 imajiner x terlalu besar, hasilnya akan menampilkan pesan kesalahan.

```
gt;sqrt(-1) // Error! gt;sqrt(complex(-1))
```

```
z=2+3I, 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 "amp;". Setiap ekspresi yang dimulai dengan "amp;" adalah ekspresi simbolis dan dikerjakan oleh Maxima.

```
(a+b)2
2 (b + a)
expand((a+b)2), factor(x2 + 5x + 6)
2 2 b + 2 a b + a
(x + 2) (x + 3)
solve(ax2 + bx + c, x)//rumusabc
2 2 (- sqrt(b - 4 a c)) - b sqrt(b - 4 a c) - b [x = —————, x =
—————] 2 a 2 a
(a2 - b2)/(a + b), ratsimp(
2 2 a - b ———- b + a
a - b
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)
3 3 2 5
:: factor(20!)
18 8 4 2 2 3 5 7 11 13 17 19
```

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)!)
(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)//menyederhanakanfungsi trigonometri
```


$$4 \sin(x) + \cos(x) \longrightarrow -3 \cos(x)$$

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

$$p1 = (x^3 + 1)/(x + 1)$$

$$3x + 1 \longrightarrow x + 1$$

$$\text{ratsimp}(p1)$$

$$2x - x + 1$$

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

$$p1 \text{ with } x=3 \text{ // } (3^3 + 1)/(3 + 1)$$

$$7$$

$$p1 \text{ with } x=a+b, \text{ ratsimp}(\text{$$

$$3(b + a) + 1 \longrightarrow b + a + 1$$

$$22b + (2a - 1)b + a - a + 1$$

$$\text{diff}(p1, x) \text{ // turunan } p1 \text{ terhadap } x$$

$$233xx + 1 \longrightarrow - \longrightarrow x + 12(x + 1)$$

$$\text{integrate}(p1, x) \text{ // integral } p1 \text{ terhadap } x$$

$$322x - 3x + 6x \longrightarrow 6$$

Tampilan Matematika Simbolik dengan LaTeX

Anda dapat menampilkan hasil perhitungagn simbolik secara lebih bagus menggunakan LaTeX. Untuk melakukan hal ini, tambahkan tanda dolar (\$) *didepantandaamp; pada setiapperintah*

Perhatikan, hal ini hanya dapat menghasilkan tampilan yang diinginkan apabila komputer Anda sudah terpasang software LaTeX.

$$(a + b)^2$$

$$\text{expand}((a + b)^2), \text{factor}(x^2 + 5x + 6)$$

$$\text{solve}(ax^2 + bx + c, x) \text{ // rumus abc}$$

$$(a^2 - b^2)/(a + b), \text{ratsimp}(\text{$$

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.

2 Aljabar

Emlys Jenny E.S₂3030630081_{EMT4aljabar} NAMA : EMLYSJENNYESTERSIRAIT

NIM : 23030630081

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:

$$6x^6 - 3y^5 - 7x^2y^6 - 9$$

Menjabarkan:

$$\text{showev}(\text{'expand'}((6x^6 - 3) + y^5)(-7x^2 - y^6 - 9)))$$

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.

$$r:=2; h:=4; \pi r^2 h/3$$

$$16.7551608191$$

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

$$\pi^2 r h,$$

$$50.2654824574 \ 100.530964915$$

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

$$x := 1;$$

$$x := \cos(x) \ // \text{ nilai cosinus (x dalam radian)}$$

$$0.540302305868$$

$$x := \cos(x)$$

$$0.857553215846$$

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

$$x := 1.5; \dots \ x := (x+2/x)/2, \ x := (x+2/x)/2, \ x := (x+2/x)/2,$$

$$1.41666666667 \ 1.41421356237 \ 1.41421356237$$

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 "

```
// This line will not be visible once the cursor is off the line
```

Garis yang dimulai dengan

```
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=1; for i=1 to 5; x := (x+2/x)/2, end; // menghitung akar 2
```

```
1.5 1.41666666667 1.41421568627 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 Epi piE; then "Thoughtso!", 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(10°)/cos(20°))
```

```
0.429875017772
```

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(2.5))
```

```
2.5
```

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 $\text{rad}(x)$. Fungsi akar kuadrat disebut kuadrat dalam Euler. Tentu saja, $x^{(1/2)}$ *jugadimungkinkan*.

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; 1/2gt^2
30.65625
```

EMT menggunakan sintaks pemrograman untuk ekspresi. Memasuki

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+4log(0.6))+1/7)
8.77908249441
```

Untuk menghitung ekspresi rumit seperti

Anda harus memasukkannya dalam bentuk baris.

```
((1/7+1/8+2)/(1/3+1/2))^2pi
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* *dayauntukpositifaataubilanganbulatb(a**bjugaberfungsi)n!operatorfaktorial* dan masih banyak lagi.

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

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

```
w=[none]; w=v—v
```

```
sandy axel nando sandy axel nando
```

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 LaTeX. (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, 116,  
116, 101, 114]
```

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";semangka"
```

```
0 1
```

"dan" adalah operator `"amp;amp;"` 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 `isnan()` untuk mengekstrak elemen tertentu dari vektor. Dalam contoh, kami menggunakan `isprime(bersyarat(n))`.

```
N=2:3:99 // N berisi elemen 2 dan bilangan2 ganjil dari 3 s.d. 99
[2, 3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48, 51, 54, 57, 60, 63,
66, 69, 72, 75, 78, 81, 84, 87, 90, 93, 96, 99]
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, 67, 71, 73, 79,
83, 89, 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.

```
printhe(pi)
3.243F6A8885A30*160
```

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 `deformat()`.
`deformat;` // default

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

```
longest pi2/2
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:="pi2"; longest fx()`
12.56637061435917

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

```
fx:="asin(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("atx2", 3, 5) // computes 4 × 32 not 53
```

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("atx2", 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("atx^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.

```
function f()x := 5x;
```

```
f(7)
```

42

```
f(3)
```

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
```

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",
```

```
@(a,b) a^2 + b^240
```

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 - ax; ... 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.

Matematika Simbolik

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 amp;. Ekspresi apa pun yang dimulai dengan amp; adalah ekspresi simbolis. Itu dievaluasi dan dicetak oleh Maxima.

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

44!

Dengan cara ini, Anda dapat menghitung hasil yang besar dengan tepat. Mari kita hitung

$33!/(23!10!)/\text{nilai}C(33,10)$

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

$\text{binomial}(33,10)/\text{menghitung}C(33,10)\text{menggunakan fungsi}\text{binomial}()$

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.

$\text{binomial}(x,3)/C(x,3)$

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

$\text{binomial}(x,3)\text{with }x=10/\text{substitusikan }x=10\text{ ke }C(x,3)$

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 *didepan& (atau Anda dapat menghilangkanan* jika Anda tidak menginstal LaTeX.

$(3+x)/(x^2+1)$

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; v²"

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.

$\text{expand}((1+x)^4), \text{factor}(\text{diff}()$

Sekali lagi,

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

$\text{fx} = (x+1)/(x^4+1); \text{fx}$

Ekspresi simbolik dapat digunakan dalam ekspresi simbolik lainnya.

$\text{factor}(\text{diff}(\text{fx}, x))$

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!)
8 4 2 2 3 5 7
:: factor(20!)
18 8 4 2 2 3 5 7 11 13 17 19
```

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

```
::: av:ga2;
2 g
fx = x3exp(x),fx
3 x x E
```

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

```
(fx with x=5),
5 125 E
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(
10 5 1000 E - 125 E
2.20079141499189e+7
factor(diff(fx, x, 2))
```

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

```
tex(fx)
x3 ex
```

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).

```
fxwithx = 1/2
```

Penugasan juga bisa bersifat simbolis.

```
fxwithx = 1 + t
```

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

```
solve(x2 + x = 4, x)
```

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

```
solve("x2 + 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 membiarkan Maxima menemukan nilai numerik.

```
sol = solve(x2 + 2x = 4, x);sol, sol(), float(sol)
[-3.23607, 1.23607]
```

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

```
solve(x2 + x = 1, x), x2 = xwith
```

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

```
sol = solve([x+y=3, x2 + y2 = 5], [x, y]);sol, xywithsol[1]
```

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((x3 - 1)/(x + 1), x) // turunan bentuk pecahan
diff((x3 - 1)/(x + 1), x) | ratsimp // menyederhanakan pecahan
factor(
```

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) := xsqrt(x2 + 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, 1.21083, 1.41421]
```

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 diinteli Euler*.

```
function overwrite sin (x) := _sin(x°) // redine 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) := ax2
```

Menghilangkan parameter ini menggunakan nilai default.

```
f(3)
```

```
9
```

Menyetelnya akan menimpa nilai default.

```
f(7)
```

```
49
```

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

```
16
```

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

```
function f(x) := ax2
```

```
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 "amp;=". Mereka didefinisikan dalam Euler dan Maxima, dan bekerja di kedua dunia. Ekspresi yang mendefinisikan dijalankan melalui Maxima sebelum definisi.

```
function g(x) = x3 - exp(-x); g(x)
```

Fungsi simbolik dapat digunakan dalam ekspresi simbolik.

```
diff(g(x), x),
```

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
```

```
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) = (2x-1)n;P(x,n)
```

```
function Q(x,n) = (x+2)n;Q(x,n)
```

```
P(x,4),expand(
```

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

625

```
P(x,4) + Q(x,3),expand(
```

```
P(x,4) - Q(x,3),expand(
```

```
P(x,4)Q(x,3),expand(
```

```
P(x,4)/Q(x,1),expand(
```

```
function f(x) = x3 - x;f(x)
```

Dengan amp;= fungsinya simbolis, dan dapat digunakan dalam ekspresi simbolik lainnya.

```
integrate(f(x),x)
```

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

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("xx", 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(26.7, 2)
```

2 6.7

Selain itu, dimungkinkan untuk menggunakan parameter yang ditetapkan.

```
mylog(E2, 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]) = a2 + b2 - ab + b;f(a,b), f(x,y)
```

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

diff(expr, y, 2) + diff(expr, x, 2)
realpart(($x + Iy$)⁴),lapl(

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)

Untuk meringkas

* amp;= mendefinisikan fungsi simbolis,

* := mendefinisikan fungsi numerik,

* amp;amp;= 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("x² - 2", 1)

1.41421356237

Ini juga berfungsi untuk ekspresi simbolis. Ambil fungsi berikut.

solve(x² = 2, x)

solve(x² - 2, x)

solve(ax² + bx + c = 0, x)

solve([ax + by = c, dx + ey = f], [x, y])

px = 4x⁸ + x⁷ - x⁴ - x;px

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(x² - x - 1, x);sol

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".

x²withsol[1],expand(x² - x - 1withsol[2])

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

solve([x + y = 2, x³ + 2y + x = 4], [x, y])

Fungsi $f()$ dapat melihat variabel global. Namun seringkali kita ingin menggunakan parameter lokal.

dengan $a=3$.

```
function f(x,a) := xa - ax;
```

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("xa - ax", 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)`

```
load(fourier_elim)
```

```
C:/Program Files/Euler x64/maxima/share/maxima/5.35.1/share/fourier_elim/fourier_elim.lisp
```

```
fourier_elim([x2 - 1 0], [x])/x2 - 1 0
```

```
fourier_elim([x2 - 1 < 0], [x])/x2 - 1 < 0
```

```
fourier_elim([x2 - 10], [x])/x-1 < 0
```

```
fourier_elim([x6], [x])
```

```
fourier_elim([x < 1, x 1], [x])/tidakmemilikipenyelesaian
```

```
fourier_elim([min f < x, x < inf], [x])/solusinyaR
```

```
fourier_elim([x3 - 1 0], [x])
```

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

```
fourier_elim([y - x < 5, x - y < 7, 10 < y], [x, y])/sistempertidaksamaan
```

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

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

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

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

```
fourier_elim([max(x,y)gt; 6, x8, mabs(y - 1)gt; 12], [x, y])
```

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

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=[2,6;1,5]
```

```
2 6 1 5
```

Produk matriks dilambangkan dengan titik.

```
b=[2;4]
```

```
2 4
```

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

```
[2, 4]
```



```

inv(A) //inverse A
1.25 -1.5 -0.25 0.5
A.b //perkalian matriks
28 22
A.inv(A)
1 0 0 1

```

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

```

A.A
10 42 7 31
A2//perpangkatanelemen2A
4 36 1 25
A.A.A
62 270 45 197
power(A,3) //perpangkatan matriks
62 270 45 197
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)
1 3 0.25 1.25
A
b // hasilkali invers A dan b,  $A^{-1}b$ 
-3.5 1.5
inv(A).b
-3.5 1.5
A
A //  $A^{-1}A$ 
1 0 0 1
inv(A).A
1 0 0 1
AA //perkalin elemen-elemen matriks seletak
4 36 1 25

```

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

```

b2//perpangkatanelemen – elemenmatriks/vektor
4 16

```

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

```

2A
4 12 2 10

```

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

```

[1,2]A
2 12 1 10

```

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

```

A[2,3]

```

```
4 18 2 15
```

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
```

```
Adup([1,2],2)
```

```
2 12 1 10
```

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)' // hasilkali 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)' // hasilkali vektor baris dan vektor kolom
```

```
55
```

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

```
55
```

Bahkan operator seperti `lt`; 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, 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, 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, 568,
576, 620, 667, 711, 719, 763, 810, 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(t2 + 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
```

Dimungkinkan 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 elemen2 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,ncols(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, 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(10:22,3)
```

```
[10, 11, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22]
```

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 1 0 0 0 0 1 0 0 0 0 1 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(bid(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
```

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

```
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, -0.192, -0.273, -0.336, -0.375, -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',
```

```
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 norma (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 \cdot

```
[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]; vv'
```

```
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)` mengemba-

likan 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
`vv, sum(vv), cumsum(vv)`

`[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.

`A4`

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 . Kita mendapatkan matriks, di mana setiap $a_{i,j} = t_j^i$, $1 \leq j \leq 101$, $1 \leq i \leq n$

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 "map" membuat vektor fungsi. Fungsinya sekarang akan bekerja untuk vektor bilangan.

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

```
Function f needs at least 2 arguments! Use: f (x, a) Error in: f([1:5]) ...
```

Sub-Matriks dan Matriks-Elemen

Untuk mengakses elemen matriks, gunakan notasi braket.

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

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

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)
[8, 10, 4, 5, 1, 3, 9, 6, 7, 2]
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=["s","e","v","e","n","t","e","n"]
```

```
s e v e n t e n
```

```
ss,ind=sort(s); ss
```

```
e e e n n s t v
```

Seperti yang Anda lihat, posisi entri ganda agak acak.

```
ind
```

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

Fungsi unik mengembalikan daftar elemen unik vektor yang diurutkan.

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

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

Ini bekerja untuk vektor string juga.

```
unique(s)
```

```
e n s t v
```

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` 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))
```

```
7.80486786311485e-14
```

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 `amp;:=`, dan kemudian menggunakannya 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
```

```
det(A),factor(
```

```
invert(A)witha = 0
```

```
A = [1,a;b,2]; A
```

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

```
det(A - xident(2)),solve(
```

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

```
eigenvalues([a, 1; 1, a])
```

Untuk mengekstrak vektor eigen tertentu perlu pengindeksan yang cermat.

```
eigenvectors([a, 1; 1, a]),
```

```
[1, - 1]
```

Matriks simbolik dapat dievaluasi dalam Euler secara numerik seperti ekspresi simbolik lainnya.

```
A(a=4,b=5)
```

```
1 4 5 2
```

Dalam ekspresi simbolik, gunakan dengan.

```
Awith[a = 4, b = 5]
```

Akses ke baris matriks simbolik bekerja seperti halnya dengan matriks numerik.

```
A[1]
```

Ekspresi simbolis dapat berisi tugas. Dan itu mengubah matriks A.

```
A[1,1]:=t+1; A
```

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
```

```
B := [1,2;3,4]; B,invert(B)
```

Hasilnya dapat dievaluasi secara numerik dalam Euler. Untuk informasi lebih lanjut tentang Maxima, lihat pengantar Maxima.

```
invert(B)()
```

```
-2 1 1.5 -0.5
```

Euler juga memiliki fungsi `xinv()` yang kuat, yang membuat upaya lebih besar dan mendapatkan hasil yang lebih tepat.

Perhatikan, bahwa dengan `amp;:=` matriks B telah didefinisikan sebagai simbolik dalam ekspresi simbolik dan sebagai numerik dalam ekspresi numerik. Jadi kita bisa menggunakannya di sini.

```
longest B.xinv(B)
```

```
1 0 0 1
```

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)
```

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 `"amp;:="`.

```
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
```

Untuk menghindarinya, ada fungsi `"mxmset(variable)"`.

```
mxmset(A); A
```

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))
```

Ketepatan angka floating point besar dapat diubah.

```
fpprec:=100; bfloat(pi)
```

```
3.14159265358979323846264338327950288419716939937510582097494 45923078164062862089986280348253
```

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)
```

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% tambahkan satu tarif sederhana dan hitung hasilnya.

```
K1.03
```

```
5150
```

Euler akan memahami sintaks berikut juga.

```
K+K3
```

```
5150
```

Tetapi lebih mudah menggunakan faktornya

```
q=1+3
```

```
1.03 5150
```

Selama 10 tahun, kita cukup mengalikan faktornya dan mendapatkan nilai akhir dengan suku bunga majemuk.

```
Kq10
```

```
6719.58189672
```

Untuk tujuan kita, kita dapat mengatur format menjadi 2 digit setelah titik desimal.

```
format(12,2); Kq10
```

```
6719.58
```

Mari kita cetak yang dibulatkan menjadi 2 digit dalam kalimat lengkap.

```
"Starting from " + K + "you get" + round(Kq10,2) + "."
```

```
Starting from 5000you get6719.58.
```

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

```
Kq(0 : 10)
```

```
Real 1 x 11 matrix
```

```
5000.00 5150.00 5304.50 5463.64 ...
```

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, 1.3439]

adalah vektor faktor q^0 sampai q^{10} . *Indikalikandengan K, dankamimendapatkanvektornilai.*

VK=Kq(0 : 10);

Tentu saja, cara realistis untuk menghitung suku bunga ini adalah dengan membulatkan ke sen terdekat setelah setiap tahun. Mari kita tambahkan fungsi untuk ini.

function oneyear (K) := round(Kq,2)

Mari kita bandingkan dua hasil, dengan dan tanpa pembulatan.

longest oneyear(1234.57), longest 1234.57q

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 ...

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) := Kq+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 ...
```

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 ...
```

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 ...
```

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

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 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 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) = Kq+R; op(K)
```

Kita sekarang dapat mengulangi ini.

```
op(op(op(op(K))))),expand(
```

Kami melihat sebuah pola. Setelah n periode yang kita miliki

Rumusnya adalah rumus untuk jumlah geometri, yang diketahui Maxima.

```
sum(qk, k, 0, n - 1);
```

Ini agak rumit. Jumlahnya dievaluasi dengan bendera "simpsum" untuk mengurangnya menjadi hasil bagi.

Mari kita membuat fungsi untuk ini.

```
function fs(K,R,P,n) = (1+P/100)nK + ((1+P/100)n - 1)/(P/100)R; fs(K,R,P,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
```

Biasanya rumus ini diberikan dalam bentuk

```
equ = (equ with P=100i); equ
```

Kita dapat memecahkan tingkat R secara simbolis.

```
solve(equ, R)
```

Seperti yang Anda lihat dari rumus, fungsi ini mengembalikan kesalahan titik mengambang untuk i=0. Euler tetap merencanakannya.

Tentu saja, kami memiliki batas berikut.

$\text{limit}(R(5000, 0, x, 10), x, 0)$

Jelas, tanpa bunga kita harus membayar kembali 10 tarif 500.

Persamaan juga dapat diselesaikan untuk n . Kelihatannya lebih bagus, jika kita menerapkan beberapa penyederhanaan untuk itu.

$\text{fn} = \text{solve}(\text{equ}, n) \text{ --- ratsimp; } fn$

—

Penyelesaian Soal Aljabar

1. Sederhanakan bentuk eksponen berikut

Penyelesaian :

$$((26a^5b^{15}c^7)/(22a^9b^4c^5))^{(-5)}$$

2. Sederhanakan bentuk eksponen berikut

Penyelesaian :

$$((26p^{15}q^5r^22)/(222p^9q^4r^{-15}))^{(-4)}$$

3. Hitunglah

Penyelesaian :

$$(9(8-3)^2 - 43 + 28)/(3^1 + 19^0)$$

229 — 4

4. Hitunglah

Penyelesaian :

$$((4(8-6)^2 + 4)(3 - 28))/(2^2(2^3 + 5))$$

- 5

5. Hitunglah

Penyelesaian :

$$(3x^2 - 2x - x^3 + 2) - (5x^2 - 8x - x^3 + 4)$$

6. Menyelesaikan pemfaktoran

Penyelesaian :

$$\text{factor}(t^2 + 8t + 15)$$

7. Menyelesaikan operasi yang di tunjukkan

Penyelesaian :

$$(2x + 3y + z - 7) + (4x - 2y - z + 8) + (-3x + y - 2z - 4)$$

8. Hitunglah :

Penyelesaian :

$$\text{factor}(x^3 - 4x^2 + 5x - 20)$$

9. Faktorkan

Penyelesaian :

$$\text{factor}(22z^9 - 4z)$$

10. Faktorkan :

Penyelesaian :

$$\text{factor}(30a^{12}b - 29ab^2)$$

11. Faktorkan

Penyelesaian :

$$\text{factor}(9ab^4 - 9az^4)$$

12. Faktorkan :

Penyelesaian :

$$\text{factor}(16a^7b + 54ab^7)$$

13. Hitunglah :

Penyelesaian :

$factor(m^6 + 8m^3 - 20)$

14. Asumsikan bahwa semua eksponen adalah bilangan asli. Kalikan

Penyelesaian :

$expand((a^n + b^n)(a^n - b^n))$

15. Asumsikan bahwa semua eksponen adalah bilangan asli. Kalikan

Penyelesaian :

$expand((x^{3m} - t^{5n})^2)$

16. Asumsikan bahwa semua eksponen adalah bilangan asli. Kalikan

Penyelesaian :

$expand((a + b + c)^2)$

17. Menyelesaikan operasi yang ditunjukkan

Penyelesaian :

$(2x^2 + 12xy - 11) + (6x^2 - 2x + 4) + (-x^2 - y - 2)$

18. Asumsikan bahwa semua variabel dalam eksponen merepresentasikan bilangan asli. Faktorkan

Penyelesaiann :

$factor(25y^{2m} - (x^{2n} - 2x^n + 1))$

19. Asumsikan bahwa semua variabel dalam eksponen merepresentasikan bilangan asli. Faktorkan

Penyelesaian :

$factor(bdy^2 + ady + bcy + ac)$

20. Asumsikan bahwa semua variabel dalam eksponen merepresentasikan bilangan asli. Faktorkan

Penyelesaian :

$factor(bdy^2 + ady + bcy + ac)$

3 Plot 2D

PLOT2D Emlys Jenny Ester Sirait (23030630081)

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).

Basic Plots

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

Macam-macam basic plot :

`clg;` // untuk membersihkan layar

`window(0,0,1024,1024);` // gunakan semua window

`setplot(0,1,0,1);` // koordinat set plot

```

hold on; // untuk memulai overwrite mode
n=100; X=random(n,2); Y=random(n,2); // untuk membuat koordinat
acak
colors=rgb(random(n),random(n),random(n)); // get random colors
loop 1 to n; color(colors[]); plot(X[],Y[]); end; // plot
hold off; // mengakhiri overwrite mode
insimg; // memasukkan ke notebook notebook
reset;

```

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

Untuk menghapus semua yang kami lakukan, maka 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.

Contoh lain, kita menggambar plot sebagai sisipan di plot lain. Hal ini dilakukan dengan mendefinisikan jendela plot yang lebih kecil. Perhatikan bahwa jendela ini tidak memberikan ruang untuk label sumbu di luar jendela plot. Maka Kita harus menambahkan beberapa margin sesuai kebutuhan.

```

plot2d("x3 - 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("x4 - x", grid = 6) :
hold off;
window(ow);

```

Plot dengan banyak gambar dicapai dengan cara yang sama. Ada fungsi utility figure() untuk ini.

Plot Aspect

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

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

```

aspect(1); // rasio panjang dan lebar 2:1
plot2d(["sin(x)", "cos(x)"], 0, 2pi);
aspect(2); // rasio panjang dan lebar 2:1
plot2d(["sin(x)", "cos(x)"], 0, 2pi);
aspect(3); // 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 membuat plot 2D

- * ekspresi
- * fungsi, variabel, atau kurva berparameter,
- * 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" (misalnya " $4x^2$ ") atau nama suatu fungsi (misalnya "f") menghasilkan grafik f.

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

Catatan: Jika Anda mengakhiri baris perintah dengan titik dua ":", plot akan dimasukkan ke dalam jendela teks.

```
plot2d("x^2") :
aspect(1.5); plot2d("x^3 - x") :
a:=5.6; plot2d("exp(-ax^2)/a"); insimg(30);
```

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

Rentang plot diatur dengan parameter yang ditetapkan sebagai berikut

- * a,b: rentang x (default -2,2)
 - * c,d: rentang y (default: skala dengan nilai)
 - * r: alternatifnya radius di sekitar pusat plot
 - * cx,cy: koordinat pusat plot (default 0,0)
- ```
plot2d("x^3 - x", -1, 2) :
plot2d("sin(x)", -2pi, 2pi):
plot2d("cos(x)", "sin(3x)", xmin=0, xmax=2pi):
```

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

Dalam opsi, plot dapat diatur agar muncul di jendela terpisah yang dapat diubah ukurannya.

Untuk membagi jendela menjadi beberapa plot, gunakan perintah figure().

Dalam contoh, kita memplot  $x^1$  hingga  $x^4$  menjadi 4 bagian jendela.gambar(0) mengatur ulang jendela default.

```
reset;
```

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

Di plot2d(), ada gaya alternatif yang tersedia dengan grid=x. Untuk gambaran umum, kami menampilkan berbagai gaya kisi dalam satu gambar (lihat

di bawah untuk perintah figure()). Gaya grid=0 tidak disertakan. Ini tidak menunjukkan kisi dan bingkai.

```
figure(3,3); ... for k=1:9; figure(k); plot2d("x3-x", -2, 1, grid = k); end; ...figure(0) :
```

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

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

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

```
aspect(1.5); plot2d("sin(x)", 0, 2pi, -1.2, 1.2, grid=3, xl="x", yl="sin(x)");
plot2d("sin(x)+cos(2x)", 0, 4pi):
```

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

```
plot2d("sign(x)exp(-x2)", -1, 1, < adaptive, n = 10000) :
```

```
plot2d("xx", r = 1.2, cx = 1, cy = 1) :
```

Parameter square=true (atau gt;square) memilih rentang y secara otomatis sehingga hasilnya adalah jendela plot persegi. Perhatikan bahwa secara default, Euler menggunakan spasi persegi di dalam jendela plot.

```
plot2d("x3 - x", square) :
```

```
plot2d("integrate("sin(x)exp(-x2)", 0, x)", 0, 2) :
```

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

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

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

```
a:=5.6; expr = exp(-ax2)/a;
```

```
plot2d(expr, -2, 2):
```

```
plot2d(expr, r=1, thickness=2):
```

```
plot2d(diff(expr, x), add, style="-", color=red):
```

```
plot2d(diff(expr, x, 2), a=-2, b=2, c=-2, d=1):
```

```
plot2d(diff(expr, x), a=-2, b=2, square):
```

```
plot2d("x2", 0, 1, steps = 1, color = red, n = 10) :
```

```
plot2d("x2", 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 di file "plot.e", yang dimuat di awal program.

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

```
function f(x,a) := x2/a + ax2 - 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) := x3 - x; ... plot2d("f", r = 1) :
Berikut ini ringkasan fungsi yang diterima
* ekspresi atau ekspresi simbolik di x
* fungsi atau fungsi simbolik dengan nama "f"
* fungsi simbolik hanya dengan nama f
Fungsi plot2d() juga menerima fungsi simbolik. Untuk fungsi simbolik, na-
manya saja yang berfungsi.
function f(x) = diff(xx, x)
x x (log(x) + 1)
plot2d(f,0,2):
Tentu saja, untuk ekspresi atau ekspresi simbolik, nama variabel sudah
cukup untuk memplotnya.
expr = sin(x)exp(-x)
- x E sin(x)
plot2d(expr,0,3pi):
function f(x) = xx;
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: Defaultnya 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
muda, kuning
* rgb(merah,hijau,biru): parameternya real di [0,1].
plot2d("exp(-x2)", r = 2, color = red, thickness = 3, style = " - - ") :
Berikut adalah tampilan warna EMT yang telah ditentukan sebelumnya.
aspect(2); columnsplot(ones(1,16),lab=0:15,grid=0,color=0:15):
Tapi Anda bisa menggunakan warna apa saja.
columnsplot(ones(1,16),grid=0,color=rgb(0,0,linspace(0,1,15))):
Menggambar Beberapa Kurva pada bidang koordinat yang sama
Memvisualisaikan data yang memiliki lebih dari satu fungsi ke dalam satu
jenis jendela/gambar yang sama, dapat dilakukan dengan beberapa cara. Salah
satu caranya adalah dengan menggunakan gt;add. gt;add ini digunakan un-
tuk memanggil fungsi-fungsi tadi secara bersamaan. Kita telah menggunakan
metode ini diatas, pada contoh contoh sebelumnya.
aspect(); plot2d("cos(x)",r=2,grid=6); plot2d("x",style=".", add):
* Baris perintah diatas digunakan untuk membuat grafik dari fungsi * "cos(x)".
* aspect() digunakan untuk mengatur rasio dari grafik yang menentukan *
proporsi antara sumbu x dan sumbu y.
* r=2, artinya radius di sekitar pusat plotnya adalah 2

```

\* grid=6, digunakan untuk mengatur jumlah garis grid atau jenis grid \* yang digunakan pada grafik.

Fungsi kedua yang akan kita plot adalah fungsi  $y=x$

- style=".", artinya kita ingin membuat grafik dengan menggunakan titik titik untuk menandai nilai-nilai pada grafik.

```
aspect(2.5); 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):
```

\* Pada perintah ini kita menggunakan aspect(1.5). Fungsi yang ingin \* kita plot adalah fungsi  $\sin(x)$  dengan nilai  $x=0$  sampai  $x=2\pi$ . Fungsi \* kedua yang akan kita tambahkan ke plot  $\sin(x)$  adalah fungsi  $\cos(x)$  \* dengan style garisnya berupa garis putus putus berwarna biru.

Salah satu kegunaan gt;add adalah untuk menambahkan titik pada kurva.

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

\* Pada perintah ini kita akan memplot grafik dari fungsi  $\sin(x)$  dengan \* interval  $x$  dari 0 sampai  $\pi$ . Lalu kita ingin menambahkan suatu titik \* pada grafik sinus  $x$  tadi, dengan cara menambahkan perintah lain yaitu \* plot2d(2, sin(2), gt;point, gt;add);. 2 dalam perintah tersebut artinya \* nilai  $x$  yang kita pilih adalah 2. Dengan menambahkan gt;point pada garis \* perintah, kita sudah dapat menambahkan satu titik pada grafik sinus \* tadi.

Pada contoh di bawah ini, kita akan memplot suatu fungsi dan menambahkan suatu titik dengan label nama ("cl" atau center left), dan menyimpan hasilnya di notebook. Kita juga akan menambahkan label atau judul untuk plot fungsinya.

```
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):
```

\* Fungsi yang kita plot di atas adalah fungsi  $\cos(x)$  dan  $y=x$ , dengan \* rentang sumbu  $x$  dan sumbu  $y$  1.1, artinya sumbu  $x$  dan  $y$  akan diperluas \* sedikit lebih dari -1.1 hingga 1.1, posisi pusat grafik pada koordinat \* adalah 0.5 relatif terhadap rentang grafik, kurva  $\cos(x)$  berwarna \* hitam dengan style garis solid dan kurva  $y=x$  berwarna biru dengan \* style titik-titik. Tampilan grid yang digunakan dalam plot ini adalah \* grid 1.

\* ( $x_0=\text{solve}(\cos(x)-x,1)$ ) merupakan perintah yang digunakan untuk \* menyelesaikan persamaan  $\cos(x)-x=0$ , dan hasilnya akan disimpan dalam \* variabel  $x_0$ . Perintah plot yang kedua digunakan untuk menambahkan \* titik pada plot yang sudah dibuat sebelumnya, dengan nilai  $x=y=x_0$ . \* Selain itu, plot kedua ini ditambahkan label nama "Intersection Demo".

\* (label("cos(x)=x", $x_0,x_0$ ,pos="Cl", offset=20) merupakan perintah \* untuk menambahkan label pada titik potong anatar grafik  $\cos(x)$  dan  $x$ . \* Label nama ini diposisikan di center left, atau sebelah kiri pusat \* (titik) dan ada offset=20 dari titik potong, supaya titik tidak \* tertutup oleh label teks.

Dalam contoh berikut ini, kita akan memplot fungsi  $\text{sinc}(x)=\sin(x)/x$  dan ekspansi Taylor ke-8 dan ke-16. Untuk mencari ekspansi Taylor ini, kita akan menggunakan Maxima melalui ekspresi simbolik. Dalam perintah berikut, dilakukan pemanggilan plot2d() sebanyak 3 kali dan dilakukan dalam perintah

multi baris. Perintah plot kedua dan ketiga memiliki set flag `gt;add`, yang membuat plot menggunakan interval sebelumnya pada pemanggilan plot pertama.

Kita menambahkan kotak label yang menjelaskan fungsi-sungsi tersebut.

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

\* Pemanggilan plot pertama digunakan untuk menggambar fungsi  $\text{sinc}(x)$  pada interval 0 sampai  $4\pi$ . Warna dari grafik ini adalah hijau, dengan \* ketebalan 2.

\* Pemanggilan plot kedua digunakan untuk menggambar grafik dari \* polinomial Taylor, dari derajat 0 sampai dengan derajat 8. Warna dari \* grafik ini adalah biru, dan style garis putus-putus.

\* Pemanggilan plot ketiga digunakan untuk menggambar grafik dari \* polinomial Taylor, dari derajat 0 sampai dengan derajat 16. Warna dari \* grafik ini adalah biru, dan style garis dengan titik-titik putus.

\* Perintah keempat digunakan untuk menambahkan kotak label untuk \* menjelaskan grafik-grafik yang digambar. Fungsi  $\text{sinc}(x)$  diberikan \* warna hitam dengan style garis solid, fungsi Taylor sampai derajat \* ke-8 diberikan warna biru dengan style garis putus-putus, dan fungsi \* Taylor sampai derajat ke-16 diberikannya warna merah dengan style garis \* garis dengan titik-titik putus.

Pada contoh berikut, kami menghasilkan Polinomial Bernstein.

```
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;
* Perintah (plot2d("(1-x)^10",0,1))digunakan untuk menggambar * grafik dari fungsi (1-x)^10 pada rentang 0 hingga 1. Grafik ini * menggambar bentuk polinomial dari fungsi tersebut.
* Perintah kedua digunakan untuk mengulang perintah dari i=1 hingga * i=10. Dalam setiap iterasi loop, perintah ini menggambar grafik dari * suku polinomial dalam ekspansi binomial dari (1-x)^10. Fungsi yang * diplot adalah bin(10,i).x^i.(1-x)^(10-i) dimana bin(10,i) adalah * koefisien binomial.
* Perintah insimg digunakan untuk menyisipkan atau menampilkan gambar
* dalam hasil output.
```

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

Kita membuat sebuah matriks nilai dengan satu Polinomial Bernstein di setiap baris. Untuk ini, kita cukup menggunakan vektor kolom  $i$ . Baca kembali pengantar tentang bahasa matriks untuk mempelajari lebih lanjut.

```
x=linspace(0,1,500);
n=10; k=(0:n)'; // n is row vector, k is column vector
y=bin(n,k)x^k(1-x)^(n-k); //y is a matrix then
plot2d(x,y):
* Perintah linspace digunakan untuk membuat vektor x yang berisi 500 * titik yang terdistribusi secara merata (linier) dari 0 hingga 1.
* Perintah kedua digunakan untuk membuat vektor.
```



\* Perintah ketiga digunakan untuk menghitung nilai-nilai untuk y \* berdasarkan ekspansi binomial. Hasil dari operasi pada baris perintah \* ini adalah sebuah matriks.

\* plot2d(x,y) digunakan untuk menggambar grafik 2d dari matriks y \* terhadap vektor x.

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) :
```

Cara lain adalah dengan menggunakan vektor ekspresi (string). Lalu kalian dapat menggunakan larik warna, larik gaya, dan larik ketebalan dengan panjang yang sama.

```
plot2d(["sin(x)","cos(x)"],0,2pi,color=2:3):
```

Perintah diatas digunakan untuk memplot fungsi sin(x) dan cos(x) dalam rentang 0 hingga 2pi. Pada plot in, kedua fungsi diberikan warna yang berbeda.

```
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)*xi(1-x)(10-i),i,0,10)//makelist
10 9 8 2 7 3 [(1-x) , 10 (1-x) x, 45 (1-x) x , 120 (1-x) x , 6 4 5 5 4 6 3
7 210 (1-x) x , 252 (1-x) x , 210 (1-x) x , 120 (1-x) x , 2 8 9 10 45 (1-x)
x , 10 (1-x) x , x]
```

\* fungsi makelist() digunakan untuk membuat list dari elemen-elemen \* yang dihasilkan oleh ekspresi di dalamnya berdasarkan parameter yang \* diberikan.

\* (binomial(10,i)) merupakan fungsi yang digunakan untuk menghitung \* koefisien binomial, yaitu angka yang muncul dalam ekspansi dari \* (a+b)<sup>10</sup>.

\* (i,0,10) merupakan rentang iterasi untuk i dalam fungsi makelist. \* Artinya i akan berubah dari 0 hingga 10 dan untuk setiap nilai i \* dalam rentang ini, ekspresi binomial(10,i)\*x<sup>i</sup>(1-x)<sup>(10-i)</sup> akan \* dihitung dan dimasukkan ke dalam list.

```
mxm2str(v) // get a vector of strings from the symbolic vector
(1-x)1010*(1-x)9*x45*(1-x)8*x2120*(1-x)7*x3210*(1-x)6*x4252*
(1-x)5*x5210*(1-x)4*x6120*(1-x)3*x745*(1-x)2*x810*(1-x)*x9x10
* mxmstr() merupakan fungsi yang digunakan untuk mengkonversi atau *
mengubah format data dari bentuk simbolik menjadi bentuk string.
```

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

\* Perintah diatas digunakan untuk memvisualisasikan fungsi yang sudah \* dikonversi sebelumnya, yaitu fungsi mxm2str(v), dari rentang x=0 \* sampai x=1.

Alternatif lain adalah dengan menggunakan bahasa matriks di Euler.

Jika sebuah ekspresi menghasilkan sebuah 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 sebuah larik warna ditambahkan, maka akan digunakan untuk setiap baris plot.

```
n=(1:10)'; plot2d("xn",0,1,color = 1 : 10) :
```

Ekspresi dan fungsi satu baris dapat melihat variabel global.

Jika kalian tidak dapat menggunakan variabel global, kalian perlu menggunakan fungsi dengan parameter ekstra, dan memberikan parameter ini sebagai parameter semicolon atau titik koma.

Hati-hati dalam meletakkan semua parameter yang diberikan di akhir perintah plot2d. Pada contoh di bawah ini, kita memasukkan nilai  $a=5$  ke dalam 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") :
```

\* Plot diatas merupakan plot dari fungsi  $f(x,a) := 1/a \cdot \exp(-x^2/a)$ ,  $x$  merupakan variabel independen dan  $a$  mempengaruhi bentuk fungsi. Interval sumbu  $x$  dari  $x = -10$  hingga  $x = 10$ , ketebalan garis kurva 2, dengan  $a = 5$ .

Atau gunakan koleksi dengan nama fungsi dan semua parameter tambahan. List atau daftar khusus ini disebut koleksi panggilan, dan itu merupakan cara yang lebih banyak digunakan untuk mengoper argumen ke fungsi yang dengan sendirinya dioper sebagai argumen ke fungsi lain.

Pada contoh berikut, kita menggunakan perulangan untuk memplot beberapa fungsi.

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

\* Plot diatas merupakan plot dari fungsi  $f$  dengan parameter  $a=1$ , dan \* dari interval  $x=-10$  sampai  $x=10$ .

\* Perintah kedua merupakan perulangan untuk menggambar grafik fungsi  $f$  \* dengan parameter  $a=2:10$  (2 sampai 10). Grafik baru dengan nilai  $a=2:10$  \* ini kemudian ditambahkan ke grafik pertama saat nilai  $a=1$ .

Kita dapat mendapatkan hasil yang sama dengan grafik diatas menggunakan cara berikut, yaitu menggunakan bahasa matriks EMT. Masing-masing matriks  $f(x,a)$  adalah satu fungsi. Selain itu, kita dapat mengatur masing-masing baris dari matriks menggunakan warna yang berbeda. Klik dua kali pada fungsi getspectral() untuk penjelasan lebih lanjut.

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

\* Perintah  $x=-10:0.01:10$  digunakan untuk membuat vektor  $x$  yang berisi \* nilai dari -10 hingga 10 dengan interval 0.01.

\* Perintah  $a=(1:10)'$  digunakan untuk mendefinisikan vektor  $a$  yang \* berisi nilai dari 10 hingga 10.

\* Perintah  $plot2d(x,f(x,a),color=getspectral(a/10))$  digunakan untuk \* menggambar grafik dari fungsi  $f$  dengan  $x$  sebagai variabel independen \* dan  $a$  sebagai parameter. getspectral() digunakan untuk menetapkan \* warna yang berbeda pada setiap kurva.

Label Teks

Dekorasi sederhana pun bisa

\* judul dengan title= "..."

\* label x dan y dengan xl="...", yl="..."

\* label teks lain dengan label("...", x, y)

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

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

```

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

```

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 ini berlabuh di kanan atas secara default, tetapi `gt;kiri` berlabuh di kiri atas. Anda dapat memindahkannya ke tempat mana pun yang Anda suka. Posisi jangkar berada di pojok kanan atas kotak, dan angkanya merupakan pecahan dari ukuran jendela grafis. Lebarinya otomatis.

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

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

```

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

```

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

```

statplot(1:10,random(3,10),color=[red,blue, green]):

```

Fitur serupa adalah fungsi `textbox()`.

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

```

function f(x) = exp(-x)sin(2pix); ... plot2d("f(x)",0,2pi); ... textbox(latex("
textExample of a damped oscillation
f(x)=e^{-x}sin(2
pix)"),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"xrarr; xsup3; -x") :

```

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

```

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

```

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

Pada 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("
textFunction

```

```
Phi"), ... xl=latex("
phi"),yl=latex("
Phi(
phi"))); ... textbox(... latex("
Phi(
phi) = e^{-phi}
frac{sin(phi)phi}"), x = 0.8, y = 0.5); ... label(latex("
Phi", color = blue), 1, 0.4) :
```

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

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

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

Tentu saja fungsinya juga bisa digunakan.

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

```
if x<0 then return x^4 else return x^2 end if end function < /pre > Parameter" map" membantumenggunakan f
plot, itu tidak perlu. Tapi untuk menunjukkan vektorisasi itu
```

```
berguna, kita menambahkan beberapa poin penting ke plot di x=-1, x=0
```

dan x=1.

Pada plot berikut, kami juga memasukkan beberapa kode LaTeX. Kami menggunakannya untuk

dua label dan kotak teks. Tentu saja, Anda hanya bisa menggunakannya

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
```

$x^2x$

$le0$

```
end cases"), ... x = 0.7, y = 0.2) :
```

Interaksi Pengguna

Saat memplot suatu fungsi atau ekspresi, parameter gt;pengguna memungkinkan pengguna untuk memperbesar dan menggeser plot dengan tombol kursor atau mouse. Pengguna bisa

- \* 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 aslinya.

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

```
plot2d("x3 - ax", 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 aktivitas mouse atau keyboard. Ini melaporkan mouse ke bawah, mouse digerakkan 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. Misalnya, kita melakukan interpolasi pada 5 titik dengan polinomial. Fungsi tersebut harus diplot ke dalam area plot yang tetap.

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

```
 d=interp(xp,yp); plot2d("interpval(xp,d,x)" ;d,xp,r=2); plot2d(xp,yp,i,points,i,add);
if select;i<0 then plot2d(xp[select],yp[select],color=red,i,points,i,add); endif; ti-
tle("Drag one point, or press space or return!"); endfunction i/pre; Perhatikan
parameter titik koma di plot2d (d dan xp), yang diteruskan ke evaluasi fungsi
interp(). Tanpa ini, kita harus menulis fungsi plotinterp() terlebih dahulu, men-
gakses nilainya secara global.
```

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

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

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

Pertama kita membutuhkan fungsi plot.

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

Kemudian kita memerlukan nama untuk parameter, nilai awal dan matriks rentang `nx2`, opsional garis judul.

Ada penggeser interaktif, yang dapat menetapkan nilai oleh pengguna. Fungsi `dragvalues()` menyediakan ini.

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

```
Error in return result. plotf: useglobal; return plot2d("exp(a*x)*cos(2pi*b*x)",0,2pi;a,b):
... Try "trace errors" to inspect local variables after errors. dragvalues: f(vv,args());
```

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

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

```
 plot2d("cos(x)",0,2pi,i,square,grid=6); plot2d("taylor(cos(x),x,0,@n)",color=blue,i,add);
 textbox("Taylor polynomial of degree "+n,0.1,0.02,style="t",i,left); endfunction
i/pre; plotf(1):
```

Sekarang kita izinkan derajat `n` bervariasi dari 0 hingga 20 dalam 20 perhentian. 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 fungsinya. Pengguna dapat menggambar jendela plot, meninggalkan jejak titik.

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

#### Gaya Plot 2D

Secara default, EMT menghitung penanda kecil sumbu otomatis dan menambahkan label ke setiap penanda. Ini dapat diubah dengan parameter tampilan. Gaya default sumbu dan label dapat diubah. Selain itu, label dan judul dapat ditambahkan secara manual. Untuk menyetel ulang ke gaya default, gunakan `reset()`.

```
aspect();
figure(3,4); ... figure(1); plot2d("2x4-8x",grid=0);...//tidakadatampilan,bingkaidansumbu
figure(2); plot2d("x3-x",grid=1);...//terdapatsumbu-x-y
figure(3); plot2d("x3-x",grid=2);...//terdapatpenandakecilotomatis
figure(4); plot2d("x3-x",grid=3);...//sumbu-x-ydenganlabeldidalamnya
figure(5); plot2d("x3-x",grid=4);...//tidakadapenandahanyalabel
figure(6); plot2d("x3-x",grid=5);...//defaulttitidakadamargin
figure(7); plot2d("x3-x",grid=6);...//hanyasumbudanpenandakecil
figure(8); plot2d("x3-x",grid=7);...//hanyasumbudanpenandakecilpadasumbu
figure(9); plot2d("x3-x",grid=8);...//hanyasumbudanpenandakecilterperincipadasumbutertentu
figure(10); plot2d("x3-x",grid=9);...//defaultdenganpenanda-penandakecildidalamnya
figure(11); plot2d("x3-x",grid=10);...//tidakadapenandakecil,hanyasumbu
figure(0):
```

Parameter `lt`, frame mematikan frame, dan `framecolor=blue` mengatur frame menjadi warna biru.

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

```
aspect(1.5);
plot2d("9x5-12x",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=ex")); //titleabovetheplot
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("x3-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 bagian tengah bawah. Ini menetapkan posisi label relatif terhadap koordinat plot.

```
function f(x) = x3 - x
3 x - 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,-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("x3-
x",grid = 1); ... settittle("y = x3 - x",color = black); ... label("x",2,0,pos =
"bc",color = gray); ... label("y",0,6,pos = "cl",color = gray); ... reset() :
```

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

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

```
fullwindow; ... gridstyle(color=darkgray,textcolor=darkgray); ... plot2d(["2x", "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(["2x", "1", "2-x"], colors =
4 : 6, x = 0.8, y = 0.2); ... reset :
```

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

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

#### Memplot Data 2D

Jika x dan y adalah vektor data, maka data dalam vektor x dan y ini akan digunakan sebagai koordinat x dan y dari sebuah kurva. Dalam hal ini, a, b, c, dan d, atau radius r dapat ditentukan, jika nilai-nilai tersebut tidak ditentukan, plot window akan menyesuaikan secara otomatis dengan data. Sebagai alternatif, perintah `gt;square` dapat digunakan untuk mempertahankan rasio aspek persegi.

Memplot sebuah ekspresi hanyalah singkatan untuk plot data. Untuk data plot, kalian membutuhkan satu atau beberapa baris nilai x, dan satu atau beberapa baris nilai y. Dari rentang dan nilai x, fungsi `plot2d` akan menghitung data untuk diplot secara default dengan evaluasi adaptif dari fungsi tersebut. Untuk menambahkan titik pada gambar grafik, gunakan perintah `"gt;points"`, untuk garis dan titik gunakan perintah `"gt;addpoints"`.

Namun, kalian juga dapat 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) :
```

- \* x=-10:0.1:10, merupakan perintah untuk membuat sebuah vektor dengan nama x, yang berisi deretan nilai dari -10 hingga 10 dengan beda \* setiap nilainya adalah 0.1.

- \* y=exp(-x^2) \* x, merupakan perintah untuk membuat vektor dengan nama \* y, yang berisi deretan nilai berdasarkan x.

- \* plot2d(x,y), merupakan perintah untuk membuat grafik 2d dari data x \* dan y.

Data juga bisa diplot sebagai titik. Gunakan perintah points=true untuk membuatnya. Plot ini bekerja seperti poligon, namun hanya menggambar sudut-sudutnya saja.

```
* style="...": Select from "[]", "lt;gt;", "o", ".", "..", "+", "*", * "[]", "lt;gt;", "o", ".", "..", "", "—".
```

Untuk memplot kumpulan titik, gunakan perintah gt;points. Jika warnanya merupakan vektor warna, setiap titik akan mendapatkan warna yang berbeda. Untuk matriks koordinat dan vektor kolom, warna tersebut diaplikasikan pada baris setiap matriks. Parameter gt;addpoints menambahkan titik-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
```

Perintah diatas digunakan untuk membuat plot 2d dari xdata dan ydata, dengan rentang sumbu x dari 0.5 - 4.5 dan rentang sumbu y dari 2.5 - 3.5. Kumpulan titik-titik data tersebut akan ditampilkan dengan style titik.

```
plot2d(xdata,ydata, points, add,style="o"): // add points
```

```
p=polyfit(xdata,ydata,1); // get regression line
```

Perintah diatas digunakan untuk menghitung koefisien polinomial orde 1, dimana hasilnya adalah nilai p yang berisi dua koefisien dari garis regresi. Dua koefisien ini merupakan kemiringan dan intercept dari garis regresi. Garis regresi adalah garis lurus yang menggambarkan hubungan antara dua variabel, yaitu variabel dependen (y) dan variabel independen (x). Garis regresi digunakan untuk menggambarkan perilaku sekumpulan data.

```
plot2d("polyval(p,x)", add,color=red): // add plot of line
```

Perintah ini digunakan untuk menambahkan garis regresi yang telah dihitung sebelumnya ke dalam grafik yang sudah ada.

- perintah "polyval(p,x)" menghitung nilai prediksi berdasarkan koefisien polinomial p dan data x. Fungsi ini menggunakan nilai koefisien dari p untuk menghitung nilai y yang sesuai dengan regresi linier.

Menggambar Daerah Yang Dibatasi Kurva

Plot data sebenarnya berbentuk poligon. Kita juga dapat memplot kurva atau kurva terisi.

- \* fillex=benar mengisi plot.

- \* style="...": Pilih dari ":", " /", ";", "—".



\* fillcolor : Lihat di atas untuk mengetahui warna yang tersedia.

Warna isian ditentukan oleh argumen "fillcolor", dan pada lt;outline optional, mencegah menggambar batas untuk semua gaya kecuali gaya default.

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

Dalam contoh berikut kita memplot 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 septagon yang kita buat dengan 7 titik pada lingkaran satuan.

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

Berikut adalah himpunan nilai maksimal dari empat kondisi linier yang kurang dari atau sama dengan 3. Ini adalah  $A[k].vlt;=3$  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 pembuatan tabel fungsi dengan mudah.

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

Kami sekarang memiliki nilai vektor x dan y. plot2d() dapat memplot nilai-nilai ini

sebagai kurva yang menghubungkan titik-titik tersebut. Plotnya bisa diisi. Dalam hal ini

ini menghasilkan hasil yang bagus karena aturan belitan, yang digunakan untuk

isi.

```
plot2d(x,y,igrid,iframe, filled):
```

Vektor interval diplot terhadap nilai x sebagai wilayah terisi

antara nilai interval yang lebih rendah dan lebih tinggi.

Hal ini dapat berguna untuk memplot kesalahan perhitungan. Tapi itu bisa juga dapat digunakan untuk memplot kesalahan statistik.

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

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

```
t=-1:0.01:1; x= t-0.01,t+0.01 ; y=x3 - x;
```

```
plot2d(t,y):
```

Dimungkinkan untuk mengisi wilayah nilai untuk fungsi tertentu. Untuk

ini, level harus berupa matriks  $2 \times n$ . Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

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

Kita juga dapat mengisi rentang nilai seperti

```
plot2d("(x^2 + y^2)^2 - x^2 + y^2", r = 1.2, level = [-1;0], style = "/") :
plot2d("cos(x)", "sin(x)^3", xmin = 0, xmax = 2pi, filled, style = "/") :
```

Grafik Fungsi Parametrik

Nilai x tidak perlu diurutkan. (x,y) hanya menggambarkan sebuah kurva.

Jika x diurutkan, kurva tersebut merupakan grafik suatu fungsi.

Dalam contoh berikut, kita memplot spiral

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

```
t=linspace(0,1,1000); ... plot2d(tcos(2pit),tsin(2pit),r=1):
```

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

```
plot2d("xcos(2pix)", "xsin(2pix)", xmin=0, xmax=1, r=1):
t=linspace(0,1,1000); r=exp(-t); x=rcos(2pit); y=rsin(2pit);
plot2d(x,y,r=1):
```

Pada contoh berikutnya, kita memplot kurvanya

dengan

```
t=linspace(0,2pi,1000); r=1+sin(3t)/2; x=rcos(t); y=rsin(t); ... plot2d(x,y, filled, fillcolor=red, style="/", r,
```

Menggambar Grafik Bilangan Kompleks

Serangkaian bilangan kompleks juga dapat diplot. Kemudian titik-titik grid akan dihubungkan. Jika sejumlah garis kisi ditentukan (atau vektor garis kisi  $1 \times 2$ ) dalam argumen `cgrid`, hanya garis kisi tersebut yang terlihat.

Matriks bilangan kompleks secara otomatis akan diplot sebagai kisi-kisi pada bidang kompleks.

Pada contoh berikut, kita 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=rexp(Ia);... 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=rexp(Ia);
plot2d(exp(z),cgrid=[40,10]):
r=linspace(0,1,10); a=linspace(0,2pi,40)'; z=rexp(Ia);
plot2d(exp(z), points, add):
```

Vektor bilangan kompleks secara otomatis diplot sebagai kurva pada bidang kompleks dengan bagian nyata dan bagian imajiner.

Dalam contoh, kita memplot lingkaran satuan dengan

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

Plot Statistik

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

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

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

Penggunaan dua baris menunjukkan jalan dalam dua dimensi.

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

Itu juga dapat menampilkan string sebagai label.

```
months=["Jan","Feb","Mar","Apr","May","Jun", ... "Jul","Aug","Sep","Oct","Nov","Dec"];
values=[10,12,12,18,22,28,30,26,22,18,12,8];
columnplot(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 `gt;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()` mengatur 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`. Batangnya akan memanjang dari `x[i]` hingga `x[i+1]` dengan nilai `y[i]`. Jika `x` berukuran sama dengan `y`, maka `x` 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 bar` (`bar=1`) dan `histogram` (`histogram=1`) dapat diberikan secara eksplisit dalam `xv` dan `yv`, atau dapat dihitung dari distribusi empiris dalam `xv` dengan `gt;distribusi` (atau `distribusi=n`). Histogram nilai `xv` akan dihitung secara otomatis dengan `gt;histogram`. Jika `gt;even` ditentukan, nilai `xv` akan dihitung dalam interval bilangan bulat.

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

Untuk distribusi, terdapat 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 bilangan bulat.

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

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

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

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

Ada juga banyak plot khusus untuk statistik. Plot kotak menunjukkan kuartil distribusi ini dan banyak outlier. Menurut definisinya, outlier dalam plot kotak adalah data yang melebihi 1,5 kali rentang 50

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

Fungsi Implisit

Fungsi implisit adalah jenis fungsi dimana variabel dependen tidak dapat dipisahkan secara eksplisit dari variabel independen. Bentuk dari fungsi implisit sendiri adalah  $f(x,y)=0$ , dimana variabel, koefisien, dan konstanta sebagai persamaan di sisi kiri, dan disamakan dengan nol.

Contoh dari fungsi implisit misalnya persamaan lingkaran dan persamaan elips.

Plot implisit menunjukkan penyelesaian garis level  $f(x,y)=level$ , dengan "level" dapat berupa nilai tunggal atau vektor nilai. Jika `level = "auto"`, akan ada garis level `nc`, yang akan tersebar antara fungsi minimum dan maksimum secara merata.

Perintah `gt;hue` untuk membuat Warna yang lebih gelap atau lebih terang untuk menunjukkan nilai fungsi. Untuk fungsi implisit, `xv` harus berupa fungsi atau ekspresi parameter `x` dan `y`, atau alternatifnya, `xv` dapat berupa matriks nilai.

Euler dapat menandai garis level dari fungsi apa pun.

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

```
aspect();
```

```
plot2d("x2 + y2 - xy - x", level = 0, r = 1.5, contourcolor = red) :
```

```
expr := "2x2 + xy + 3y4 + 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=10): // nice
```

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

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

```
x=-2:0.05:1; y=x'; z=expr(x,y);
```

```
plot2d(z,level=0,a=-1,b=2,c=-2,d=1, hue):
```

```
plot2d("x3 - y2", hue, spectral, contour) :
```

Perintah `contour` adalah untuk menentukan bahwa grafik yang dihasilkan adalah grafik kontur. grafik kontur menunjukkan garis-garis yang menghubungkan

titik-titik dengan nilai fungsi yang sama, sehingga membuat visualisasi bentuk permukaan fungsi.

```
plot2d("x3 - y2", level = 0, contourwidth = 3, contourcolor = red, add) :
z=z+normal(size(z))2;
plot2d(z,level=0.5,a=-1,b=2,c=-2,d=1):
```

Perintah ini untuk menambahkan noise atau gangguan acak pada fungsi  $z$ . nilai yang dikalikan adalah standar deviasi atau besaran gangguan yang diberikan. semakin besar nilai yang dimasukkan maka semakin besar pula gangguan pada fungsi.

```
plot2d(expr,level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
plot2d("x2 + y3 + xy", level = 1, r = 4, n = 100) :
plot2d("x2 + 2y2 - xy", level = 0 : 0.1 : 10, n = 100, contourcolor =
white, hue) :
```

Dimungkinkan juga untuk mengisi set dengan rentang level.

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

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

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

```
plot2d("x4 + y4", level = [0; 1], r = 1.5, color = blue, style = "/") :
plot2d("x2 + y3 + xy", level = [0, 2, 4; 1, 3, 5], style = "/", r = 2, n = 100) :
plot2d("x2 + y3 + xy", level = -10 : 20, dl = 0.1, r = 5, style = " - ", n =
100) :
```

```
plot2d("sin(x)cos(y)", levels,r=2pi, hue,n=100):
```

Dimungkinkan juga untuk menandai suatu wilayah

Hal ini dilakukan dengan menambahkan level dengan dua baris.

```
plot2d("(x2+y2-1)3-x2y3", r = 6, ...style = "", color = red, < outline, ...level =
[-2;0], n = 100) :
```

```
plot2d("4x2-4.5", -0.75,0.75,thickness = 3,add);plot2d([0,0],[3,5],thickness =
3, add) :
```

```
plot2d("(x2+y2-1)3-x2y3", r = 6, ...style = "", color = red, < outline, ...level =
[-1;0], n = 100) :
```

```
plot2d([-3.5,-3.5],[-1.1,1.1],thickness=3, add); plot2d("4(x-4)2-1", 3.3,4.7,thickness =
3, add) :
```

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

```
plot2d("x3 - xy + x2y2", r = 6, level = 1, n = 100) :
```

```
function starplot1 (v, style="/", color=green, lab=none)... ipre class="udf" i
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 i/pre;
```

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

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

```
reset; starplot1(normal(1,10)+2,color=red,lab=1:10):
```

\* Starplot1 adalah perintah untuk menggambar plot berbentuk bintang. \* Ini adalah fungsi untuk menghasilkan vektor acak yang mengikuti \* distribusi normal dengan rata-rata 1 dan deviasi standar 10. Fungsi \* ini menghasilkan data yang terdistribusi normal dengan nilai-nilai \* acak. \* +5 adalah untuk menambahkan 5 ke setiap elemen dari vektor yang \* terdistribusi normal.

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

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

```
function logimpulseplot1 (x,y) ...
x0,y0=makeimpulse(x,log(y)/log(10)); plot2d(x0,y0,i,bar,grid=0); h=holding(1);
frame(); xgrid(ticks(x)); p=plot(); for i=-10 to 10; if ij=p[4] and iL=p[3] then
ygrid(i,yt="10"+i); endif; end; holding(h); endfunction < /pre > Marikitauidengannilaiyangterdistribusi
aspect(1.5); x=1:10; y=-log(random(size(x)))200; ... logimpulseplot1(x,y):
```

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

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

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

Tekan tombol apa saja untuk menghentikan animasi ini.

```
animliss(5,5); // lihat hasilnya, jika sudah puas, tekan ENTER
```

Plot Logaritmik

EMT menggunakan parameter "logplot" untuk skala logaritmik.

Plot logaritma dapat diplot menggunakan skala logaritma di y dengan logplot=1, atau menggunakan skala logaritma di x dan y dengan logplot=2, atau di x dengan logplot=3.

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

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

Ini juga berfungsi dengan plot data.



For lines use  
`"-", "--", "-.", ". ", ".-", "-.-", "-gt;"`  
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 gt;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 gt;user, you get  
the same user interaction as in plot2d. The range will be set  
before each call to your function.  
maps : map expressions (0 is faster), functions are always mapped.  
contourcolor : color of contour lines



contourwidth : width of contour lines  
clipping : toggles the clipping (default is true)  
title :

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

cgrid :

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

Overview

The function can plot

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

If a function or expression for `xv` is given, `plot2d()` will compute values in the given range using the function or expression. The expression must be an expression in the variable `x`. The range must be defined in the parameters `a` and `b` unless the default range `[-2,2]` should be used. The y-range will be computed automatically, unless `c` and `d` are specified, or a radius `r`, which yields the range `[-r,r]` for `x` and `y`. For plots of functions, `plot2d` will use an adaptive evaluation of the function by default. To speed up the plot for complicated functions, switch this off with `lt;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 `lt;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`.

## 4 Plot 3D

PLOT3D Emlys Jenny Ester Sirait (23030630081)  
Menggambar Plot 3D dengan EMT

Ini adalah pengenalan plot 3D di Euler. Kita memerlukan plot 3D untuk memvisualisasikan fungsi dua variabel.

Euler menggambar fungsi tersebut menggunakan algoritma pengurutan untuk menyembunyikan bagian-bagian di latar belakang. Secara umum, Euler menggunakan proyeksi pusat. Standarnya adalah dari kuadran x-y positif ke arah titik asal  $x=y=z=0$ , tetapi sudut= $0^\circ$  terlihat dari arah sumbu y. Sudut pandang dan ketinggian dapat diubah.

Euler dapat memetakan

- \* permukaan dengan bayangan dan garis-garis level atau rentang level,
- \* awan titik-titik,
- \* kurva parametrik,
- \* permukaan implisit.

Plot 3D dari sebuah fungsi menggunakan plot3d. Cara termudah adalah memplot ekspresi dalam x dan y. Parameter r mengatur rentang plot di sekitar (0,0).

```
aspect(1.5); plot3d("x^2 + sin(y)", -5, 5, 0, 6pi) :
plot3d("x^2 + xsin(y)", -5, 5, 0, 6pi) :
```

Silakan lakukan modifikasi agar gambar "talang bergelombang" tersebut tidak lurus melainkan melengkung/melingkar, baik melingkar secara mendatar maupun melingkar turun/naik (seperti papan peluncur pada kolam renang. Temukan rumusnya.

Fungsi dari dua Variabel

Untuk grafik sebuah fungsi, gunakan

- \* ekspresi sederhana dalam x dan y,
- \* nama fungsi dari dua variabel
- \* atau matriks data.

Standarnya adalah kisi-kisi kawat yang terisi dengan warna yang berbeda di kedua sisi. Perhatikan bahwa jumlah default interval grid adalah 10, namun plot menggunakan jumlah default 40x40 persegi panjang untuk membangun permukaan. Hal ini dapat diubah.

- \*  $n=40$ ,  $n=[40,40]$ : jumlah garis kisi 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 dapat dilakukan dengan parameter `gt;user`. Pengguna dapat menekan tombol berikut ini.

- \* kiri, kanan, atas, bawah: memutar sudut pandang
- \* +,-: memperbesar atau memperkecil
- \* a: menghasilkan anaglyph (lihat di bawah)
- \* l: beralih memutar sumber cahaya (lihat di bawah)
- \* spasi: mengatur ulang ke default
- \* kembali: mengakhiri interaksi

```
plot3d("exp(-x^2+y^2)", user, ...title = "Turnwiththevectorkeys(pressreturntofinish)") :
```

Rentang plot untuk fungsi dapat ditentukan dengan

- \* a, b: rentang x
- \* c, d: rentang y

\* r: bujur sangkar simetris di sekitar (0,0).

\* n: jumlah subinterval untuk plot.

Terdapat beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.

fscale: skala untuk nilai fungsi (standarnya adalah 1;fscale).

scale: angka atau vektor 1x2 untuk menskalakan ke arah x dan y.

frame: jenis bingkai (default 1).

plot3d("exp(-(x<sup>2</sup>+y<sup>2</sup>)/5)", r = 10, n = 80, fscale = 4, scale = 1.2, frame = 3, user) :

Tampilan dapat diubah dengan berbagai cara.

\* jarak: jarak pandang ke plot.

\* zoom: nilai zoom.

\* angle: sudut ke sumbu y negatif dalam radian.

\* height: ketinggian tampilan dalam radian.

Nilai default dapat diperiksa atau diubah dengan fungsi view(). Fungsi ini mengembalikan parameter dalam urutan di atas.

view

[5, 2.6, 2, 0.4]

Jarak yang lebih dekat membutuhkan zoom yang lebih sedikit. Efeknya lebih seperti lensa sudut lebar.

Dalam contoh berikut ini, sudut = 0 dan tinggi = 0 terlihat dari sumbu y negatif. Label sumbu untuk y disembunyikan dalam kasus ini.

plot3d("x<sup>2</sup> + y<sup>2</sup>", distance = 3, zoom = 1, angle = pi/2, height = 0) :

Plot terlihat selalu ke bagian tengah kubus plot. Anda dapat memindahkan bagian tengah dengan parameter center.

plot3d("x<sup>4</sup> + y<sup>2</sup>", 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 melakukan zoom, tergantung pada ukuran plot. Namun demikian, label mengacu ke ukuran yang sesungguhnya.

Jika Anda menonaktifkannya dengan scale=false, Anda harus berhati-hati agar plot tetap muat di dalam jendela plotting, dengan mengubah jarak tampilan atau zoom, dan memindahkan bagian tengahnya.

plot3d("5exp(-x<sup>2</sup> - y<sup>2</sup>)", r = 2, < fscale, < scale, distance = 13, height = 50, ... center = [0, 0, -2], frame = 3) :

Plot polar juga tersedia. Parameter polar=true menggambar plot polar. Fungsi harus tetap merupakan fungsi dari x dan y. Parameter "fscale" menskalakan fungsi dengan skala sendiri. Jika tidak, fungsi akan diskalakan agar sesuai dengan kubus.

plot3d("1/(x<sup>2</sup> + y<sup>2</sup> + 1)", r = 5, polar, ... fscale = 2, hue, n = 100, zoom = 4, contour, color = blue) :

function f(r) := exp(-r/2)cos(r); ... plot3d("f(x<sup>2</sup> + y<sup>2</sup>)", polar, scale = [1, 1, 0.4], r = pi, frame = 3, zoom = 4) :

Parameter rotate memutar fungsi dalam x di sekitar sumbu x.

\* rotate = 1: Menggunakan sumbu x

\* rotate=2: Menggunakan sumbu z

```
plot3d("x2 + 1", a = -1, b = 1, rotate = true, grid = 5) :
```

```
plot3d("x2 + 1", a = -1, b = 1, rotate = 2, grid = 5) :
```

```
plot3d("sqrt(25-x2)", a = 0, b = 5, rotate = 1) :
```

```
plot3d("xsin(x)", a=0, b=6pi, rotate=2):
```

Berikut ini adalah plot dengan tiga fungsi.

```
plot3d("x", "x2 + y2", "y", r = 2, zoom = 3.5, frame = 3) :
```

Plot Kontur

Untuk plot, Euler menambahkan garis kisi-kisi. Sebagai gantinya, dimungkinkan untuk menggunakan garis level dan rona satu warna atau rona berwarna spektral. Euler dapat menggambar ketinggian fungsi pada plot dengan bayangan. Pada semua plot 3D, Euler dapat menghasilkan anaglyph merah/cyan.

\* Rona: Mengaktifkan bayangan cahaya, bukan kabel.

\* gt;contour: Memplot garis kontur otomatis pada plot.

\* level=... (atau level): Vektor nilai untuk garis kontur.

Defaultnya 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 ini, kami menggunakan grid yang lebih halus untuk titik 100x100, skala fungsi dan plot, dan menggunakan sudut pandang yang berbeda.

```
plot3d("exp(-x2-y2)", r = 2, n = 100, level = "thin", ...contour, spectral, fscale = 1, scale = 1.1, angle = 45, height = 20) :
```

```
plot3d("exp(xy)", angle=100°, contour,color=green):
```

Bayangan default menggunakan warna abu-abu. Tetapi, kisaran warna spektral juga tersedia.

\* gt;spektral: Menggunakan skema spektral default

\* color =...: Menggunakan warna khusus atau skema spektral

Untuk plot berikut ini, kami menggunakan skema spektral default dan menambah jumlah titik untuk mendapatkan tampilan yang sangat mulus.

```
plot3d("x2 + y2", spectral, contour, n = 100) :
```

Alih-alih garis level otomatis, kita juga dapat menetapkan nilai garis level. Hal ini akan menghasilkan garis level yang tipis, alih-alih rentang level.

```
plot3d("x2 - y2", 0, 5, 0, 5, level = -1 : 0.1 : 1, color = redgreen) :
```

Pada plot berikut ini, 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-batas level sebagai kolom.

Selain itu, kami menghamparkan grid dengan 10 interval di setiap arah.

```
plot3d("x2 + y3", level = [-0.1, 0.9; 0, 1], ... spectral, angle = 30, grid = 10, contourcolor = gray) :
```

Pada contoh berikut, kami memplot himpunan, di mana

Kita menggunakan satu garis tipis untuk garis level.

```
plot3d("xy - yx", level = 0, a = 0, b = 6, c = 0, d = 6, contourcolor = red, n = 100) :
```

Dimungkinkan untuk menampilkan bidang kontur di bawah plot. Warna dan jarak ke plot dapat ditentukan.

```
plot3d("x2 + y4", cp, cpcolor = green, cpdelta = 0.2) :
```

Berikut ini beberapa gaya lainnya. Kami selalu mematikan bingkai, dan menggunakan berbagai skema warna untuk plot dan kisi-kisi.

```
figure(2,2); ... expr="y3-x2"; ...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, yang diberi nomor dari 1 hingga 9. Tetapi Anda juga dapat menggunakan color=value, di mana value

- \* spektral: untuk rentang dari biru ke merah
- \* putih: untuk rentang yang lebih redup
- \* kuningbiru, ungu-hijau, biru-kuning, hijau-merah
- \* biru-kuning, hijau-ungu, kuning-biru, merah-hijau

```
figure(3,3); ... for i=1:9; ... figure(i); plot3d("x2 + y2", spectral =
i, contour, cp, < frame, zoom = 4); ... end; ...figure(0) :
```

Sumber cahaya dapat diubah dengan l dan tombol kursor selama interaksi pengguna. Ini juga dapat ditetapkan dengan parameter.

- \* light: arah cahaya
- \* amb: cahaya sekitar antara 0 dan 1

Perhatikan, bahwa program ini tidak membuat perbedaan di antara sisi-sisi plot. Tidak ada bayangan. Untuk ini Anda akan membutuhkan Povray.

```
plot3d("-x2-y2", ...hue = true, light = [0, 1, 1], amb = 0, user = true, ...title =
"Presslandcursorkeys(returntoexit)") :
```

Parameter warna mengubah warna permukaan. Warna garis level juga dapat diubah.

```
plot3d("-x2-y2", 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 yang istimewa.

```
plot3d("x2/(x2 + y2 + 1)", color = 0, hue = true, grid = 10) :
```

Permukaannya juga bisa transparan.

```
plot3d("x2 + y2", transparent, grid = 10, wirecolor = red) :
```

Plot Implisit

Ada juga plot implisit dalam tiga dimensi. Euler menghasilkan potongan melalui objek. Fitur plot3d termasuk plot implisit. Plot-plot ini menunjukkan himpunan nol dari sebuah fungsi dalam tiga variabel.

Solusi dari

lateks:  $f(x,y,z) = 0$

dapat divisualisasikan dalam potongan yang sejajar dengan bidang x-y, bidang x-z, dan bidang y-z.

- \* implisit = 1: potong sejajar dengan bidang-y-z
- \* implicit = 2: memotong sejajar dengan bidang x-z
- \* implicit = 4: memotong sejajar dengan bidang x-y

Tambahkan nilai-nilai ini, jika Anda mau. Pada contoh, kami memplot

```
plot3d("x2 + y3 + zy - 1", r = 5, implicit = 3) :
c=1; d=1;
plot3d("((x2 + y2 - c2)2 + (z2 - 1)2)((y2 + z2 - c2)2 + (x2 - 1)2)((z2 + x2 -
c2)2 + (y2 - 1)2) - d", r = 2, < frame, implicit, user) :
```

```

plot3d("xy+x3y2 + xz3 - 9", r = 4, implicit = 3)
plot3d("x2 + y2 - z2 - 1", implicit = 3) :
plot3d("x2 + y2 + 4xz + z3", implicit, r = 2, zoom = 2.5) :
Memplot 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  $f_x(x,y)$ ,  $f_y(x,y)$ ,  $f_z(x,y)$ .

Karena x,y,z adalah matriks, kita mengasumsikan bahwa (t,s) berjalan melalui kotak persegi. Hasilnya, Anda dapat memplot gambar persegi panjang dalam ruang.

Anda dapat menggunakan bahasa matriks Euler untuk menghasilkan koordinat secara efektif.

Pada contoh berikut, kita menggunakan vektor nilai t dan vektor kolom nilai s untuk memparameterkan permukaan bola. Pada 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,iframe,grid=[10,20], ...
values=s,contourcolor=red,level=[90°-24°;90°-22°], ... scale=1.4,height=50°):

```

Berikut ini adalah contoh, yang merupakan grafik suatu fungsi.

```

t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,ts,grid=10):

```

Namun demikian, kita bisa membuat segala macam permukaan. Berikut ini adalah permukaan yang sama dengan suatu fungsi

```

plot3d(ts,t,s,angle=180°,grid=10):

```

Dengan lebih banyak upaya, kita bisa menghasilkan banyak permukaan.

Dalam contoh berikut ini, kami membuat tampilan berbayang dari bola yang terdistorsi. Koordinat yang biasa digunakan untuk bola adalah dengan

Kami mengubahnya dengan sebuah faktor

```

t=linspace(0,2pi,320); s=linspace(-pi/2,pi/2,160)'; ... d=1+0.2(cos(4 *
t)+cos(8s)); ... 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, awan titik juga dimungkinkan. Untuk memplot data titik dalam ruang, kita membutuhkan tiga vektor untuk koordinat titik.

Gaya-gayanya sama seperti pada plot2d dengan points=true;

```

n=500; ... plot3d(normal(1,n),normal(1,n),normal(1,n),points=true,style=".");

```

Anda juga dapat memplot kurva dalam bentuk 3D. Dalam hal ini, akan lebih mudah untuk menghitung titik-titik kurva. Untuk kurva pada bidang, 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 membuat plot dalam mode anaglyph. Untuk melihat plot semacam itu, Anda memerlukan kacamata merah/cyan.

```

plot3d("x2 + y3", anaglyph, contour, angle = 30) :

```

Sering kali, skema warna spektral digunakan untuk plot. Hal ini menekankan ketinggian fungsi.

```

plot3d("x2y3 - y", spectral, contour, zoom = 3.2) :

```

Euler juga dapat memplot permukaan yang diparameterkan, ketika parameternya adalah nilai  $x$ ,  $y$ , dan  $z$  dari sebuah gambar kisi-kisi persegi panjang di dalam ruang.

Untuk demo berikut ini, kami menyiapkan parameter  $u$  dan  $v$ , dan menghasilkan koordinat ruang dari parameter ini.

```
u=linspace(-1,1,10); v=linspace(0,2pi,50)'; ... X=(3+ucos(v/2))cos(v);
Y=(3+ucos(v/2))sin(v); Z=usin(v/2); ... plot3d(X,Y,Z, anaglyph,iframe, wire,scale=2.3):
```

Berikut ini contoh yang lebih rumit, yang tampak megah dengan kacamata merah/cyan.

```
u:=linspace(-pi,pi,160); v:=linspace(-pi,pi,400)'; ... x:=(4(1+.25sin(3 ×
v))+cos(u))cos(2v); ... y:=(4(1+.25sin(3v))+cos(u))sin(2v); ... z=sin(u)+2 ×
cos(3v); ... plot3d(x,y,z,frame=0,scale=1.5,hue=1,light=[1,0,-1],zoom=2.8, anaglyph):
```

Plot Statistik

Plot batang juga dapat digunakan. Untuk ini, kita harus menyediakan

- \*  $x$ : vektor baris dengan  $n+1$  elemen
- \*  $y$ : vektor kolom dengan  $n+1$  elemen
- \*  $z$ : matriks nilai berukuran  $n \times n$ .

$z$  dapat lebih besar, tetapi hanya nilai  $n \times n$  yang akan digunakan.

Pada contoh, pertama-tama kita menghitung nilainya. Kemudian kita sesuaikan  $x$  dan  $y$ , sehingga vektor berada di tengah-tengah 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 bagian atau lebih.

```
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 `gt;zscale`. Hal ini dapat dikombinasikan dengan faktor penskalaan individual yang diterapkan sebagai tambahan.

```
i=1:20; j=i'; ... plot3d(ij^2+100normal(20,20), zscale, scale = [1, 1, 1.5], angle =
-40, zoom = 1.8) :
```

```
SZ=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^2y^3", r = 1.3, ...style = "", color = red, < outline, ...level =
[-2;0], n = 100) :
```

```
ekspresi = (x^2 + y^2 - 1)^3 - x^2y^3;ekspresi
```

Kami ingin memutar kurva jantung di sekitar sumbu  $y$ . Berikut ini adalah ekspresi yang mendefinisikan jantung:

Selanjutnya kita atur

```
function fr(r,a) = ekspresi with [x=rcos(a),y=rsin(a)] — trigreduce; fr(r,a)
```

Hal ini memungkinkan untuk mendefinisikan fungsi numerik, yang menyelesaikan untuk  $r$ , jika  $a$  diberikan. Dengan fungsi tersebut kita dapat memplotkan jantung yang diputar sebagai permukaan parametrik.





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 defaultnya adalah `eulerhome()`, biasanya `c:`. Povray menghasilkan sebuah file PNG, yang dapat dimuat oleh Euler ke dalam notebook. Untuk membersihkan berkas-berkas ini, gunakan `povclear()`.

Fungsi `pov3d` memiliki semangat yang sama dengan `plot3d`. Fungsi 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-fungsi ini, mulai file Povray dengan `povstart()`. Kemudian gunakan `writeln(...)` untuk menulis objek ke file scene. Terakhir, akhiri file dengan `povend()`. Secara default, raytracer akan dimulai, dan PNG akan dimasukkan ke dalam buku catatan 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. Fungsi ini memiliki parameter untuk warna, transparansi, Phong Shading, dll.

Perhatikan bahwa Povray universe memiliki sistem koordinat lain. Antarmuka ini menerjemahkan semua koordinat ke sistem Povray. Jadi Anda dapat tetap berpikir dalam sistem koordinat Euler dengan  $z$  yang mengarah vertikal ke atas, dan sumbu  $x, y, z$  di tangan kanan.

Anda perlu memuat file povray.

```
load povray;
```

Pastikan direktori bin povray berada di dalam path. Jika tidak, edit variabel berikut sehingga berisi jalur ke povray yang dapat dieksekusi.

```
defaultpovray="C:
```

```
Program Files
```

```
POV-Ray
```

```
v3.7
```

```
bin
```

```
pvenGINE.exe"
```

```
C:\Files-Ray\3.7.exe
```

Untuk kesan pertama, kita plot sebuah fungsi sederhana. Perintah berikut ini menghasilkan file povray di direktori pengguna Anda, dan menjalankan Povray untuk melacak sinar pada 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 dijalankan. Anda dapat menekan cancel untuk menghentikan pertanyaan lebih lanjut. Anda mungkin harus menekan OK pada jendela Povray untuk mengetahui dialog awal Povray.

```
plot3d("x2 + y2", zoom = 2) :
```

```
pov3d("x2 + y2", zoom = 3);
```

Kita dapat membuat fungsi menjadi transparan dan menambahkan hasil akhir lainnya. Kita juga dapat menambahkan garis level ke plot fungsi.

```
pov3d("x2+y3", axiscolor = red, angle = -45, anaglyph, ...look = povlook(cyan, 0.2), level =
-1 : 0.5 : 1, zoom = 3.8);
```

Kadang-kadang perlu untuk mencegah penskalaan fungsi, dan menskalakan fungsi dengan tangan.

Kami memplot kumpulan titik pada bidang kompleks, di mana hasil kali jarak ke 1 dan -1 sama dengan 1.

```
pov3d("((x-1)2 + y2)((x+1)2 + y2)/40", r = 2, ... angle = -120, level =
1/40, dlevel = 0.005, light = [-1, 1, 1], height = 10, n = 50, ... < fscale, zoom =
3.8);
```

Merencanakan dengan Koordinat

Sebagai pengganti fungsi, kita dapat membuat plot dengan koordinat. Seperti pada plot3d, kita membutuhkan tiga matriks untuk mendefinisikan objek.

Pada contoh, kita memutar sebuah fungsi pada sumbu z.

```
function f(x) := x3-x+1; ...x = -1 : 0.01 : 1; t = linspace(0, 2pi, 50)'; ...Z =
x; X = cos(t)f(x); Y = sin(t)f(x); ...pov3d(X, Y, Z, angle = 40, look = povlook(red, 0.1), height =
50, axis = 0, zoom = 4, light = [10, 5, 15]);
```

Pada contoh berikut, kita 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 sesuai dengan kubus satuan.

```
r=linspace(0,1,80); phi=linspace(0,2pi,80)'; ... x=rcos(phi); y=rsin(phi);
z=exp(-5r)cos(8pir)/3; ... pov3d(x,y,z,zoom=6,axis=0,height=30°,add=povsphere([0.5,0,0.25],0.15,povlook(re
... w=500,h=300);
```

Dengan metode bayangan canggih Povray, hanya sedikit titik yang bisa menghasilkan permukaan yang sangat halus. Hanya pada batas-batas dan bayangan, trik ini bisa terlihat jelas.

Untuk itu, kita perlu menambahkan vektor normal di setiap titik matriks.

$$Z = x^2y^3$$

$$\begin{matrix} 2 & 3 & x & y \end{matrix}$$

Persamaan permukaannya adalah  $[x,y,Z]$ . Kami menghitung dua turunan terhadap x dan y dari persamaan ini dan mengambil hasil perkalian silang sebagai normal.

$$dx = \text{diff}([x,y,Z],x); dy = \text{diff}([x,y,Z],y);$$

Kami mendefinisikan normal sebagai hasil kali silang dari turunan ini, dan mendefinisikan fungsi koordinat.

$$N = \text{crossproduct}(dx,dy); NX = N[1]; NY = N[2]; NZ = N[3]; N,$$

$$\begin{matrix} 3 & 2 & 2 & [- & 2 & x & y & , & - & 3 & x & y & , & 1] \end{matrix}$$

Kami hanya menggunakan 25 poin.

$$x=-1:0.5:1; y=x';$$

$$\text{pov3d}(x,y,Z(x,y),\text{angle}=10^\circ, \dots \quad xv=NX(x,y),yv=NY(x,y),zv=NZ(x,y),jshadow);$$

Berikut ini adalah simpul Trefoil yang dibuat oleh A. Busser di Povray. Ada versi yang lebih baik dari ini dalam contoh.

Lihat: ContohTrefoil — Simpul Trefoil

Untuk tampilan yang bagus dengan tidak terlalu banyak titik, kita tambahkan vektor normal di sini. Kita menggunakan Maxima untuk menghitung

normal untuk kita. Pertama, tiga fungsi untuk koordinat sebagai ekspresi simbolik.

```
X = ((4+sin(3y))+cos(x))cos(2y); ... Y = ((4+sin(3y))+cos(x))sin(2y);
... Z = sin(x)+2cos(3y);
```

Kemudian dua vektor turunan terhadap x dan y.

```
dx = diff([X,Y,Z],x); dy = diff([X,Y,Z],y);
```

Sekarang yang normal, yang merupakan produk silang dari dua turunan.

```
dn = crossproduct(dx,dy);
```

Kami sekarang mengevaluasi semua ini secara numerik.

```
x:=linspace(-
```

Vektor normal adalah evaluasi dari ekspresi simbolik `dn[i]` untuk  $i=1,2,3$ .

Sintaks untuk ini adalah `amp;"ekspresi"(parameter)`. Ini adalah sebuah 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), anaglyph,axis=0,zoom=5,w=450,h=350, ...
;shadow,look=povlook(blue), ... xv="dn[1]"(x,y), yv="dn[2]"(x,y), zv="dn[3]"(x,y));
```

Kami juga dapat menghasilkan kisi-kisi dalam bentuk 3D.

```
povstart(zoom=4); ... x=-1:0.5:1; r=1-(x+1)^2/6; ... t = (0 : 30 : 360)'; y =
rcos(t); z = rsin(t); ... writeln(povgrid(x,y,z,d = 0.02,dballs = 0.05)); ... povend();
```

Dengan `povgrid()`, kurva dapat dibuat.

```
povstart(center=[0,0,1],zoom=3.6); ... t=linspace(0,2,1000); r=exp(-t); ...
x=cos(2pi10t)r; y=sin(2pi10t)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.

Kita memulai output dengan `povstart()`.

```
povstart(zoom=4);
```

Pertama, kita mendefinisikan tiga silinder, dan menyimpannya dalam bentuk 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(yellow));
... c3=povcylinder(-povz,povz,1,povlook(blue)); ... String berisi kode
```

Povray, yang tidak perlu kita pahami pada saat itu.

```
c2
```

```
cylinder lt;0,0,-1gt;; lt;0,0,1gt;; 1 texture pigment color rgb lt;0.941176,0.941176,0.392157gt;
finish ambient 0.2
```

Seperti yang Anda lihat, kami menambahkan tekstur ke objek dalam tiga warna berbeda.

Hal ini dilakukan dengan `povlook()`, yang mengembalikan sebuah string dengan kode Povray yang relevan. Kita dapat menggunakan warna default Euler, atau menentukan warna kita sendiri. Kita juga dapat menambahkan transparansi, atau mengubah cahaya sekitar.

```
povlook(rgb(0.1,0.2,0.3),0.1,0.5)
```

```
texture pigment color rgbf lt;0.101961,0.2,0.301961,0.1gt; finish ambient
0.5
```

Sekarang kita mendefinisikan objek perpotongan, dan menulis hasilnya ke file.

```
writeln(povintersection([c1,c2,c3]));
```

Perpotongan tiga silinder sulit untuk divisualisasikan, jika Anda belum pernah melihatnya.

```
povend;
```

Fungsi-fungsi berikut ini menghasilkan fraktal secara rekursif.

Fungsi pertama menunjukkan, bagaimana Euler menangani objek Povray sederhana. Fungsi `povbox()` mengembalikan sebuah 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) ... i/pre class="udf" i 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 i/pre i povstart(fade=10,ishadow);
fractal(-1,-1,-1,2,4);
povend();
```

Perbedaan memungkinkan pemotongan satu objek dari objek lainnya. Seperti persimpangan, ada bagian dari objek CSG Povray.

```
povstart(light=[5,-5,5],fade=10);
```

Untuk demonstrasi ini, kita mendefinisikan sebuah objek di Povray, alih-alih menggunakan string di Euler. Definisi akan langsung dituliskan ke file.

Koordinat kotak -1 berarti [-1,-1,-1].

```
povdefine("mycube",povbox(-1,1));
```

Kita dapat menggunakan objek ini dalam `povobject()`, yang mengembalikan sebuah string seperti biasa.

```
c1=povobject("mycube",povlook(red));
```

Kami menghasilkan kubus kedua, dan memutar serta menskalakannya sedikit.

```
c2=povobject("mycube",povlook(yellow),translate=[1,1,1], ... rotate=xrotate(10°)+yrotate(10°),
scale=1.2);
```

Kemudian kita ambil selisih dari kedua objek 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 fungsi-fungsi tersebut sedikit berbeda. Anda tidak dapat menggunakan output dari ekspresi Maxima atau Euler.

```
povstart(angle=25°,height=10°);
writeln(povsurface("pow(x,2)+pow(y,2)+pow(z,2)-1",povlook(blue),povbox(-
2,2,"")));
```

```

 povend();
 povstart(angle=70°,height=50°,zoom=4);
 Membuat permukaan implisit. Perhatikan sintaks yang berbeda dalam ek-
 spresi.
 writeln(povsurface("pow(x,2)y-pow(y,3)-pow(z,2)",povlook(green))); ... writeAxes();
... povend();

```

#### Objek Jaring

Pada contoh ini, kami menunjukkan cara membuat objek mesh, dan menggambar-nya dengan informasi tambahan.

Kami ingin memaksimalkan  $xy$  di bawah kondisi  $x+y = 1$  dan mendemonstrasikan sentuhan tangensial dari garis level.

```
povstart(angle=-10°,center=[0.5,0.5,0.5],zoom=7);
```

Kita tidak dapat menyimpan objek dalam sebuah string seperti sebelumnya, karena ukurannya terlalu besar. Jadi kita mendefinisikan objek dalam file Povray menggunakan declare. Fungsi povtriangle() melakukan hal ini secara otomatis. Fungsi ini dapat menerima vektor normal seperti halnya pov3d().

Berikut ini mendefinisikan objek mesh, dan menuliskannya langsung ke dalam file.

```

x=0:0.02:1; y=x'; z=xy; vx=-y; vy=-x; vz=1;
mesh=povtriangles(x,y,z,"",vx,vy,vz);

```

Sekarang kita tentukan dua cakram, yang akan berpotongan dengan permukaan.

```
cl=povdisc([0.5,0.5,0],[1,1,0],2); ... ll=povdisc([0,0,1/4],[0,0,1],2);
```

Tuliskan permukaan dikurangi kedua cakram.

```
writeln(povdifference(mesh,povunion([cl,ll],povlook(green)));
```

Tuliskan kedua perpotongan tersebut.

```
writeln(povintersection([mesh,cl],povlook(red))); ... writeln(povintersection([mesh,ll],povlook(gray)));
```

Tulislah satu titik secara maksimal.

```
writeln(povpoint([1/2,1/2,1/4],povlook(gray),size=2defaultpointsize));
```

Tambahkan sumbu dan selesaikan.

```
writeAxes(0,1,0,1,0,1,d=0.015); ... povend();
```

#### Anaglyph dalam Povray

Untuk menghasilkan anaglyph untuk kacamata merah/cyan, Povray harus dijalankan dua kali dari posisi kamera yang berbeda. Ini menghasilkan dua file Povray dan dua file PNG, yang dimuat dengan fungsi loadanaglyph().

Tentu saja, Anda membutuhkan kacamata merah/cyan untuk melihat contoh berikut dengan benar.

Fungsi pov3d() memiliki tombol sederhana untuk menghasilkan anaglyph.

```
pov3d("-exp(-x2-y2)/2",r=2,height=45,anaglyph,...center=[0,0,0.5],zoom=3.5);
```

Jika Anda membuat scene dengan objek, Anda harus menempatkan pembuatan scene ke dalam suatu fungsi, dan menjalankannya dua kali dengan nilai yang berbeda untuk parameter anaglyph.

```
function myscene ...
```

```

s=povsphere(povc,1); cl=povcylinder(-povz,povz,0.5); clx=povobject(cl,rotate=xrotate(90°));
cly=povobject(cl,rotate=yrotate(90°)); c=povbox([-1,-1,0],1); un=povunion([cl,clx,cly,c]);

```

```
obj=povdifference(s,un,povlook(red)); writeln(obj); writeAxes(); endfunction
i/prej Fungsi povanaglyph() melakukan semua ini. Parameter-parameternya
seperti pada povstart() dan povend() yang digabungkan.
```

```
povanaglyph("myscene",zoom=4.5);
Mendefinisikan Objek sendiri
```

Antarmuka povray Euler berisi banyak objek. Namun Anda tidak dibatasi pada objek-objek tersebut. Anda dapat membuat objek sendiri, yang menggabungkan objek-objek lain, atau objek yang benar-benar baru.

Kami mendemonstrasikan sebuah torus. Perintah Povray untuk ini adalah "torus". Jadi kita mengembalikan sebuah string dengan perintah ini dan parameternya. Perhatikan bahwa torus selalu berpusat pada titik asal.

```
function povdonat (r1,r2,look="") ...
return "torus "+r1+" "+r2+look+""; endfunction i/prej Inilah torus per-
tama kami.
```

```
t1=povdonat(0.8,0.2)
torus 0.8,0.2
```

Mari kita gunakan objek ini untuk membuat torus kedua, ditranslasikan dan diputar.

```
t2=povobject(t1,rotate=xrotate(90°),translate=[0.8,0,0])
object torus 0.8,0.2 rotate 90 *x translate lt;0.8,0,0gt;
```

Sekarang, kita tempatkan semua benda ini ke dalam suatu pemandangan.

Untuk tampilannya, kami menggunakan Phong Shading.

```
povstart(center=[0.4,0,0],angle=0°,zoom=3.8,aspect=1.5); ... writeln(povobject(t1,povlook(green,phong=
... writeln(povobject(t2,povlook(green,phong=1))); ... gt;povend();
```

memanggil program Povray. Namun, jika terjadi kesalahan, program ini tidak menampilkan kesalahan. Oleh karena itu, Anda harus menggunakan

```
gt;povend(lt;keluar);
```

jika ada yang tidak berhasil. Ini akan membiarkan jendela Povray tetap terbuka.

```
povend(h=320,w=480);
```

Berikut adalah contoh yang lebih rumit. Kami menyelesaikan dan menunjukkan titik-titik yang layak dan optimal dalam plot 3D.

```
A=[10,8,4;5,6,8;6,3,2;9,5,6];
b=[10,10,10,10]';
c=[1,1,1];
```

Pertama, mari kita periksa, apakah contoh ini memiliki solusi atau tidak.

```
x=simplex(A,b,c, max, check)'
```

```
[0, 1, 0.5]
```

Ya, benar.

Selanjutnya kita mendefinisikan dua objek. Yang pertama adalah pesawat

```
function oneplane (a,b,look="") ...
```

```
return povplane(a,b,look) endfunction i/prej Kemudian kita tentukan per-
potongan semua setengah ruang dan kubus.
```

```
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 i/pre;
Sekarang, kita bisa merencanakan adegan tersebut.
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); ... Berikut ini adalah lingkaran di sekeliling
optimal.
writeln(povintersection([povsphere(x,0.5),povplane(c,c.x')], ... povlook(red,0.9)));
Dan kesalahan pada arah yang optimal.
writeln(povarrow(x,c0.5,povlook(red)));
Kami menambahkan teks ke layar. Teks hanyalah sebuah objek 3D. Kita
perlu menempatkan dan memutarinya sesuai dengan pandangan kita.
writeln(povtext("Linear Problem",[0,0.2,1.3],size=0.05,rotate=5°)); ... povend();
Contoh Lainnya
Anda dapat menemukan beberapa contoh lain untuk Povray di Euler dalam
file-file berikut.
ja href="Examples/Dandelin Spheres.html" ;Examples/Dandelin Spheres;/a;
ja href="Examples/Donat Math.html" ;Examples/Donat Math;/a;
ja href="Examples/Trefoil Knot.html" ;Examples/Trefoil Knot;/a;
ja href="Examples/Optimization by Affine Scaling.html" ;Examples/Optimization
by Affine Scaling;/a;

```

#### LATIHAN SOAL

1. Buatlah plot 3D dari fungsi

pada rentang

```

function f(x,y) := x2 + y2
plot3d("f(x,y)", -2,2, -2,2)

```

2. Gambarkan permukaan dari fungsi

pada rentang

```

function g(x,y) :=sin(x) cos(y)
plot3d("g(x,y)", -pi,pi, -pi,pi)

```

3. Buat plot 3D dari fungsi

pada rentang

```

function h(x,y) := xexp(-x2 - y2)
plot3d("h(x,y)", -2,2, -2,2, hue)

```

4. Gambarkan dua permukaan dalam satu plot:

dan

pada rentang

```

function f1(x,y) := x2 + y2
function f2(x,y) := 10-x2 + y2
plot3d("f1(x,y)", "f2(x,y)", -3,3, -3,3)

```

5. Buatlah plot 3D dari fungsi

(kerucut) pada rentang

Gunakan opsi `gt;contour` untuk menambahkan garis kontur pada plot.

```

function cone(x,y) := sqrt(x2 + y2)
plot3d("cone(x,y)", -5,5, -5,5, contour)

```

6. Gambarkan permukaan dari fungsi

(fungsi sinc 2D) pada rentang

Gunakan opsi `gt;hue` untuk pewarnaan berdasarkan ketinggian.

```
function sinc2d(x,y) := sin(sqrt(x2 + y2))/(sqrt(x2 + y2) + 1e - 10)
plot3d("sinc2d(x,y)", -10,10, -10,10, hue)
```

7. Gambarkan permukaan dari fungsi (saddle point) pada rentang

Gunakan opsi `gt;frame` untuk menampilkan bingkai plot.

```
function saddle(x,y) := x y exp(-x2 - y2)
plot3d("saddle(x,y)", -2,2, -2,2, frame)
```

8. Buat plot 3D dari fungsi implisit (Steiner Surface) pada rentang

```
plot3d("x2 + y2 + z2 + 2xyz - 1", -2, 2, -2, 2, -2, 2, 50, implicit)
```

9. Gambarkan permukaan dari fungsi pada rentang

Gunakan opsi `gt;points` untuk menampilkan titik-titik sampel.

```
function g(x,y) := exp(-x2 - y2)cos(2pi x)sin(2pi y)
plot3d("g(x,y)", -2,2, -2,2, points)
```

10. Gambarkan permukaan dari fungsi (monkey saddle) pada rentang

Gunakan opsi `gt;contour` untuk menambahkan garis kontur.

```
function monkey(x,y) := x3 - 3xy2
plot3d("monkey(x,y)", -2,2, -2,2, contour)
```

11. Gambarkan permukaan dari fungsi pada rentang

Gunakan opsi `gt;frame` dan `gt;hue` untuk tampilan yang lebih menarik.

```
function s(x,y) := sin(x) + cos(y) + sin(x)cos(y)
plot3d("s(x,y)", -pi,pi, -pi,pi, frame, hue)
```

12. Buat plot 3D dari fungsi implisit (Kummer's Surface) pada rentang dengan  $a = 1.5$ .

```
a = 1.5
1.5
plot3d("x2 + y2 + z2 + 2xyz - a", -2, 2, -2, 2, -2, 2, 50, implicit)
```

13. Gambarkan plot 3D dari fungsi

```
function B(x,y) = y2 - 2x2y + 4x3 + 20x2
2 2 3 2 y - 2 x y + 4 x + 20 x
plot3d("B")
```

14. Gambarkan plot 3D dari fungsi

```
function C(x,y) = cosec(9x)-tan(2y)
cosec(9 x) - tan(2 y)
plot3d("C")
plot3d("C(x,y)", -100,50,0,2pi)
```

15. Gambarkan permukaan dari fungsi

```
z = x*exp(-x2 - y2)
pada rentang
Gunakan opsi gt;contour dan gt;hue bersamaan.
function g(x,y) := xexp(-x2 - y2)
```



```

plot3d("g(x,y)", -2,2, -2,2, contour, hue)
16. Buat plot 3D dari fungsi implisit
(Superquadric) pada rentang
plot3d("x4 + y4 + z4 - 1", -1.5, 1.5, -1.5, 1.5, -1.5, 1.5, 50, implicit)
17. Buat plot 3D dari fungsi implisit
(Cartan's Umbrella) pada rentang
plot3d("x2 + y2 - z3", -2, 2, -2, 2, -2, 2, 50, implicit)
18. Buatlah gabungan 2 silinder dengan fungsi povx() berwarna merah dan
povz() berwarna kuning dan zoom 4
povstart(zoom=4);
c1=povcylinder(-povx,povx,1,povlook(red));... c2=povcylinder(-povy,povy,1,povlook(yellow));...
writeln(povintersection([c1,c2]))
povend();

```

## 5 Kalkulus

Kalkulus dengan EMT Emlys Jenny Ester Sirait (23030630081)

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 `namafungsi := rumusfungsi(untukfungsi*numerik)`,

- \* Menggunakan format `namafungsiamp; = rumusfungsi(untukfungsi * simbolik, namundapatdihitungsecaranumerik)`,

- \* Menggunakan format `namafungsiamp; amp; = rumusfungsi(untukfungsi* simbolikmurni, tidakdapatdihitunglangsung)`,

- \* Fungsi sebagai program EMT.

Setiap format harus diawali dengan perintah `function` (bukan sebagai ekspresi).

Berikut adalah beberapa contoh cara mendefinisikan fungsi:

```
function f(x) := 2x2 + exp(sin(x))//fungsinumerik
```

```
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 ...
```

Silakan Anda plot kurva fungsi di atas!

```

Berikutnya kita definisikan fungsi:
function g(x) := sqrt(x^2 - 3x)/(x + 1)
g(3)
0
g(0)
0
g(1) // kompleks, tidak dapat dihitung oleh fungsi numerik
Function g not found. Try list ... to find functions! Error in: g(1) //
kompleks, tidak dapat dihitung oleh fungsi numerik ...
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:
dan
bersama-sama kurva fungsi f dan g dalam satu bidang koordinat.
f(0:10) // nilai-nilai f(0), f(1), f(2), ..., f(10)
[1, 4.31978, 10.4826, 19.1516, 32.4692, 50.3833, 72.7562, 99.929, 130.69,
163.51, 200.58]
fmap(0:10) // sama dengan f(0:10), berlaku untuk semua fungsi
[1, 4.31978, 10.4826, 19.1516, 32.4692, 50.3833, 72.7562, 99.929, 130.69,
163.51, 200.58]
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
Fungsi tersebut tidak dapat didefinisikan sebagai fungsi numerik secara "in-
line" menggunakan format :=, melainkan didefinisikan sebagai program. Per-
hatikan, kata "map" digunakan agar fungsi dapat menerima vektor sebagai in-
put, 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 end if; end function </pre> 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) = 2E^x // fungsisimbolik
x 2 E
f(a) // nilai fungsisekarasimbolik
f(E) // nilai fungsi berupa bilangan desimal

```

```

30.308524483
f(E),float(
function g(x) = 3x+1
3 x + 1
function h(x) = f(g(x)) // komposisi fungsi
3 x + 1 2 E
plot2d("h(x)",-1,1):
Latihan
1. diberikan fungsi
tentukan nilai k(2,1)+k(2,2)
k(2,1)
96
k(2,2)
108
k(2,1)+k(2,2)
204
2. diberikan fungsi
tentukan nilai k(1,1)+k(2,3)
k(1,1)
15
k(2,3)
128
k(1,1)+k(2,3)
143
3. diberikan fungsi:
function b(x):= 12x^3/(7x^2 - 3x)
b(1)
3
b(1:4)
[3, 4.36364, 6, 7.68]
function f(x,y):= 3(x-1)(y-2)/2+(y-2)(x-3)/2-2(x-1)(y-3)
f(20,10)
30

```

#### 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 - 3x)/(x + 1), x, inf))
limit((x^3 - 13x^2 + 51x - 63)/(x^3 - 4x^2 - 3x + 18), x, 3)

```

maxima: 'limit(( $x^3 - 13 * x^2 + 51 * x - 63$ )/( $x^3 - 4 * x^2 - 3 * x + 18$ ),  $x, 3$ ) =  
 $\text{limit}((x^3 - 13 * x^2 + 51 * x - 63)/(x^3 - 4 * x^2 - 3 * x + 18), x, 3)$   
 Fungsi tersebut diskontinu di titik  $x=3$ . Berikut adalah grafik fungsinya.  
 $\text{aspect}(1.5); \text{plot2d}((x^3 - 13x^2 + 51x - 63)/(x^3 - 4x^2 - 3x + 18), 0, 4); \text{plot2d}(3, -4/5, \text{points}, \text{style} =$   
 $\text{"ow"}, \text{add}) :$   
 $\text{limit}(2x \sin(x)/(1 - \cos(x)), x, 0)$   
 maxima: 'limit( $2 * x * \sin(x)/(1 - \cos(x))$ ),  $x, 0$ ) = limit( $2 * x * \sin(x)/(1 - \cos(x))$ ),  $x, 0$ )  
 Fungsi tersebut diskontinu di titik  $x=0$ . Berikut adalah grafik fungsinya.  
 $\text{plot2d}(2x \sin(x)/(1 - \cos(x)), -\pi, \pi); \text{plot2d}(0, 4, \text{points}, \text{style} = \text{"ow"}, \text{add}) :$   
 $\text{limit}(\cot(7h)/\cot(5h), h, 0)$   
 maxima: showev('limit( $\cot(7 * h)/\cot(5 * h)$ ),  $h, 0$ ))  
 Fungsi tersebut juga diskontinu (karena tidak terdefinisi) di  $x=0$ . Berikut  
 adalah grafiknya.  
 $\text{plot2d}(\cot(7x)/\cot(5x), -0.001, 0.001); \text{plot2d}(0, 5/7, \text{points}, \text{style} = \text{"ow"}, \text{add}) :$   
 $\text{showev}(\text{'limit}(((x/8)^{1/3} - 1)/(x - 8), x, 8))$   
 Tunjukkan limit tersebut dengan grafik, seperti contoh-contoh sebelumnya.  
 $\text{showev}(\text{'limit}(1/(2x - 1), x, 0))$   
 Tunjukkan limit tersebut dengan grafik, seperti contoh-contoh sebelumnya.  
 $\text{showev}(\text{'limit}((x^2 - 3x - 10)/(x - 5), x, 5))$   
 Tunjukkan limit tersebut dengan grafik, seperti contoh-contoh sebelumnya.  
 $\text{showev}(\text{'limit}(\sqrt{x^2 + x} - x, x, \text{inf}))$   
 Tunjukkan limit tersebut dengan grafik, seperti contoh-contoh sebelumnya.  
 $\text{showev}(\text{'limit}(\text{abs}(x - 1)/(x - 1), x, 1, \text{minus}))$   
 Hitung limit di atas untuk  $x$  menuju 1 dari kanan.  
 Tunjukkan limit tersebut dengan grafik, seperti contoh-contoh sebelumnya.  
 $\text{showev}(\text{'limit}(\sin(x)/x, x, 0))$   
 $\text{plot2d}(\sin(x)/x, -\pi, \pi); \text{plot2d}(0, 1, \text{points}, \text{style} = \text{"ow"}, \text{add}) :$   
 $\text{showev}(\text{'limit}(\sin(x^3)/x, x, 0))$   
 Tunjukkan limit tersebut dengan grafik, seperti contoh-contoh sebelumnya.  
 $\text{showev}(\text{'limit}(\log(x), x, \text{minf}))$   
 $\text{showev}(\text{'limit}((-2)^x, x, \text{inf}))$   
 $\text{showev}(\text{'limit}(t - \sqrt{2 - t}, t, 2, \text{minus}))$   
 $\text{showev}(\text{'limit}(t - \sqrt{2 - t}, t, 2, \text{plus}))$   
 $\text{showev}(\text{'limit}(t - \sqrt{2 - t}, t, 5, \text{plus})) // \text{Perhatikan hasilnya}$   
 $\text{plot2d}(x - \sqrt{2 - x}, 0, 2) :$   
 $\text{showev}(\text{'limit}((x^2 - 9)/(2x^2 - 5x - 3), x, 3))$   
 Tunjukkan limit tersebut dengan grafik, seperti contoh-contoh sebelumnya.  
 $\text{showev}(\text{'limit}((1 - \cos(x))/x, x, 0))$   
 Tunjukkan limit tersebut dengan grafik, seperti contoh-contoh sebelumnya.  
 $\text{showev}(\text{'limit}((x^2 + \text{abs}(x))/(x^2 - \text{abs}(x))), x, 0))$   
 Tunjukkan limit tersebut dengan grafik, seperti contoh-contoh sebelumnya.  
 $\text{showev}(\text{'limit}((1 + 1/x)^x, x, \text{inf}))$   
 $\text{plot2d}((1 + 1/x)^x, 0, 1000) :$   
 $\text{showev}(\text{'limit}((1 + k/x)^x, x, \text{inf}))$   
 $\text{showev}(\text{'limit}((1 + x)^{1/x}, x, 0))$   
 Tunjukkan limit tersebut dengan grafik, seperti contoh-contoh sebelumnya.

```
showev('limit((x/(x+k))^x,x,inf))
showev('limit((E^x - E^2)/(x-2),x,2))
```

Tunjukkan limit tersebut dengan grafik, seperti contoh-contoh sebelumnya.

```
showev('limit(sin(1/x),x,0))
showev('limit(sin(1/x),x,inf))
plot2d("sin(1/x)",-0.001,0.001):
```

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. Fungsi 1

```
showev('limit((3x-6)/(x+2),x,2))
plot2d("(3x-6)/(x+2)",-2,3.5,-1,5):
```

2. Fungsi 2

```
showev('limit(cos(2x)/(sin(x)-cos(x)),x,0))
plot2d("cos(2x)/(sin(x)-cos(x))",-1,1):
```

3. Fungsi 3

```
showev('limit(((2x^2-2x+5)/(3x^2+x-6)),x,3))
plot2d("(2x^2-2x+5)/(3x^2+x-6)",-2,10,-10,5):
```

4. Fungsi 4

```
showev('limit((4x^2-3),x,0))
plot2d("(4x^2-3)":
```

5. Fungsi 5

```
showev('limit((x^(x(x))),x,0,plus))
plot2d("(x^(x(x)))",-3,3,-1,7):
```

6. Fungsi 6

```
showev('limit((3xtan(x))/(1-cos(4x)),x,0))
plot2d("(3xtan(x))/(1-cos(4x))",-pi/2,2pi,0,2pi):
```

Turunan Fungsi

Definisi turunan:

Berikut adalah contoh-contoh menentukan turunan fungsi dengan menggunakan definisi turunan (limit).

```
showev('limit(((x+h)^2-x^2)/h,h,0))/turunanx^2
```

p = expand((x+h)<sup>2</sup> - x<sup>2</sup>)|simplify;p //pembilang dijabarkan dan disederhanakan

```
q=ratsimp(p/h); q//ekspresiyangakandihitunglimitnyadisederhanakan
limit(q,h,0)//nilailimitsebagaiturunan
```

```
showev('limit(((x+h)^n-x^n)/h,h,0))/turunanx^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, ekspansikan (x+h)<sup>n</sup> dengan menggunakan teorema binomial.

```
showev('limit((sin(x+h)-sin(x))/h,h,0))/turunansin(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.

```
showev('limit((log(x+h)-log(x))/h,h,0))//turunanlog(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.

```
showev('limit((1/(x+h)-1/x)/h,h,0))//turunan1/x
```

```
showev('limit((E^(x+h)-E^x)/h,h,0))//turunanf(x)=e^x
```

Answering "Is x an integer?" with "integer" Answering "Is x an integer?" with "integer" Answering "Is x an integer?" with "integer" 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))//turunanf(x)=e^x...`

Maxima bermasalah dengan limit:

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))//turunanf(x)=e^x
```

function  $f(x) = x^x$

x x

```
showev('limit(f(x),x,0))
```

Silakan Anda gambar kurva

```
showev('limit((f(x+h)-f(x))/h,h,0))//turunanf(x)=x^x
```

Di sini Maxima juga bermasalah terkait limit:

Dalam hal ini diperlukan asumsi nilai x.

```
assume(x 0); showev('limit((f(x+h)-f(x))/h,h,0))//turunanf(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 gt; 0]
```

```
forget(x;0)
```

```
[x lt; 0]
```

```
facts()
```

```
[]
```

```
showev('limit((asin(x+h)-asin(x))/h,h,0))//turunanarcsin(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))/turunantan(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

```
plot2d(["f(x)", "df(x)"],-pi,pi,color=[blue,red]):
function f(x) = sin(3x5 + 7)2
```

```
2 5 sin (3 x + 7)
```

```
diff(f,3), diffc(f,3)
```

```
1198.32948904 1198.72863721
```

Apakah perbedaan diff dan diffc?

```
showev('diff f(f(x),x))
```

```
float(
```

```
plot2d(f,0,3.1):
```

```
function f(x) = 5cos(2x)-2xsin(2x) // mendefinisikan fungsi f
```

```
5 cos(2 x) - 2 x sin(2 x)
```

```
function df(x) = diff(f(x),x) // fd(x) = f'(x)
```

```
- 12 sin(2 x) - 4 x cos(2 x)
```

```
'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)"],0,2pi,color=[blue,red]): //grafik fungsi dan turunan-nya
```

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.

1. Fungsi 1

```
function f(x) := sqrt(x)4
```

```
showev('limit((sqrt(x+h)-sqrt(x)4)/h,h,0))/turunansqrt(x)4
```

```
function df(x) = limit((sqrt(x+h)-sqrt(x)4)/h,h,0); df(x)/df(x) = f'(x)
```

```

plot2d(["f(x)", "df(x)"],-pi,pi,color=[blue,red]), label("f(x)",-2,11), label("df(x)",-
2,-10):
2. Fungsi 2
function f(x) := x^2
showev('limit(((x+h)^2 - x^2)/h), h, 0))/turunanx^2
function df(x) = limit(((x+h)^2 - x^2)/h), h, 0); df(x) // df(x) = f'(x)
plot2d(["f(x)", "df(x)"],-pi,pi,color=[blue,red]), label("f(x)",2,0.6), label("df(x)",2,0.17):
3. Fungsi 3
function f(x) := sqrt(tan(x))
showev('limit((sqrt(tan(x+h))-sqrt(tan(x)))/h, h, 0))/turunanexp(x) x
cos(x)
function df(x) = limit((sqrt(tan(x+h))-sqrt(tan(x)))/h, h, 0); df(x) // df(x) =
f'(x)
plot2d(["f(x)", "df(x)"],-10,10,-10,10,color=[blue,red]), label("f(x)",4.5,0), la-
bel("df(x)",5.5,5):
4. Fungsi 4
function f(x) := sin(x)cos(x)
showev('limit(((sin(x+h)cos(x+h))-sin(x)cos(x))/h, h, 0))/turunansin(x) x
cos(x)
function df(x) = limit(((sin(x+h)cos(x+h))-sin(x)cos(x))/h, h, 0); df(x) // df(x) =
f'(x)
plot2d(["f(x)", "df(x)"],-pi,pi,color=[blue,red]), label("f(x)",1,0), label("df(x)",2.3,1.2):
5. Fungsi 5
function f(x) := cos(1/x)
showev('limit((cos(1/(x+h)) - cos(1/x))/h, h, 0))/turunancos(1/x)
function df(x) = limit((cos(1/(x+h))-cos(1/x))/h, h, 0); df(x) // df(x) = f'(x)
plot2d(["f(x)", "df(x)"],-pi,pi,color=[blue,red]), label("f(x)",2,0.4), label("df(x)",1,-
0.5):
6. Fungsi 6
function f(x) := cos(1/x)
showev('limit((cos(1/(x+h)) - cos(1/x))/h, h, 0))/turunancos(1/x)
function df(x) = limit((cos(1/(x+h))-cos(1/x))/h, h, 0); df(x) // df(x) = f'(x)
plot2d(["f(x)", "df(x)"],-pi,pi,color=[blue,red]), label("f(x)",2,0.4), label("df(x)",1,-
0.5):

```

#### 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:

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, 2pi):
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(2x + 1), x))
showev('integrate(x^2*sqrt(2x + 1), x, 0, 2))
ratsimp(
showev('integrate((sin(sqrt(x) + a)*sqrt(x))/sqrt(x), x, 0, pi^2))
factor(
function map f(x) = E(-x^2)
2 - x E
showev('integrate(f(x), x))
```

Fungsi  $f$  tidak memiliki antiturunan, integralnya masih memuat integral lain.

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  $x = 0.1, 0.2, 0.3, \dots, 3.2$ .

```
0.1sum(f(x+0.1)) // cek langsung dengan perhitungan numerik EMT
```

```
Function f not found. Try list ... to find functions! Error in: 0.1*sum(f(x+0.1))
// cek langsung dengan perhitungan numerik E ...
```

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
```

```
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(3x^5 + 7)^2
```

```
2 5 sin (3 x + 7)
```

```
integrate(f,0,1)
```

```
0.542581176074
```

```
showev('integrate(f(x),x,0,1))
```

```
1 1 pi / gamma(-) sin(14) sin(-) [2 5 5 10 I sin (3 x + 7) dx = ——
——] 1/5 / 10 6 0 4/5 1 4/5 1 - (((6 gamma_incomplete(-, 6I) +
6gamma_incomplete(-, -6I))554/51sin(14)+(6Igamma_incomplete(-, 6I)54/51pi-
6Igamma_incomplete(-, -6I))cos(14))sin(--) - 60)/120510
```

```
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(0.2,6.0I)-4.192962712629476Igam
```

```
showev('integrate(xexp(-x), x, 0, 1))//Integraltentu(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) : //Plotdaerahantara2kurva
```

```
a=solve("x^3-x+x^2", 0), b = solve("x^3-x+x^2", 1)//absistitik-titikpotongkeduakurva
```

```
0 0.61803398875
```

```
integrate("(-x^2) - (x^3 - x)", a, b)//luasdaerahyangdiarsir
```

```
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))//Nilaiintegralsecaraeksak
float(
Panjang Kurva
Hitunglah panjang kurva berikut ini dan luas daerah di dalam kurva terse-
but.
dengan
t=linspace(0,2pi,1000); r=1+sin(3t)/2; x=rcos(t); y=rsin(t); ... plot2d(x,y, filled,fillcolor=red,style=" ",r
// Kita gambar kurvanya terlebih dahulu
function r(t) = 1+sin(3t)/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,0,2pi)//panjang(keliling)kurva
Maxima gagal melakukan perhitungan eksak integral tersebut.
Berikut kita hitung integralnya secara umerik dengan perintah EMT.
integrate("ds(x)",0,2pi)
Function ds not found. Try list ... to find functions! Error in expression:
ds(x) return expr(x,args()); Error in map. if maps then return gauss: if maps
then y=adaptivegauss: t1=gauss(f,c,c+h;args(),=maps);Try"traceerrors"toinspectlocalvariablesaftererro
returnadaptivegauss(f,a,b,eps*1000;args(),=maps);
Spiral Logaritmik
a=0.1; plot2d("exp(ax)cos(x)","exp(ax)sin(x)",r=2,xmin=0,xmax=2pi):
kill(a) // hapus ekspresi a
done
function fx(t) = exp(at)cos(t); 'fx(t) = fx(t)
function fy(t) = exp(at)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
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 dihipotesis 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 mendap-
atkan 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:

```
z = x3 + y3 - 3xy; 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.

```
zwithy = lx, sol = solve(
```

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),xf(x),xmin=-0.5,xmax=2,a=0,b=2,c=0,d=2,r=1.5):
```

Elemen panjang kurva adalah:

```
function ds(l) = ratsimp(sqrt(diff(f(l),l)2 + diff(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.

```
2integrate("ds(x)",0,1)
```

```
4.91748872168
```

2romberg(ds(x),0,1)// perintah Euler lain untuk menghitung nilai hampiran integral yang sama

```
4.91748872168
```

Perhitungan di atas dapat dilakukan untuk sebarang fungsi x dan y dengan mendefinisikan fungsi EMT, misalnya 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)"); returnromberg(ds,a,b); endfunction <
/pre > panjangkurva("x", "x2", -1, 1)// cek untuk menghitung panjang kurva parabola sebelumnya
2.95788571509
```

Bandingkan dengan nilai eksak di atas.

2panjangkurva(mxm("f(x)"),mxm("xf(x)"),0,1) // cek contoh terakhir, bandingkan hasilnya!

```
4.91748872168
```

Kita hitung panjang spiral Archimides berikut ini dengan fungsi tersebut.

```
plot2d("xcos(x)", "xsin(x)", xmin=0, xmax=2pi, square=1):
```

```
panjangkurva("xcos(x)", "xsin(x)", 0, 2pi)
```

```
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)
2 2 sqrt(diff (fy, x) + diff (fx, x))
sol = ds(xcos(x),xsin(x)); sol//Kitagunakanuntukmenghitungpanjangkurvaterakhir diatas
sol|trigreduce|expand,integrate(
21.2562941482

```

Hasilnya sama dengan perhitungan menggunakan fungsi EMT.  
Berikut adalah contoh lain penggunaan fungsi Maxima tersebut.

```

plot2d("3x^2-1", "3x^3-1", xmin = -1/sqrt(3), xmax = 1/sqrt(3), square =
1) :
sol = radcan(ds(3x^2 - 1, 3x^3 - 1));sol
showev('integrate(sol, x, 0, 1/sqrt(3))),2float(

```

Sikloid

Berikut kita akan menghitung panjang kurva lintasan (sikloid) suatu titik pada lingkaran yang berputar ke kanan pada permukaan datar. Misalkan jari-jari lingkaran tersebut adalah  $r$ . Posisi titik pusat lingkaran pada saat  $t$  adalah: Misalkan posisi titik pada lingkaran tersebut mula-mula  $(0,0)$  dan posisinya pada saat  $t$  adalah:

Berikut kita plot lintasan tersebut dan beberapa posisi lingkaran ketika  $t=0$ ,  $t=\pi/2$ ,  $t=r\pi$ .

```

x = r(t-sin(t))
[0, 1.66665833335744e-7 r, 1.33330666692022e-6 r, 4.499797504338432e-6 r,
1.066581336583994e-5 r, 2.083072932167196e-5 r, 3.599352055540239e-5 r, 5.71526624672386e-
5 r, 8.530603082730626e-5 r, 1.214508019889565e-4 r, 1.665833531718508e-4
r, 2.216991628251896e-4 r, 2.877927110806339e-4 r, 3.658573803051457e-4 r,
4.568853557635201e-4 r, 5.618675264007778e-4 r, 6.817933857540259e-4 r, 8.176509330039827e-
4 r, 9.704265741758145e-4 r, 0.001141105023499428 r, 0.001330669204938795
r, 0.001540100153900437 r, 0.001770376919130678 r, 0.002022476464811601 r,
0.002297373572865413 r, 0.002596040745477063 r, 0.002919448107844891 r, 0.003268563311168871
r, 0.003644351435886262 r, 0.004047774895164447 r, 0.004479793338660443 r,
0.0049413635565565 r, 0.005433439383882244 r, 0.005956971605131645 r, 0.006512907859185624
r, 0.007102192544548636 r, 0.007725766724910044 r, 0.00838456803503801 r,
0.009079530587017326 r, 0.009811584876838586 r, 0.0105816576913495 r, 0.01139067201557714
r, 0.01223954694042984 r, 0.01312919757078923 r, 0.01406053493400045 r, 0.01503446588876983
r, 0.01605189303448024 r, 0.01711371462093175 r, 0.01822082445851714 r, 0.01937411182884202
r, 0.02057446139579705 r, 0.02182275311709253 r, 0.02311986215626333 r, 0.02446665879515308
r, 0.02586400834688696 r, 0.02731277106934082 r, 0.02881380207911666 r, 0.03036795126603076
r, 0.03197606320812652 r, 0.0336389770872163 r, 0.03535752660496472 r, 0.03713253989951881
r, 0.03896483946269502 r, 0.0408552420577305 r, 0.04280455863760801 r, 0.04481359426396048
r, 0.04688314802656623 r, 0.04901401296344043 r, 0.05120697598153157 r, 0.05346281777803219
r, 0.05578231276230905 r, 0.05816622897846346 r, 0.06061532802852698 r, 0.0631303649963022
r, 0.06571208837185505 r, 0.06836123997666599 r, 0.07107855488944881 r, 0.07386476137264342
r, 0.07672058079958999 r, 0.07964672758239233 r, 0.08264390910047736 r, 0.0857128256298576
r, 0.08885417027310427 r, 0.09206862889003742 r, 0.09535688002914089 r, 0.0987195948597075
r, 0.1021574371047232 r, 0.1056710629744951 r, 0.1092611211010309 r, 0.1129282524731764
r, 0.1166730903725168 r, 0.1204962603100498 r, 0.1243983799636342 r, 0.1283800591162231

```

```

r, 0.1324418995948859 r, 0.1365844952106265 r, 0.140808431699002 r, 0.1451142866615502
r, 0.1495026295080298 r, 0.1539740213994798 r]
y = r(1-cos(t))
[0, 4.999958333473664e-5 r, 1.999933334222437e-4 r, 4.499662510124569e-
4 r, 7.998933390220841e-4 r, 0.001249739605033717 r, 0.00179946006479581
r, 0.002448999746720415 r, 0.003198293697380561 r, 0.004047266988005727 r,
0.004995834721974179 r, 0.006043902043303184 r, 0.00719136414613375 r, 0.00843810628521191
r, 0.009784003787362772 r, 0.01122892206395776 r, 0.01277271662437307 r, 0.01441523309043924
r, 0.01615630721187855 r, 0.01799576488272969 r, 0.01993342215875837 r, 0.02196908527585173
r, 0.02410255066939448 r, 0.02633360499462523 r, 0.02866202514797045 r, 0.03108757828935527
r, 0.03361002186548678 r, 0.03622910363410947 r, 0.03894456168922911 r, 0.04175612448730281
r, 0.04466351087439402 r, 0.04766643011428662 r, 0.05076458191755917 r, 0.0539576564716131
r, 0.05724533447165381 r, 0.06062728715262111 r, 0.06410317632206519 r, 0.06767265439396564
r, 0.07133536442348987 r, 0.07509094014268702 r, 0.07893900599711501 r, 0.08287917718339499
r, 0.08691105968769186 r, 0.09103425032511492 r, 0.09524833678003664 r, 0.09955289764732322
r, 0.1039475024744748 r, 0.1084317118046711 r, 0.113005077220716 r, 0.1176671413898787
r, 0.1224174381096274 r, 0.1272554923542488 r, 0.1321808203223502 r, 0.1371929294852391
r, 0.1422913186361759 r, 0.1474754779404944 r, 0.152744888986584 r, 0.1580990248377314
r, 0.1635373500848132 r, 0.1690593208998367 r, 0.1746643850903219 r, 0.1803519821545206
r, 0.1861215433374662 r, 0.1919724916878484 r, 0.1979042421157076 r, 0.2039162014509444
r, 0.2100077685026351 r, 0.216178334119151 r, 0.2224272812490723 r, 0.2287539850028937
r, 0.2351578127155118 r, 0.2416381240094921 r, 0.2481942708591053 r, 0.2548255976551299
r, 0.2615314412704124 r, 0.2683111311261794 r, 0.2751639892590951 r, 0.2820893303890569
r, 0.2890864619877229 r, 0.2961546843477643 r, 0.3032932906528349 r, 0.3105015670482534
r, 0.3177787927123868 r, 0.3251242399287333 r, 0.3325371741586922 r, 0.3400168541150183
r, 0.3475625318359485 r, 0.3551734527599992 r, 0.3628488558014202 r, 0.3705879734263036
r, 0.3783900317293359 r, 0.3862542505111889 r, 0.3941798433565377 r, 0.4021660177127022
r, 0.4102119749689023 r, 0.418316910536117 r, 0.4264800139275439 r, 0.4347004688396462
r, 0.4429774532337832 r, 0.451310139418413 r]

```

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,col
... 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 pan-
jang kurva sikloid
ds = trigsimp(ds); ds
showev('integrate(ds,x,0,2pi))//hitungpanjangsikloidsatuputaranpenuh
integrate(mxm("ds"),0,2pi) // hitung secara numerik
8
romberg(mxm("ds"),0,2pi) // cara lain hitung secara numerik
8

```

Perhatikan, seperti terlihat pada gambar, panjang sikloid lebih besar daripada keliling lingkarannya, yakni:

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:

dengan  $R$  adalah radius kelengkungan. (Setiap lingkaran memiliki kelengkungan ini pada setiap titiknya, dapat diartikan, setiap lingkaran berputar  $2\pi$  sejauh  $2\pi R$ .)

Definisi ini sulit dimanipulasi dan dinyatakan ke dalam rumus untuk kurva umum. Oleh karena itu digunakan definisi lain yang ekuivalen.

Definisi Kurvatur dengan Fungsi Parametrik Panjang Kurva

Setiap kurva diferensiabel dapat dinyatakan dengan persamaan parametrik terhadap panjang kurva  $s$ :

dengan  $x$  dan  $y$  adalah fungsi riil yang diferensiabel, yang memenuhi:

Ini berarti bahwa vektor singgung

memiliki norm 1 dan merupakan vektor singgung satuan.

Apabila kurvanya memiliki turunan kedua, artinya turunan kedua  $x$  dan  $y$  ada, maka  $T'(s)$  ada. Vektor ini merupakan normal kurva yang arahnya menuju pusat kurvatur, norm-nya merupakan nilai kurvatur (kelengkungan):

Nilai

disebut jari-jari (radius) kelengkungan kurva.

Bilangan riil

disebut nilai kelengkungan bertanda.

Contoh:

Akan ditentukan kurvatur lingkaran

$f_x = r \cos(t)$ ;  $f_y = r \sin(t)$ ;

$s = \int_0^t \sqrt{(\frac{df_x}{dt})^2 + (\frac{df_y}{dt})^2} dt$ ;  $s // \text{elemen panjang kurva, panjang busur}$

$t = t(s)$ ;  $f_x = r \cos(s/r)$ ;  $f_y = r \sin(s/r)$ ; // definisi ulang persamaan parametrik terhadap  $s$  dengan substitusi  $t=s/r$

$k = \frac{1}{r} = \frac{1}{\sqrt{(\frac{df_x}{ds})^2 + (\frac{df_y}{ds})^2}}$ ;  $k // \text{nilai kurvatur lingkaran}$

dengan menggunakan definisi di atas

Untuk representasi parametrik umum, misalkan

merupakan persamaan parametrik untuk kurva bidang yang terdiferensialkan dua kali. Kurvatur untuk kurva tersebut didefinisikan sebagai

Selanjutnya, pembilang pada persamaan di atas dapat dicari sebagai berikut.

Jadi, rumus kurvatur untuk kurva parametrik

adalah

Jika kurvanya dinyatakan dengan persamaan parametrik pada koordinat kutub

maka rumus kurvturnya adalah

(Silakan Anda turunkan rumus tersebut!)

Contoh:

Lingkaran dengan pusat (0,0) dan jari-jari r dapat dinyatakan dengan persamaan parametrik

Nilai kelengkungan lingkaran tersebut adalah

Hasil cocok dengan definisi kurvatur suatu kelengkungan.

Kurva

dapat dinyatakan ke dalam persamaan parametrik

sehingga kurvturnya adalah

Contoh:

Akan ditentukan kurvatur parabola

function f(x) = ax<sup>2</sup> + bx + c; y=f(x)

function k(x) = (diff(f(x),x,2))/(1+diff(f(x),x)<sup>2</sup>)<sup>(3/2)</sup>; k(x)=k(x) // kelengkungan parabola

function f(x) = x<sup>2</sup> + 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)<sup>2</sup>)<sup>(3/2)</sup>; k(x)=k(x) // kelengkungan parabola

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

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

dengan turunan-turunan parsial

berlaku

sehingga kurvturnya adalah

(Silakan Anda turunkan sendiri!)

Contoh 1:

Parabola

dapat dinyatakan ke dalam persamaan implisit

function F(x,y) = ax<sup>2</sup> + bx + 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),  
Fyy = diff(F(x,y),y,2)

2 a x + b

2 a

- 1

0

0

function k(x) = (Fy<sup>2</sup>Fxx - 2FxFyFxy + Fx<sup>2</sup>Fyy)/(Fx<sup>2</sup> + Fy<sup>2</sup>)<sup>(3/2)</sup>; k(x)=k(x)  
// kurvatur parabola tersebut

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 \* diintegrasikan (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

(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:

(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:

\* 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).

\* Gambarkan kurva asli, kurva kurvatur, kurva jari-jari lingkaran \* oskulasi, dan salah satu lingkaran oskulasinya.

1. Fungsi 1

```
function f(x) = sqrt(24-x^2);f(x)
showev('integrate(f(x),x)')
showev('integrate(f(x),x,1,2)')
x=-2:0.04:1; plot2d(x,f(x+0.01),bar); plot2d("f(x)",1,2,add):
```

2. Fungsi 2

```
t = makelist(a,a,0,1-0.01,0.01);
fx = makelist(f(t[i]+0.01),i,1,length(t));
function f(x) = x^2 + 50;f(x)
x=0:0.1:pi-0.01; plot2d(x,f(x+0.01),bar); plot2d("f(x)",0,pi,add):
0.01sum(f(x+0.01))
17.051552
```

3. Fungsi 3

```
function f(x) = 5x^2;f(x)
```

```
showev('integrate(f(x), x))
showev('integrate(f(x), x, 2, 3))
x=0.01:0.03:4; plot2d(x,f(x+0.01), bar); plot2d("f(x)",2,3, add):
```

#### 4. Fungsi 4

```
function f(x) = cos(2x+5); f(x)
showev('integrate(f(x), x))
showev('integrate(f(x), x, pi, 2pi))
x=0:0.05:pi-0.1; plot2d(x,f(x+0.03), bar); plot2d("f(x)",pi,2pi, add):
```

#### 5. Fungsi 5

```
function f(x) = (sin(x))(cos((x)))^2;f(x)
showev('integrate(f(x), x))
showev('integrate(f(x), x, 0, pi))
x=-pi:0.04:pi; plot2d(x,f(x+0.01), bar); plot2d("f(x)",0,pi, add):
```

#### 6. Fungsi 6

```
function f(x) = (x^2(2 - x^3)^(1/2));f(x)
showev('integrate(f(x), x))
showev('integrate(f(x), x, 0, 1))
x=-1:0.04:1; plot2d(x,f(x+0.01), bar); plot2d("f(x)",0,1, 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 berbagai 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<sub>i</sub>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]

1:4:50

[1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49]

Iterasi dan Barisan

EMT menyediakan fungsi `iterate("g(x)", x0, n)` untuk melakukan iterasi

Berikut ini disajikan contoh-contoh penggunaan iterasi dan rekursi dengan EMT. Contoh pertama menunjukkan pertumbuhan dari nilai awal 1000 dengan laju pertambahan 5

```
q=1.05; iterate("xq",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 610000 per bulan, tabungan sebesar 1 juta tanpa diambil selama sekitar 10 tahunan akan habis diambil oleh bank!

```
r=0.005; plot2d(iterate("(1+0.8r)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(0.8rA - a, A),
```

Berikut didefinisikan fungsi untuk menghitung saldo tabungan, kemudian dilakukan iterasi.

```
function saldo(x,r,a) := round(((1+0.8r)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.

```
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  $n = 2, 4, 5, \dots$

```
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

dengan  $A$  suatu matriks  $2 \times 2$ , dan setiap  $x[n]$  merupakan matriks/vektor  $2 \times 1$ .

```
A=[1,1;1,2]; function suku(x,n) := A.x[n-1]
```

```
sequence("suku",[1;1],6)
```

```
1 2 5 13 34 89 1 3 8 21 55 144
```

Hasil yang sama juga dapat diperoleh dengan menggunakan fungsi perpangkatan matriks "matrixpower()". Cara ini lebih cepat, karena hanya menggunakan perkalian matriks sebanyak  $\log_2(n)$ .

```
sequence("matrixpower(A,n).[1;1]",1,6)
```

```
1 5 13 34 89 233 1 8 21 55 144 377
```

Spiral Theodorus

image: *Spiral<sub>o</sub>fTheodorus.png*

Spiral Theodorus (spiral segitiga siku-siku) dapat digambar secara rekursif.

Rumus rekursifnya adalah:

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"),0.4):
```

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 j/prej. Jika dilakukan it-
erasi 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

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 ival().

```
function s(x) := ival("(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:

```
Satu-satunya solusi adalah
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.

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

```
0.999999999999999778,1.000000000000000022 ... 4.99999999999999911,5.000000000000000089 ...
```

Iterasi berikut konvergen sangat lambat.

```
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 < /pre > Misalnya, berikut adalah iterasi untuk mendapatkan
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, 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 = x3 - 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 < /pre >

```

*Kitamula dengan menentukan titik-titik grid pada bidang kompleksnya.*

```

r=1.5; x=linspace(-r,r,501); Z=x+Ix'; 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 ex

```

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, ex
plot2d(mxm2str("deret"), add,color=4:6): // plot ketiga deret Taylor ham-
piran fungsi tersebut

```

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(kx)/k,k,1,5)

```

Berikut adalah cara menggambar kurva  
`plot2d(sum(sin((2k+1)x)/(2k+1),k,0,20),0,2pi):`  
Hal serupa juga dapat dilakukan dengan menggunakan matriks, misalkan kita akan menggambar kurva  
`x=linspace(0,2pi,1000); k=1:100; y=sum(sin(kx')/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  $x^x dx = 1$ .  
`mxmtable("diffat(x^x, 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 :
```

*menghitungderetsecarasimbolik*  
`' 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 : menghitungnilaiekspresi`  
`' 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/(2k-1), k, 1, inf) = factor(ev(sum((-1)^k/(2k-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(ax^k, k, 0, inf) = ev(sum(ax^k, k, 0, inf), simpsum = true), forget(abs(x)<1);`  
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:  
`' e^x = taylor(exp(x), x, 0, 10)//deretTaylore^x disekitar x = 0, sampaisukuke-`  
11  
`' log(x) = taylor(log(x), x, 1, 10)//deretlog(x) disekitar x = 1`

## 6 Geometri

MATERI GEOMETRI Nama : Emlys Jenny Ester Sirait  
NIM : 23030630081  
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 setPlotrangle(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 vektor v 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 lt;ABC
angleBisector(A, B, C): garis bagi sudut lt;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 lingkaran  $c$   
`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 membentuk suatu segitiga  
`doublespread(sa)`: Spread sudut rangkap  $2\phi$ , dengan  $sa = \sin(\phi)^2$  *spreada*.  
 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 poin dan plot mereka.  
`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 meliputi fungsi untuk membuat garis dan lingkaran. Format garis adalah  $[a, b, c]$ , yang mewakili garis dengan persamaan  $ax+by=c$ .  
`lineThrough(B,C)` // garis yang melalui  $B$  dan  $C$   
`[-1, 2, 2]`  
 Hitunglah 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  $ABC$ :  
`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 di C
degprint(computeAngle(B,C,A))
36°52'11.63"
Sekarang lingkaran luar segitiga.
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"): // gam-
bar lingkaran dalam
Latihan
1. Tentukan ketiga titik singgung lingkaran dalam dengan sisi-sisi segitiga
ABC.
setPlotRange(-3,5,-3,5);
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.
plotSegment(A,B,"c")
plotSegment(B,C,"a")
plotSegment(A,C,"b")
aspect(1);
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.

5. Gambar jari-jari lingkaran dalam.

```

r=norm(P-projectToLine(P,lineThrough(A,B)))
1.52896119631
plotCircle(circleWithCenter(P,r),"Lingkaran dalam segitiga ABC"):

```

Contoh 2: Geometri Simbolik

Kita dapat menghitung geometri eksak dan simbolik menggunakan Maxima.

File `geometri.e` menyediakan fungsi yang sama (dan lebih banyak lagi) di Maxima. Namun, kita dapat menggunakan perhitungan simbolis 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 memberikan perhitungan simbolis.

```
c = lineThrough(B,C) // c=BC
```

```
[- 1, 2, 2]
```

Kita bisa mendapatkan persamaan garis dengan mudah.

```
getLineEquation(c,x,y),solve(
```

```
getLineEquation(lineThrough(A,[x1,y1]),x,y)//persamaangaris 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 titik potong garis c=BC dan h
```

```
2 6 [-, -] 5 5
```

```
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 bagi sudut
```

```
P() // hasilnya sama dengan perhitungan sebelumnya
```

```
[0.86038, 0.86038]
```

Garis dan Lingkaran yang Berpotongan

Tentu saja, kita juga dapat 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 potong.

```
P1,P2,f=lineCircleIntersections(l,c);
P1, P2,
[4.64575, -1.64575] [-0.645751, 3.64575]
plotPoint(P1); plotPoint(P2):
```

Begitu pula di 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, //titikpotonglingkarancdangarisl
```

Akan ditunjukkan bahwa sudut-sudut yang menghadap busur 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 terlibat. Tetapi kita dapat menyederhanakannya, jika kita memecahkan 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:  
Artinya kedua garis tegak lurus.

Contoh 3: Rumus Heron

Rumus Heron menyatakan bahwa luas segitiga dengan panjang sisi-sisi a, b dan c adalah:

Untuk membuktikan hal ini kita misalkan C(0,0), B(a,0) dan A(x,y), b=AC, c=AB. Luas segitiga ABC adalah

Nilai y didapat dengan menyelesaikan sistem persamaan:

```

sol = solve([x^2 + y^2 = b^2, (x - a)^2 + y^2 = c^2], [x, y]) :
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);
sol = solve([x^2 + y^2 = b^2, (x - a)^2 + y^2 = c^2], [x, y]) :

```

Ekstrak solusi y.

```

ysol = y with sol[2][2]; ysol

```

Maxima said: part: invalid index of list or matrix. - an error. To debug this try: debugmode(true);

Error in: ysol amp;= y with sol[2][2]; ysol...

Kami mendapatkan rumus Heron.

```

function H(a,b,c) = sqrt(factor((ysola/2)^2)); 'H(a,b,c)=H(a,b,c)
'Luas = H(3,4,5)//luassegitigadenganpanjangsisi - sisi3,4,5

```

Tentu saja, setiap segitiga persegi panjang adalah kasus yang terkenal.

```

H(3,4,5) //luas segitiga siku-siku dengan panjang sisi 3, 4, 5

```

Variable or function ysol not found. Try "trace errors" to inspect local variables after errors. H: useglobal; return abs(a)\*abs(ysol)/2 Error in: H(3,4,5) //luas segitiga siku-siku dengan panjang sisi 3, 4, 5 ...

Dan juga jelas, bahwa ini adalah segitiga dengan luas maksimal dan dua sisi 3 dan 4.

```

aspect (1.5); plot2d(H(3,4,x),1,7): // Kurva luas segitiga sengan panjang
sisi 3, 4, x (1i= x i=7)

```

Variable or function ysol not found. Error in expression: 3\*abs(ysol)/2

```

y0=f(x[1], args()); adaptiveevalone : s = Try"traceerrors" toinspectlocalvariables after errors.plot2d :
dw/n, dw/n^2, dw/n, auto; args());

```

Kasus umum juga berfungsi.

```

solve(diff(H(a,b,c)^2, c) = 0, c)

```

Maxima said: diff: second argument must be a variable; found [1,0,4] - an error. To debug this try: debugmode(true);

Error in: solve(diff(H(a,b,c)^2, c) = 0, c)...

Sekarang mari kita cari himpunan semua titik di mana b+c=d untuk beberapa konstanta d. Diketahui bahwa ini adalah elips.

```

s1 = subst(d-c,b,sol[2]); s1

```

Maxima said: part: invalid index of list or matrix. – an error. To debug this try: debugmode(true);

Error in: s1 amp;:= subst(d-c,b,sol[2]); s1...

Dan buat 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 bisa menggambar setnya. Sisi b bervariasi dari 1 hingga 4.

Diketahui bahwa kita mendapatkan elips.

aspect(1); plot2d(fx(3,x,5),fy(3,x,5),xmin=1,xmax=4,square=1):

Kita dapat memeriksa persamaan umum untuk elips ini, yaitu.

di mana (xm,ym) adalah pusat, dan u dan v adalah setengah sumbu.

$ratsimp((fx(a, c, d) - a/2)^2/u^2 + fy(a, c, d)^2/v^2 \text{ with } [u = d/2, v = \sqrt{d^2 - a^2}]/2)$

Kita lihat bahwa tinggi dan luas segitiga adalah maksimal untuk x=0. Jadi luas segitiga dengan a+b+c=d maksimal jika segitiga sama sisi. Kami ingin menurunkan ini secara analitis.

eqns = [diff(H(a,b,d-(a+b))^2, a) = 0, diff(H(a,b,d-(a+b))^2, b) = 0]; eqns

Kami mendapatkan beberapa minima, yang termasuk dalam segitiga dengan satu sisi 0, dan solusinya a=b=c=d/3.

solve(eqns, [a, b])

Ada juga metode Lagrange, memaksimalkan  $H(a,b,c)^2$  terhadap  $a+b+d = d$ .

solve([diff(H(a,b,c)^2, a) = la, diff(H(a,b,c)^2, b) = la :

Closing bracket missing. Found: solve([diff(H(a,b,c)^2, a) = la, diff(H(a,b,c)^2, b) = la : Brackets : 1(, 1[Symbolic expression expected. Error in : ...ve([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])

Variable or function a not found. Error in: diff(H(a,b,c)^2, c) = la, a + b + c = d], [a, b, c, la])...

Kita bisa membuat plot situasinya

Pertama-tama atur poin di Maxima.

A = at([x,y],sol[2]); A...

Maxima said: part: invalid index of list or matrix. – an error. To debug this try: debugmode(true);

Error in: A amp;:= at([x,y],sol[2]); A...

B = [0,0]; B, C = [a, 0]; C

Kemudian atur rentang 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");

Variable a1 not found! Use global variables or parameters for string evaluation. Error in Evaluate, superfluous characters found. Try "trace errors" to inspect local variables after errors. mxmeval: return evaluate(mxmeval(s)); Error in: ... otPoint(mxmeval("C"), "C"); plotPoint(mxmeval("A"), "A"): ...

Plot segmen.

plotSegment(mxmeval("A"), mxmeval("C")); ... plotSegment(mxmeval("B"), mxmeval("C")); ... plotSegment(mxmeval("B"), mxmeval("A"));

Variable a1 not found! Use global variables or parameters for string evaluation. Error in Evaluate, superfluous characters found. Try "trace errors" to

```

inspect local variables after errors. mxmeval: return evaluate(mxm(s)); Error
in: plotSegment(mxmeval("A"),mxmeval("C")); plotSegment(mxmeval("B ...
Hitung tegak lurus tengah di Maxima.
h = middlePerpendicular(A,B); g = middlePerpendicular(B,C);
Dan pusat lingkaran.
U = lineIntersection(h,g);
Kami mendapatkan rumus untuk jari-jari lingkaran.
assume(a 0,b 0,c 0); distance(U,B)|radcan
Mari kita tambahkan ini ke plot.
plotPoint(U()); ... plotCircle(circleWithCenter(mxmeval("U"),mxmeval("distance(U,C)"))):
Variable a2 not found! Use global variables or parameters for string evalua-
tion. Error in Errorinexpression : [a/2, (a2 + a12 - a * a1)/(2 * a2)]Errorin :
plotPoint(U());plotCircle(circleWithCenter(mxmeval("U"),mxmev...
Menggunakan geometri, kami memperoleh rumus sederhana
untuk radiusnya. Kami dapat memeriksa, apakah ini benar dengan Maxima.
Maxima akan memfaktorkan ini hanya jika kita kuadratkan.
c2/sin(computeAngle(A,B,C))2|factor
Contoh 4: Garis Euler dan Parabola
Garis Euler adalah garis yang ditentukan dari sembarang segitiga yang tidak
sama sisi. Ini adalah garis tengah segitiga, dan melewati beberapa titik penting
yang ditentukan dari segitiga, termasuk orthocenter, circumcenter, centroid,
titik Exeter dan pusat lingkaran sembilan titik segitiga.
Untuk demonstrasi, kami menghitung dan memplot garis Euler dalam se-
buah segitiga.
Pertama, kita mendefinisikan sudut-sudut segitiga di Euler. Kami menggu-
nakan definisi, yang terlihat dalam ekspresi simbolis.
A::=[-1,-1]; B::=[2,0]; C::=[1,2];
Untuk memplot objek geometris, kami menyiapkan area plot, dan menam-
bahkan titik ke sana. 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 segitiga.
plotSegment(A,B,""); plotSegment(B,C,""); plotSegment(C,A,"");
Berikut adalah luas segitiga, menggunakan rumus determinan. Tentu saja,
kita harus mengambil nilai absolut dari hasil ini.
areaTriangle(A,B,C)
Kita dapat menghitung koefisien sisi c.
c = lineThrough(A,B)
[- 1, 3, - 2]
Dan juga dapatkan rumus 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 meng-
hasilkan jarak positif ke garis.
getHesseForm(c,x,y,C),at(
Sekarang kita hitung lingkaran luar ABC.

```



```

LL = circleThrough(A,B,C); getCircleEquation(LL, x, y)
O = getCircleCenter(LL); O

```

Gambarkan lingkaran dan pusatnya. Cu dan U adalah simbolis. Kami mengevaluasi ekspresi ini untuk Euler.

```

plotCircle(LL()); plotPoint(O(), "O");

```

Kita dapat menghitung perpotongan ketinggian di ABC (orthocenter) secara numerik dengan perintah berikut.

```

H = lineIntersection(perpendicular(A,lineThrough(C,B)),... perpendicular(B,lineThrough(A,C))); H

```

Sekarang kita dapat menghitung garis Euler dari segitiga.

```

el = lineThrough(H,O); getLineEquation(el, x, y)

```

Tambahkan ke plot kami.

```

plotPoint(H(), "H"); plotLine(el(), "Garis Euler");

```

Pusat gravitasi harus berada di garis ini.

```

M = (A+B+C)/3; getLineEquation(el, x, y) with [x = M[1], y = M[2]]
plotPoint(M(), "M"); // titik berat

```

Teorinya memberitahu kita  $MH=2*MO$ . Kita perlu menyederhanakan dengan radcan untuk mencapai ini.

```

distance(M, H)/distance(M, O)|radcan

```

Fungsi termasuk fungsi untuk sudut juga.

```

computeAngle(A, C, B), degprint(
60°15'18.43"

```

Persamaan untuk pusat incircle 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 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 menjadi beberapa fungsi, tetapi pemecah default Maxima hanya dapat menemukan solusinya, jika kita kuadratkan persamaannya. Akibatnya, kami mendapatkan solusi palsu.

```

akar = solve(getHesseForm(lineThrough(A,B),x,y,C)2-distance([x,y],C)2, y)
[y = - 3 x - sqrt(70) sqrt(9 - 2 x) + 26, y = - 3 x + sqrt(70) sqrt(9 - 2 x) +
26]

```

Solusi pertama adalah

```

maxima: akar[1]

```

Menambahkan solusi pertama ke plot menunjukkan, bahwa itu memang jalan yang kita cari. Teorinya memberi tahu kita bahwa itu adalah parabola yang diputar.

```
plot2d(rhs(akar[1]),add=1):
function g(x) = rhs(akar[1]); 'g(x) = g(x)//fungsiyangmende finisikankurvadiatas
T = [-1, g(-1)]; // ambil sebarang titik pada kurva tersebut
dTC = distance(T,C); fullratsimp(dTC),float(
U = projectToLine(T,lineThrough(A,B)); U//proyeksiTpadagarisAB
dU2AB = distance(T,U); fullratsimp(dU2AB),float(
```

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 ceramah N.J.Wildberger. Dalam bukunya "Divine Proportions", Wildberger mengusulkan untuk mengganti pengertian klasik tentang jarak dan sudut dengan kuadrat dan penyebaran. Dengan menggunakan ini, memang mungkin untuk menghindari fungsi trigonometri dalam banyak contoh, dan tetap "rasional".

Berikut ini, saya memperkenalkan konsep, dan memecahkan beberapa masalah. Saya menggunakan perhitungan simbolik Maxima di sini, yang menyembunyikan keuntungan utama dari trigonometri rasional bahwa perhitungan hanya dapat dilakukan dengan kertas dan pensil. Anda diundang untuk memeriksa hasil tanpa komputer.

Intinya adalah bahwa perhitungan rasional simbolis sering kali menghasilkan hasil yang sederhana. Sebaliknya, trigonometri klasik menghasilkan hasil trigonometri yang rumit, yang hanya mengevaluasi perkiraan numerik.

```
load geometry;
```

Untuk pengenalan pertama, kami menggunakan segitiga persegi panjang dengan proporsi Mesir terkenal 3, 4 dan 5. Perintah berikut adalah perintah Euler untuk merencanakan geometri bidang yang terdapat dalam file Euler "geometry.e".

```
C:=[0,0]; A:=[4,0]; B:=[0,3]; ... setPlotRange(-1,5,-1,5); ... plot-
Point(A,"A"); plotPoint(B,"B"); plotPoint(C,"C"); ... plotSegment(B,A,"c");
plotSegment(A,C,"b"); plotSegment(C,B,"a"); ... insimg(30);
```

Tentu saja,

di mana  $\alpha$  adalah sudut di A. Cara yang biasa untuk menghitung sudut ini, adalah dengan mengambil invers dari fungsi sinus. Hasilnya adalah sudut yang tidak dapat dicerna, yang hanya dapat dicetak kira-kira.

```
wa := arcsin(3/5); degprint(wa)
36°52'11.63"
```

Trigonometri rasional mencoba menghindari hal ini.

Gagasan pertama trigonometri rasional adalah kuadran, yang menggantikan jarak. Sebenarnya, itu hanya jarak kuadrat. Berikut ini,  $a$ ,  $b$ , dan  $c$  menunjukkan kuadrat dari sisi-sisinya.

Teorema Pythagoras menjadi  $a+b=c$ .

```
a = 3^2; b = 4^2; c = 5^2; a + b = c
```

$$25 = 25$$

Pengertian kedua dari trigonometri rasional adalah penyebaran. Spread mengukur pembukaan antar baris. Ini adalah 0, jika garis-garisnya sejajar, dan 1, jika garis-garisnya persegi panjang. Ini adalah kuadrat sinus sudut antara dua garis.

Penyebaran garis AB dan AC pada gambar di atas didefinisikan sebagai:

di mana  $a$  dan  $c$  adalah kuadrat dari sembarang segitiga siku-siku dengan salah satu sudut di A.

$$sa = a/c; \quad sa$$

Ini lebih mudah dihitung daripada sudut, tentu saja. Tetapi Anda kehilangan properti bahwa sudut dapat ditambahkan dengan mudah.

Tentu saja, kita dapat mengonversi nilai perkiraan untuk sudut  $wa$  menjadi sprad, dan mencetaknya sebagai pecahan.

$$\text{fracprint}(\sin(wa)^2)$$

$$9/25$$

Hukum kosinus trigonometri klasik diterjemahkan menjadi "hukum silang" berikut.

Di sini  $a$ ,  $b$ , dan  $c$  adalah kuadrat dari sisi-sisi segitiga, dan  $sa$  adalah penyebaran sudut A. Sisi  $a$ , seperti biasa, berhadapan dengan sudut A.

Hukum ini diimplementasikan dalam file `geometri.e` yang kami muat ke Euler.

$$\text{crosslaw}(aa, bb, cc, saa)$$

Dalam kasus kami, kami mendapatkan

$$\text{crosslaw}(a, b, c, sa)$$

Mari kita gunakan `crosslaw` ini untuk mencari spread di A. Untuk melakukan ini, kita buat `crosslaw` untuk kuadran  $a$ ,  $b$ , dan  $c$ , dan selesaikan untuk spread yang tidak diketahui  $sa$ .

Anda dapat melakukannya dengan tangan dengan mudah, tetapi saya menggunakan Maxima. Tentu saja, kami mendapatkan hasilnya, kami sudah memilikinya.

$$\text{crosslaw}(a, b, c, x), \text{solve}(\text{crosslaw}(a, b, c, x), x)$$

Kita sudah tahu ini. Definisi spread adalah kasus khusus dari `crosslaw`.

Kita juga dapat menyelesaikan ini untuk umum  $a, b, c$ . Hasilnya adalah rumus yang menghitung penyebaran sudut segitiga yang diberikan kuadrat dari ketiga sisinya.

$$\text{solve}(\text{crosslaw}(aa, bb, cc, x), x)$$

Kita bisa membuat fungsi dari hasilnya. Fungsi seperti itu sudah didefinisikan dalam file `geometri.e` dari Euler.

$$\text{spread}(a, b, c)$$

Sebagai contoh, kita dapat menggunakannya untuk menghitung sudut segitiga dengan sisi

Hasilnya rasional, yang tidak begitu mudah didapat jika kita menggunakan trigonometri klasik.

$$\text{spread}(a, a, 4a/7)$$

Ini adalah sudut dalam derajat.

$$\text{degprint}(\arcsin(\sqrt{6/7}))$$

67°47'32.44"

Contoh lain

Sekarang, mari kita coba contoh yang lebih maju.

Kami mengatur tiga sudut segitiga sebagai berikut.

```
A:=[1,2]; B:=[4,3]; C:=[0,4]; ... setPlotRange(-1,5,1,7); ... plot-
Point(A,"A"); plotPoint(B,"B"); plotPoint(C,"C"); ... plotSegment(B,A,"c");
plotSegment(A,C,"b"); plotSegment(C,B,"a"); ... insimg;
```

Menggunakan Pythagoras, mudah untuk menghitung jarak antara dua titik. Saya pertama kali menggunakan jarak fungsi file Euler untuk geometri. Jarak fungsi menggunakan geometri klasik.

$distance(A, B)$

Euler juga mengandung fungsi untuk kuadran antara dua titik.

Dalam contoh berikut, karena  $c+b$  bukan  $a$ , maka segitiga itu bukan persegi panjang.

$c = quad(A, B); c, b = quad(A, C); b, a = quad(B, C); a,$

Pertama, mari kita hitung sudut tradisional. Fungsi `computeAngle` menggunakan metode biasa berdasarkan hasil kali titik dua vektor. Hasilnya adalah beberapa pendekatan floating point.

$wb = computeAngle(A, B, C); wb, (wb/pi180)()$

32.4711922908

Dengan menggunakan pensil dan kertas, kita dapat melakukan hal yang sama dengan hukum silang. Kami memasukkan kuadran  $a$ ,  $b$ , dan  $c$  ke dalam hukum silang dan menyelesaikan  $x$ .

$crosslaw(a, b, c, x), solve($

Yaitu, apa yang dilakukan oleh penyebaran fungsi yang didefinisikan dalam "geometry.e".

$sb = spread(b, a, c); sb$

Maxima mendapatkan hasil yang sama menggunakan trigonometri biasa, jika kita memaksanya. Itu menyelesaikan istilah  $\sin(\arccos(...))$  menjadi hasil pecahan. Sebagian besar siswa tidak dapat melakukan ini.

$\sin(computeAngle(A, B, C))^2$

Setelah kita memiliki `spread` di  $B$ , kita dapat menghitung tinggi  $h_a$  di sisi  $a$ . Ingat bahwa

Menurut definisi.

$ha = csb; ha$

Gambar berikut telah dihasilkan dengan program geometri C.a.R., yang dapat menggambar kuadrat dan menyebar.

image: (20) RationalGeometryCaR.png

Menurut definisi, panjang  $h_a$  adalah akar kuadrat dari kuadratnya.

$\sqrt{ha}$

Sekarang kita dapat menghitung luas segitiga. Jangan lupa, bahwa kita berhadapan dengan kuadrat!

$\sqrt{ha} \sqrt{a} / 2$

Rumus determinan 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 kita hitung spread di B untuk segitiga dengan sisi a, b, dan c. Kemudian kita menghitung luas kuadrat ("quadrea"?), faktorkan dengan Maxima, dan kita mendapatkan rumus Heron yang terkenal.

Memang, ini sulit dilakukan dengan pensil dan kertas.

```
spread(b2, c2, a2), factor(
```

Aturan Triple Spread

Kerugian dari spread adalah mereka tidak lagi hanya menambahkan sudut yang sama.

Namun, tiga spread dari sebuah segitiga memenuhi aturan "triple spread" berikut.

```
remvalue(sa,sb,sc); triplespread(sa, sb, sc)
```

Aturan ini berlaku untuk setiap tiga sudut yang menambah 180°.

Sejak menyebar

sama, aturan triple spread juga benar, jika

Karena penyebaran sudut negatif adalah sama, aturan penyebaran rangkap tiga juga berlaku, jika

Misalnya, kita dapat menghitung penyebaran sudut 60°. Ini 3/4. Persamaan memiliki solusi kedua, bagaimanapun, di mana semua spread adalah 0.

```
solve(triplespread(x, x, x), x)
```

Sebaran 90° jelas 1. Jika dua sudut dijumlahkan menjadi 90°, sebarannya menyelesaikan persamaan sebaran rangkap tiga dengan a,b,1. Dengan perhitungan berikut kita mendapatkan a+b=1.

```
triplespread(x, y, 1), solve(
```

Karena sebaran 180°-t sama dengan sebaran t, rumus sebaran rangkap tiga juga berlaku, jika satu sudut adalah jumlah atau selisih dua sudut lainnya.

Jadi kita dapat menemukan penyebaran sudut berlipat ganda. Perhatikan bahwa ada dua solusi lagi. Kami membuat ini fungsi.

```
solve(triplespread(a, a, x), x), functiondoublespread(a) = factor(rhs(
- 4 (a - 1) a
```

Pembagi Sudut

Ini situasinya, kita sudah tahu.

```
C:=[0,0]; A:=[4,0]; B:=[0,3]; ... setPlotRange(-1,5,-1,5); ... plot-
Point(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 umum a,b,c.

```
remvalue(a,b,c);
```

Jadi pertama-tama kita hitung penyebaran sudut yang dibagi dua di A, dengan menggunakan rumus sebaran rangkap tiga.

Masalah dengan rumus ini muncul lagi. Ini memiliki dua solusi. Kita harus memilih yang benar. Solusi lainnya mengacu pada sudut terbelah 180°-wa.

```
triplespread(x, x, a/(a + b)), solve(
```

Mari kita periksa persegi panjang Mesir.

```
sa2with[a = 32, b = 42]
```

Kami dapat mencetak sudut dalam Euler, setelah mentransfer penyebaran ke radian.

```
wa2 := arcsin(sqrt(1/10)); degprint(wa2)
18°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 dalam contoh spesifik kita.

```
computeAngle(C,A,P), computeAngle(P,A,B)
0.321750554397 0.321750554397
```

Sekarang kita hitung panjang garis bagi AP.

Kami menggunakan teorema sinus dalam segitiga APC. Teorema ini menyatakan bahwa

berlaku dalam segitiga apa pun. Kuadratkan, itu diterjemahkan ke dalam apa yang disebut "hukum penyebaran" di mana a,b,c menunjukkan quadrances.

Karena spread CPA adalah  $1-sa^2$ , kita dapatkan darinya  $bisa/1=b/(1-sa^2)$  dan dapat menghitung bisa (kuadran dari garis-bagi sudut).

```
factor(ratsimp(b/(1-sa2))); bisa =
```

Mari kita periksa rumus ini untuk nilai-nilai Mesir kita.

```
sqrt(mxmeval("at(bisa,[a=3^2,b=4^2])")), distance(A,P)
4.21637021356 4.21637021356
```

Kita juga dapat menghitung P menggunakan rumus spread.

```
py=factor(ratsimp(sa2bisa)); py
```

Nilainya sama dengan yang kita dapatkan dengan rumus trigonometri.

```
sqrt(mxmeval("at(py,[a=3^2,b=4^2])"))
1.33333333333
```

Sudut Akord

Perhatikan situasi berikut.

```
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"); plot-
Segment(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 rangkap tiga untuk sudut-sudut di pusat O untuk r. Jadi kita mendapatkan rumus untuk jari-jari kuadrat dari pericircle dalam hal kuadrat dari sisi.

Kali ini, Maxima menghasilkan beberapa nol kompleks, 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]);
```

*rabc*

Kita dapat menjadikannya sebagai 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);
```

Jari-jarinya memang 1.

periradius(a,b,c)

1

Faktanya, spread CBA hanya bergantung pada b dan c. Ini adalah teorema sudut chord.

$spread(b, a, c) = rabc/ratsimp$

Sebenarnya spreadnya adalah  $b/(4r)$ , dan kita melihat bahwa sudut chord dari chord b adalah setengah dari sudut pusat.

$doublespread(b/(4r)) = spread(b, r, r)|ratsimp$

Contoh 6: Jarak Minimal pada Bidang

Catatan awal

Fungsi yang, ke titik M di bidang, menetapkan jarak AM antara titik tetap A dan M, memiliki garis level yang agak sederhana: lingkaran berpusat di A.

remvalue();

A=[-1,-1];

function d1(x,y):=sqrt((x-A[1])<sup>2</sup> + (y - A[2])<sup>2</sup>)

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 agak sederhana: bagian atas kerucut:

plot3d("d1",xmin=-2,xmax=0,ymin=-2,ymax=0);

Tentu saja minimal 0 dicapai di A.

Dua poin

Sekarang kita lihat fungsi MA+MB dimana A dan B adalah dua titik (tetap).

Ini adalah "fakta yang diketahui" bahwa kurva level adalah elips, titik fokusnya adalah A dan B; kecuali untuk AB minimum yang konstan pada segmen [AB]:

B=[1,-1];

function d2(x,y):=d1(x,y)+sqrt((x-B[1])<sup>2</sup> + (y - B[2])<sup>2</sup>)

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 garis (AB) lebih terkenal:

plot2d("abs(x+1)+abs(x-1)",xmin=-3,xmax=3);

Tiga poin

Sekarang hal-hal yang kurang sederhana: Ini sedikit kurang terkenal bahwa MA+MB+MC mencapai minimum pada satu titik pesawat tetapi untuk menentukan itu kurang sederhana:

1) Jika salah satu sudut segitiga ABC lebih dari 120° (katakanlah di A), maka minimum dicapai pada titik ini (misalnya AB+AC).

Contoh:

C=[-4,1];

function d3(x,y):=d2(x,y)+sqrt((x-C[1])<sup>2</sup> + (y - C[2])<sup>2</sup>)

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  $120^\circ$ , minimumnya adalah pada titik F di bagian dalam segitiga, yang merupakan satu-satunya titik yang melihat sisi-sisi ABC dengan sudut yang sama (maka masing-masing  $120^\circ$ ):

```
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 mewujudkan gambar di atas dengan perangkat lunak geometri; misalnya, saya tahu soft yang ditulis di Jawa yang memiliki instruksi "garis kontur" ...

Semua ini di atas telah ditemukan oleh seorang hakim Perancis bernama Pierre de Fermat; dia menulis surat kepada dilettants lain seperti pendeta Marin Mersenne dan Blaise Pascal yang bekerja di pajak penghasilan. Jadi titik unik F sedemikian rupa sehingga  $FA+FB+FC$  minimal, disebut titik Fermat segitiga. Tetapi tampaknya beberapa tahun sebelumnya, Torricelli Italia telah menemukan titik ini sebelum Fermat melakukannya! Bagaimanapun tradisinya adalah mencatat poin ini F...

Empat poin

Langkah selanjutnya adalah menambahkan 4 titik D dan mencoba meminimalkan  $MA+MB+MC+MD$ ; katakan bahwa Anda adalah operator TV kabel dan ingin mencari 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 minimum dan tidak tercapai di salah satu simpul A, B, C atau 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...

Sekarang ABCD adalah persegi, kami berharap bahwa titik optimal akan menjadi pusat 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 pvcengine.exe di jalur program.
Pertama kita hitung jari-jari bola.
Jika Anda melihat gambar di bawah, Anda melihat bahwa kita membu-
tuhkan dua lingkaran yang menyentuh dua garis yang membentuk kerucut, dan
satu garis yang membentuk bidang yang memotong kerucut.
Kami menggunakan file geometri.e dari Euler untuk 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 saya 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 ke dalam gambar.
color(red);
plotCircle(circleWithCenter([0,u[1]],dd[1]),"");
plotCircle(circleWithCenter([0,u[2]],dd[2]),"");
insimg;
Plot dengan Povray

```

Selanjutnya kami merencanakan semuanya dengan Povray. Perhatikan bahwa Anda mengubah perintah apa pun dalam urutan perintah Povray berikut, dan menjalankan 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:\Files-Ray\3.7.exe

Kami mengatur adegan dengan tepat.

povstart(zoom=11,center=[0,0,0.5],height=10°,angle=140°);

Selanjutnya kita menulis dua bidang ke file Povray.

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 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]],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)));

Kemudian kami menghasilkan dua titik di mana bola menyentuh bidang. Ini adalah fokus dari 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 hitung 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 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 kemudian.
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)));
 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);
 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)));
endfunction ;/prej. Anda membutuhkan kaca mata merah/sian untuk menghar-
gai efek berikut.
 povanaglyph("scene",zoom=11,center=[0,0,0.5],height=10°,angle=140°);

```

#### Contoh 8: Geometri Bumi

Dalam buku catatan ini, kami ingin melakukan beberapa perhitungan sferis. Fungsi-fungsi tersebut terdapat dalam file "spherical.e" di folder contoh. Kita perlu memuat file itu terlebih dahulu.

```
load "spherical.e";
```

Untuk memasukkan posisi geografis, kami menggunakan vektor dengan dua koordinat dalam radian (utara dan timur, nilai negatif untuk selatan dan barat). Berikut koordinat 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 spherical).

```
sposprint(FMIPA) // posisi garis lintang dan garis bujur FMIPA UNY
S 7°46.467' E 110°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°34.333' E 110°49.683' S 6°59.050' E 110°24.533'
```

Pertama kita menghitung vektor dari satu ke yang lain pada bola ideal. Vektor ini [pos,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°)- km // perkiraan
 jarak FMIPA-Solo
 65°20'26.60" 53.8945384608
 Ini adalah perkiraan yang baik. Rutinitas berikut menggunakan perkiraan
 yang lebih baik. Pada jarak yang begitu pendek hasilnya hampir sama.
 esdist(FMIPA,Semarang)- " km", // perkiraan jarak FMIPA-Semarang
 88.0114026318 km
 Ada fungsi untuk heading, 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);
 degprint(asum)
 180°0'10.77"
 Ini dapat digunakan untuk menghitung luas segitiga. Catatan: Untuk segitiga
 kecil, ini tidak akurat karena kesalahan pengurangan dalam asum-pi.
 (asum-pi)rearth(48°)2 - "km2", // perkiraan luas segitiga FMIPA – Solo – Semarang
 2116.02948749 km2
 Ada fungsi untuk ini, yang menggunakan garis lintang rata-rata segitiga
 untuk menghitung jari-jari bumi, dan menangani kesalahan pembulatan untuk
 segitiga yang sangat kecil.
 esarea(Solo,FMIPA,Semarang)- " km2", // perkiraan yang sama dengan fungsi esarea()
 2123.64310526 km2
 Kita juga dapat menambahkan vektor ke posisi. Sebuah vektor berisi heading
 dan jarak, keduanya dalam radian. Untuk mendapatkan vektor, kami menggunakan
 vektor. Untuk menambahkan vektor ke posisi, kami menggunakan
 vektor sadd.
 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(FMIPA,esdir(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
 pendekatan yang baik.
 esdist(Tugu,Monas)- " km", // perkiraan jarak Tugu Jogja - Monas Jakarta
 431.565659488 km
 Judulnya sama dengan judul yang dihitung di Google Earth.
 degprint(esdir(Tugu,Monas))
 294°17'2.85"

```

Namun, kita tidak lagi mendapatkan posisi target yang tepat, jika kita menambahkan heading dan jarak ke posisi semula. Hal ini terjadi, karena kita tidak menghitung fungsi invers secara tepat, tetapi mengambil perkiraan jari-jari bumi di sepanjang jalan.

```
sposprint(esadd(Tugu,esdir(Tugu,Monas),esdist(Tugu,Monas)))
```

```
S 6°10.500' E 106°48.717'
```

Namun, kesalahannya tidak besar.

```
sposprint(Monas),
```

```
S 6°10.500' E 106°48.717'
```

Tentu kita tidak bisa berlayar dengan tujuan yang sama dari satu tujuan ke tujuan lainnya, jika kita ingin menempuh jalur terpendek. Bayangkan, Anda terbang NE mulai dari titik mana pun di bumi. Kemudian Anda akan berputar ke kutub utara. Lingkaran besar tidak mengikuti heading yang konstan!

Perhitungan berikut menunjukkan bahwa kami jauh dari tujuan yang benar, jika kami menggunakan pos yang sama selama perjalanan kami.

```
dist=esdist(Tugu,Monas); hd=esdir(Tugu,Monas);
```

Sekarang kita tambahkan 10 kali sepersepuluh dari jarak, menggunakan pos ke Monas, kita sampai di Tugu.

```
p=Tugu; loop 1 to 10; p=esadd(p,hd,dist/10); end;
```

Hasilnya jauh.

```
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 garis lintang 30°, melainkan jalur terpendek yang dimulai 10° lebih jauh ke utara di P1.

```
sdegprint(esdir(P1,P2))
```

```
79.69°
```

Tapi, jika kita mengikuti pembacaan kompas ini, kita akan berputar ke kutub utara! Jadi kita harus menyesuaikan arah kita di sepanjang jalan. Untuk tujuan kasar, kami menyesuaikannya pada 1/10 dari total jarak.

```
p=P1; dist=esdist(P1,P2); ... loop 1 to 10; dir=esdir(p,P2); sdegprint(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 heading yang sama terlalu lama.

```
skmprint(esdist(p,P2))
```

```
0.203km
```

Kami mendapatkan perkiraan yang baik, jika kami menyesuaikan pos setelah 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 lingkaran besar menuju Monas dengan fungsi navigasi.

```
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 i/prej. Sekarang rencanakan 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/sian.

```
plot3d("testplot",angle=25,height=6,distance=5,own=1,anaglyph=1,zoom=4):
```

#### MENCoba RUMUS-RUMUS PADA MATERI DI ATAS

##### Geometri Simbolik

```
A := [2,0]; B := [0,2]; C := [3,3]; // menentukan tiga titik A, B, C
```

```
c := lineThrough(B,C) // c=BC
```

```
[-1, 3, 6]
```

```
getLineEquation(c,x,y),solve(
```

```
h = perpendicular(A,lineThrough(B,C)) // h melalui A tegak lurus BC
```

```
[3, 1, 6]
```

```
Q = lineIntersection(c,h) // Q titik potong garis c=BC dan h
```

```
6 12 [-, -] 5 5
```

```
projectToLine(A,lineThrough(B,C))//proyeksiApadaBC
```

```
distance(A,Q)//jarakAQ
```

```
cc = circleThrough(A,B,C); cc//(titikpusatdanjari-jari)lingkaranmelaluiA,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//titikpotong2
```

Garis dan Lingkaran yang berpotongan

```
A := [2,0]; c = circleWithCenter(A,4);
```

```
B := [2,3]; C := [3,2]; l = lineThrough(B,C);
```

```
setPlotRange(5); plotCircle(c); plotLine(l);
```

```
P1,P2,f=lineCircleIntersections(l,c);
```

```
P1, P2,
```

```
[5.89792, -0.897916] [1.10208, 3.89792]
```

```
plotPoint(P1); plotPoint(P2):
```

```
c = circleWithCenter(A,4) // lingkaran dengan pusat A jari-jari 4
```

```
[2, 0, 4]
```

```
l = lineThrough(B,C) // garis l melalui B dan C
```

```

[1, 1, 5]
lineCircleIntersections(l,c)|radcan, //titikpotonglingkarancdangarisl
C=A+normalize([-3,-4])4; plotPoint(C); plotSegment(P1,C); plotSegment(P2,C);
degprint(computeAngle(P1,C,P2))
57°58'20.06"
C=A+normalize([-4,-5])4; plotPoint(C); plotSegment(P1,C); plotSegment(P2,C);
degprint(computeAngle(P1,C,P2))
57°58'20.06"
insimg;
Garis Sumbu
A :=[3,3]; B :=[-2,-3];
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):
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];
g = getLineEquation(lineThrough(P1,P2),x,y);
solve(g,y)
solve(getLineEquation(middlePerpendicular(A,B),x,y),y)
h =getLineEquation(lineThrough(A,B),x,y);
solve(h,y)
Garis Euler dan Parabola
A :=[-1.5,-1.5]; B :=[3,0]; C :=[1.5,3];
setPlotRange(3); plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C");
plotSegment(A,B,""); plotSegment(B,C,""); plotSegment(C,A,"");
areaTriangle(A,B,C)
c = lineThrough(A,B)
3 9 9 [- -, - -] 2 2 2
getLineEquation(c,x,y)
getHesseForm(c,x,y,C),at(
LL = circleThrough(A,B,C); getCircleEquation(LL,x,y)
O = getCircleCenter(LL); O
plotCircle(LL()); plotPoint(O(),"O");
H = lineIntersection(perpendicular(A,lineThrough(C,B)),... perpendicular(
lar(B,lineThrough(A,C))); H
el = lineThrough(H,O); getLineEquation(el,x,y)
plotPoint(H(),"H"); plotLine(el(),"Garis Euler");
M = (A+B+C)/3; getLineEquation(el,x,y)with[x = M[1],y = M[2]]
plotPoint(M(),"M"); // titik berat
distance(M,H)/distance(M,O)|radcan
computeAngle(A,C,B),degprint(

```

```

60°15'18.43"
Q = lineIntersection(angleBisector(A,C,B),angleBisector(C,B,A))—radcan;
Q
r = distance(Q,projectToLine(Q,lineThrough(A,B)))—ratsimp; r
LD = circleWithCenter(Q,r); // Lingkaran dalam
color(5); plotCircle(LD());
contoh lain dari materi trigonometri rasional
A :=[2,3]; B :=[5,4]; C :=[0,5]; ... setPlotRange(-1,5,1,7); ... plot-
Point(A,"A"); plotPoint(B,"B"); plotPoint(C,"C"); ... plotSegment(B,A,"c");
plotSegment(A,C,"b"); plotSegment(C,B,"a"); ... insimg;
distance(A,B)
c = quad(A,B); c, b = quad(A,C); b, a = quad(B,C); a,
wb = computeAngle(A,B,C); wb,(wb/pi180)()
29.7448812969
crosslaw(a,b,c,x),solve(
sb = spread(b,a,c); sb
sin(computeAngle(A,B,C))2
ha = csb; ha
sqrt(ha)
sqrt(ha)sqrt(a)/2
areaTriangle(B,A,C)
Aturan penyebaran 3 kali lipat
setPlotRange(1); ... 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"); plot-
Segment(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;
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]);
rabc
function periradius(a,b,c) = rabc;
a:=quadrance(B,C); b:=quadrance(A,C); c:=quadrance(A,B);
periradius(a,b,c)
1
spread(b,a,c)rabc|ratsimp
doublespread(b/(4r)) - spread(b,r,r)|ratsimp
Contoh 6: Jarak Minimal pada Bidang
Catatan awal
Fungsi yang, ke titik M di bidang, menetapkan jarak AM antara titik tetap
A dan M, memiliki garis level yang agak sederhana: lingkaran berpusat di A.
remvalue();
A=[-2,-2];
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 agak sederhana: bagian atas kerucut:

```



```

plot3d("d1",xmin=-2,xmax=0,ymin=-2,ymax=0):
Ternyata setelah mencoba yang bisa hanya dengan memasukkan angka 1,
karena ketika memakai angka 2, plot tidak membentuk kerucut diatas.
Dua poin
B=[2,-2];
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 garis (AB) lebih terkenal:
plot2d("abs(x+1)+abs(x-1)",xmin=-3,xmax=3):
Tiga poin
Contoh:
C=[-3,2];
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 mini-
mum is on A");
P=(A_B_C_A)'; plot2d(P[1],P[2],add=1,color=12);
insimg;
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 (maka masing-masing 120 °):
C=[-1,2];
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;
Empat poin
Langkah selanjutnya adalah menambahkan 4 titik D dan mencoba memi-
nimalkan MA+MB+MC+MD; katakan bahwa Anda adalah operator TV ka-
bel dan ingin mencari di bidang mana Anda harus meletakkan antena sehingga
Anda dapat memberi makan empat desa dan menggunakan panjang kabel sesedikit
mungkin!
D=[2,21];
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;
Contoh 7: Bola Dandelin dengan Povray
load geometry;
Pertama dua garis yang membentuk kerucut.
g1 = lineThrough([0,0],[2,a])

```

```

[- a, 2, 0]
g2 = lineThrough([0,0],[-2,a])
[- a, - 2, 0]
g = lineThrough([-2,0],[2,2])
[- 2, 4, 4]
setPlotRange(-2,2,0,3);
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.
u := sol()
[0.558482, 4.77485]
dd := d()
[0.394906, 3.37633]
Plot lingkaran ke dalam gambar.
color(red);
plotCircle(circleWithCenter([0,u[1]],dd[1]), "");
plotCircle(circleWithCenter([0,u[2]],dd[2]), "");
insimg;
Latihan
1. Gambarkan 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(-4,4,-4,4);
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  $y = ax^2 + bx + c$ .
- Substitusikan 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,16a+4b=-c,c=-4],[a,b,c])
[[a = - 1, b = 5, c = - 4]]

```

Sehingga didapatkan nilai  $a = -1$ ,  $b = 5$  dan  $c = -4$

function  $y = -x^2 + 5x - 4$

$2 - x + 5x - 4$

```

plot2d("-x^2 + 5x - 4", -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.

$AB = \text{norm}(A-B)$  //panjang sisi AB

6

$CD = \text{norm}(C-D)$  //panjang sisi CD

6

$AD = \text{norm}(A-D)$  //panjang sisi AD

6

$BC = \text{norm}(B-C)$  //panjang sisi BC

6

$AB \cdot CD$

36

$AD \cdot BC$

36

Terbukti bahwa hasil kali panjang sisi-sisi yang berhadapan sama yaitu 36. Jadi dapat dipastikan bahwa segiempat tersebut merupakan segiempat garis singgung.

4. Gambarkanlah 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])<sup>2</sup> + (y - P[2])<sup>2</sup>)

$Q = [1,-1]$ ; function d2(x,y):=sqrt((x-P[1])<sup>2</sup> + (y - P[2])<sup>2</sup>) + sqrt((x-Q[1])<sup>2</sup> + (y - Q[2])<sup>2</sup>)

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. Gambarkanlah 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):
plot2d("abs(x+1)+abs(x-1)",xmin=-3,xmax=3):
```

## 7 Statistika

EMT untuk Statistika Nama : Emlys Jenny Ester Sirait

NIM : 23030630081

Kelas : Matematika B

Dalam buku catatan ini, kami mendemonstrasikan plot statistik utama, tes, dan distribusi dalam Euler.

Mari kita mulai dengan beberapa statistik deskriptif. Ini bukanlah sebuah pengantar statistik. Jadi, Anda mungkin memerlukan latar belakang untuk memahami detailnya.

Asumsikan pengukuran berikut ini. Kita ingin menghitung nilai rata-rata dan deviasi standar yang diukur.

```
M=[1000,1004,998,997,1002,1001,998,1004,998,997]; ... median(M), mean(M),
dev(M),
999 999.9 2.72641400622
```

Kita dapat memplot plot kotak dan kumis untuk data tersebut. Dalam kasus kami, tidak ada pencilan.

```
aspect(1.75); boxplot(M):
```

Kami menghitung probabilitas bahwa suatu nilai lebih besar dari 1005, dengan mengasumsikan nilai yang diukur dari distribusi normal.

Semua fungsi untuk distribusi dalam Euler diakhiri dengan ...dis dan menghitung distribusi probabilitas kumulatif (CPF).

Kami mencetak hasilnya dalam fungsi cetak.

```
print((1-normaldis(1005,mean(M),dev(M)))100,2,unit="
3.07
```

Untuk contoh berikutnya, kami mengasumsikan jumlah pria berikut ini dalam rentang ukuran tertentu.

```
r=155.5:4:187.5; v=[22,71,136,169,139,71,32,8];
```

Berikut ini adalah plot distribusinya.

```
plot2d(r,v,a=150,b=200,c=0,d=190,bar=1,style="
/"):

```

Kita dapat memasukkan data mentah tersebut ke dalam tabel.

Tabel adalah sebuah metode untuk menyimpan data statistik. Tabel kita harus berisi tiga kolom: Awal rentang, akhir rentang, jumlah orang dalam rentang.

Tabel dapat dicetak dengan header. Kami menggunakan vektor string untuk mengatur header.

```
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
```

Jika kita membutuhkan nilai rata-rata dan statistik lain dari ukuran, kita perlu menghitung titik tengah rentang. Kita dapat menggunakan dua kolom pertama dari tabel kita untuk hal ini.

Symbol “—” digunakan untuk memisahkan kolom, fungsi “writetable” digunakan untuk menulis tabel, dengan opsi “labc” untuk menentukan judul kolom.

```
(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
```

Tetapi akan lebih mudah, untuk melipat rentang dengan vektor  $[1/2, 1/2]$ .

```
M=fold(r,[0.5,0.5])
```

```
[157.5, 161.5, 165.5, 169.5, 173.5, 177.5, 181.5, 185.5]
```

Sekarang kita dapat menghitung rata-rata dan deviasi sampel dengan frekuensi yang diberikan.

```
m,d=meandev(M,v); m, d,
```

```
169.901234568 5.98912964449
```

Mari kita tambahkan distribusi normal dari nilai-nilai tersebut ke dalam diagram batang di atas. Rumus untuk distribusi normal dengan rata-rata  $m$  dan deviasi standar  $d$  adalah:

Karena nilainya antara 0 dan 1, untuk memplotnya pada diagram batang, nilai tersebut harus dikalikan dengan 4 kali jumlah data.

```
plot2d("qnormal(x,m,d)sum(v)4", ... xmin=min(r),xmax=max(r),thickness=3,add=1):
```

Tabel

Dalam direktori buku catatan ini, Anda dapat menemukan file dengan tabel. Data tersebut mewakili hasil survei. Berikut adalah empat baris pertama dari file tersebut. Data berasal dari sebuah buku online berbahasa Jerman “Einführung in die Statistik mit R” oleh 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
```

Tabel berisi 7 kolom angka atau token (string). Kita ingin membaca tabel tersebut dari file. Pertama, kita menggunakan terjemahan kita sendiri untuk token-token tersebut.

Untuk itu, kita mendefinisikan set token. Fungsi `strtokens()` mendapatkan vektor string token dari string yang diberikan.

```
mf=["m","f"]; yn=["y","n"]; ev:=strtokens("g vg m b vb");
```

Sekarang kita membaca tabel dengan terjemahan ini.

Argumen tok2, tok4, dan lain-lain adalah terjemahan dari kolom-kolom tabel. Argumen-argumen ini tidak ada dalam daftar parameter readtable(), jadi Anda perlu memberikannya dengan “:=”.

```
MT,hd=readtable("table.dat",tok2:=mf,tok4:=yn,tok5:=ev,tok7:=yn);
load over statistics;
```

Untuk mencetak, kita perlu menentukan set token yang sama. Kami mencetak empat baris pertama saja.

```
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);
Fungsi readtable() sekarang mengembalikan satu set token.
tok
m n f y g vg
```

Tabel berisi entri dari file dengan token yang diterjemahkan menjadi angka. String khusus NA="." ditafsirkan sebagai "Tidak Tersedia", dan mendapatkan NAN (bukan angka) dalam tabel. Terjemahan ini dapat diubah dengan parameter NA, dan NAvail.

```
MT[1]
[1, 1, 30, 2, NAN, 1.8, 2]
Berikut ini adalah isi tabel dengan angka yang tidak diterjemahkan.
```

```
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
```

Untuk kenyamanan, Anda dapat menaruh output dari readtable() ke dalam sebuah daftar.

```
Table=readtable("table.dat",ctok=ctok);
```

Dengan menggunakan kolom token yang sama dan token yang dibaca dari file, kita dapat mencetak tabel. Kita dapat menentukan ctok, tok, dll. atau menggunakan daftar Tabel.

```
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
```





Dengan `getstatistics()`, kita juga dapat menghubungkan hitungan dalam dua kolom tabel satu sama lain.

```
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
```

Tabel dapat ditulis ke sebuah file.

```
filename="test.dat"; ... writetable(count,labr=tok[xu1],labc=tok[xu2],file=filename);
```

Kemudian kita dapat membaca tabel dari file tersebut.

```
MT2,hd,tok2,hdr=readtable(filename, clabs, rlabs); ... writetable(MT2,labr=hdr,labc=hd)
n y m 7 5 f 1 12
```

Dan menghapus file tersebut.

```
fileremove(filename);
```

Distribusi

Dengan `plot2d`, ada metode yang sangat mudah untuk memplot distribusi data eksperimen.

```
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
```

Perhatikan perbedaan antara plot batang (sampel) dan kurva normal (distribusi sesungguhnya). Masukkan kembali ketiga perintah tersebut untuk melihat hasil pengambilan sampel yang lain.

Berikut ini adalah perbandingan 10 simulasi dari 1000 nilai terdistribusi normal dengan menggunakan apa yang disebut plot kotak. Plot ini menunjukkan median, kuartil 25maksimal, serta pencilan.

```
p=normal(10,1000); boxplot(p):
```

Untuk menghasilkan bilangan bulat acak, Euler memiliki `inrandom`. Mari kita simulasikan pelemparan dadu dan memplot distribusinya.

Kita menggunakan fungsi `getmultiplicities(v,x)`, yang menghitung seberapa sering elemen-elemen dari `v` muncul di dalam `x`. Kemudian kita memplot hasilnya menggunakan `columnplot`.

```
k=inrandom(1,6000,6); ... columnplot(getmultiplicities(1:6,k)); ...
ygrid(1000,color=red):
```

Meskipun `inrandom(n,m,k)` menghasilkan bilangan bulat yang terdistribusi secara seragam dari 1 sampai `k`, adalah mungkin untuk menggunakan distribusi bilangan bulat yang lain dengan `randpint()`.

Pada contoh berikut, probabilitas untuk 1,2,3 adalah 0.4, 0.1, 0.5 secara berurutan.

```
randpint(1,1000,[0.4,0.1,0.5]); getmultiplicities(1:3,
[378, 102, 520])
```

Euler dapat menghasilkan nilai acak dari lebih banyak distribusi. Lihatlah ke dalam referensi.

Misalnya, kita mencoba distribusi eksponensial. Sebuah variabel acak kontinu `X` dikatakan memiliki distribusi eksponensial, jika PDF-nya diberikan oleh dengan parameter

```

plot2d(randexponential(1,1000,2), distribution):
Untuk banyak distribusi, Euler dapat menghitung fungsi distribusi dan ke-
balikannya.
plot2d("normaldis",-4,4):
Berikut ini adalah salah satu cara untuk memplot kuantil.
plot2d("qnormal(x,1,1.5)",-4,6); ... plot2d("qnormal(x,1,1.5)",a=2,b=5, add, filled):
Probabilitas untuk berada di area hijau adalah sebagai berikut.
normaldis(5,1,1.5)-normaldis(2,1,1.5)
0.248662156979
Hal ini dapat dihitung secara numerik dengan integral berikut ini.
gauss("qnormal(x,1,1.5)",2,5)
0.248662156979
Mari kita bandingkan distribusi binomial dengan distribusi normal dengan
rata-rata dan deviasi yang sama. Fungsi invbindis() menyelesaikan interpolasi
linier antara nilai bilangan bulat.
invbindis(0.95,1000,0.5), invnormaldis(0.95,500,0.5sqrt(1000))
525.516721219 526.007419394
Fungsi qdis() adalah densitas dari distribusi chi-square. Seperti biasa, Euler
memetakan vektor ke fungsi ini. Dengan demikian kita mendapatkan plot semua
distribusi chi-kuadrat dengan derajat 5 hingga 30 dengan mudah dengan cara
berikut.
plot2d("qchidis(x,(5:5:50))",0,50):
Euler memiliki fungsi-fungsi yang akurat untuk mengevaluasi distribusi-
distribusi. Mari kita periksa chidis() dengan sebuah integral.
Penamaannya diusahakan untuk konsisten. Sebagai contoh,
* distribusi chi-kuadrat adalah chidis(),
* fungsi kebalikannya adalah invchidis(),
* densitasnya adalah qchidis().
Pelengkap dari distribusi (ekor atas) adalah chicdis().
chidis(1.5,2), integrate("qchidis(x,2)",0,1.5)
0.527633447259 0.527633447259
Distribusi Diskrit
Untuk menentukan distribusi diskrit Anda sendiri, Anda dapat menggu-
nakan metode berikut.
Pertama, kita tetapkan fungsi distribusinya.
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]
Artinya, dengan probabilitas wd[i+1]-wd[i] kita menghasilkan nilai acak i.
Ini hampir merupakan distribusi yang seragam. Mari kita definisikan sebuah
generator bilangan acak untuk ini. Fungsi find(v,x) menemukan nilai x dalam
vektor v. Fungsi ini juga dapat digunakan untuk vektor x.
function wrongdice (n,m) := find(wd,random(n,m))
Kesalahan ini sangat halus sehingga kita hanya bisa melihatnya setelah
melakukan iterasi yang sangat banyak.
columnplot(getmultiplicities(1:6,wrongdice(1,1000000))):

```

Berikut ini adalah fungsi sederhana untuk memeriksa distribusi seragam dari nilai 1... K dalam v. Kami menerima hasilnya, jika untuk semua frekuensi

```
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 i/prej
```

Memang fungsi ini menolak distribusi seragam.

```
checkrandom(wrongdice(1,1000000))
0
```

Dan ini menerima generator acak bawaan.

```
checkrandom(intrand(1,1000000,6))
1
```

Kita dapat menghitung distribusi binomial. Pertama, ada `binomials()`, yang mengembalikan probabilitas i atau kurang dari n percobaan.

```
bindis(410,1000,0.4)
0.751401349654
```

Fungsi Beta invers digunakan untuk menghitung interval kepercayaan Clopper-Pearson untuk parameter p. Tingkat defaultnya adalah alpha.

Arti dari interval ini adalah jika p berada di luar interval, hasil yang diamati sebesar 410 dalam 1000 jarang terjadi.

```
clopperpearson(410,1000)
[0.37932, 0.441212]
```

Perintah berikut ini adalah cara langsung untuk mendapatkan hasil di atas. Tetapi untuk n yang besar, penjumlahan langsung tidak akurat dan lambat.

```
p=0.4; i=0:410; n=1000; sum(bin(n,i)p^i(1-p)^(n-i))
0.751401349655
```

Omong-omong, `invbinsum()` menghitung kebalikan dari `binomials()`.

```
invbindis(0.75,1000,0.4)
409.932733047
```

Dalam Bridge, kita mengasumsikan 5 kartu yang terbuka (dari 52 kartu) di dua tangan (26 kartu). Mari kita hitung probabilitas distribusi yang lebih buruk dari 3:2 (misalnya 0:5, 1:4, 4:1, atau 5:0).

```
2hypergeomsum(1,5,13,26)
0.321739130435
```

Ada juga simulasi distribusi multinomial.

```
randmultinomial(10,1000,[0.4,0.1,0.5])
381 100 519 376 91 533 417 80 503 440 94 466 406 112 482 408 94 498 395
107 498 399 96 505 428 87 485 400 99 501
```

Memplot Data

Untuk memplot data, kami mencoba hasil pemilihan umum Jerman sejak tahun 1990, yang diukur dalam kursi.

```
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];
```

Untuk pesta, kami menggunakan serangkaian nama.

```
P:=["CDU/CSU","SPD","FDP","Gr","Li"];
```

Mari kita cetak persentasenya dengan baik.

Pertama kita ekstrak kolom-kolom yang diperlukan. Kolom 3 sampai 7 adalah kursi masing-masing partai, dan kolom 2 adalah jumlah total kursi. kolom adalah tahun pemilihan.

```
BT:=BW[3:7]; BT:=BT/sum(BT); YT:=BW[1]';
```

Kemudian kita mencetak statistik dalam bentuk tabel. Kita menggunakan nama sebagai judul kolom, dan tahun sebagai judul baris. Lebar default untuk kolom adalah  $wc = 10$ , tetapi kami lebih suka output yang lebih padat. Kolom-kolom akan diperluas untuk label-label kolom, jika perlu.

```
writetable(BT100,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
```

Perkalian matriks berikut ini mengekstrak jumlah persentase dua partai besar yang menunjukkan bahwa partai-partai kecil telah memperoleh suara di parlemen hingga tahun 2009.

```
BT1:=(BT.[1;1;0;0;0])'100
```

```
[84.29, 81.25, 81.1659, 82.7529, 72.9642, 61.8971, 79.8732]
```

Ada juga plot statistik sederhana. Kita menggunakannya untuk menampilkan garis dan titik secara bersamaan. Alternatif lainnya adalah memanggil `plot2d` dua kali dengan `gt;add`.

```
statplot(YT,BT1,"b"):
```

Tentukan beberapa warna untuk masing-masing pihak.

```
CP:=[rgb(0.5,0.5,0.5),red,yellow,green,rgb(0.8,0,0)];
```

Sekarang kita dapat memplot hasil pemilu 2009 dan perubahannya ke dalam satu plot menggunakan `figure`. Kita dapat menambahkan vektor kolom pada setiap 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):
```

Plot data menggabungkan baris data statistik dalam satu plot.

```
J:=BW[1]'; DP:=BW[3:7]'; ... dataplot(YT,BT',color=CP); ... label-box(P,color=CP,styles="[]", points,w=0.2,x=0.3,y=0.4):
```

Plot kolom 3D menunjukkan deretan data statistik dalam bentuk kolom. Kami menyediakan label untuk baris dan kolom. `angle` adalah sudut pandang.

```
columnsplot3d(BT,scols=P,srows=YT, ... angle=30°,ccols=CP):
```

Representasi lainnya adalah plot mosaik. Perhatikan bahwa kolom-kolom pada plot mewakili kolom-kolom pada matriks di sini. Karena panjangnya label CDU/CSU, kita mengambil jendela yang lebih kecil dari biasanya.

```
shrinkwindow(smaller); ... mosaicplot(BT',srows=YT,scols=P,color=CP,style="");
... shrinkwindow():
```

Kita juga bisa membuat diagram lingkaran. Karena warna hitam dan kuning membentuk koalisi, kita menyusun ulang elemen-elemennya.

```
i=[1,3,5,4,2]; piechart(BW[6,3:7][i],color=CP[i],lab=P[i]):
```

Berikut ini jenis plot yang lain.

```
starplot(normal(1,10)+4,lab=1:10, rays):
```

Beberapa plot di `plot2d` bagus untuk statika. Berikut ini adalah plot impuls dari data acak, yang terdistribusi secara seragam dalam  $[0,1]$ .

```
plot2d(makeimpulse(1:10,random(1,10)), bar):
```

Tetapi untuk data yang terdistribusi secara eksponensial, kita mungkin memerlukan plot logaritmik.

```
logimpulseplot(1:10,-log(random(1,10))10):
```

Fungsi `columnsplot()` lebih mudah digunakan, karena hanya membutuhkan sebuah vektor nilai. Selain itu, fungsi ini dapat mengatur labelnya menjadi apa pun yang kita inginkan, kita telah mendemonstrasikan hal ini dalam tutorial ini.

Berikut ini adalah aplikasi lain, di mana kita menghitung karakter dalam sebuah kalimat dan memplot statistik.

```
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):
```

Anda juga dapat menetapkan sumbu secara manual.

```
n=10; p=0.4; i=0:n; x=bin(n,i)pi(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(["Binomialdistribution","withp =
0.4"]):
```

Berikut ini adalah cara untuk memplot frekuensi angka dalam vektor.

Kami membuat vektor angka acak bilangan bulat 1 hingga 6.

```
v=intrandom(1,10,10)
```

```
[8, 5, 8, 8, 6, 8, 8, 3, 5, 5]
```

Kemudian ekstrak nomor unik dalam `v`.

```
vu=unique(v)
```

```
[3, 5, 6, 8]
```

Dan memplot frekuensi dalam plot kolom.

```
columnsplot(getmultiplicities(vu,v),lab=vu,style=" /"):
```

Kami ingin mendemonstrasikan fungsi untuk distribusi nilai empiris.

```
x=normal(1,20);
```

Fungsi `empdist(x,vs)` membutuhkan larik nilai yang telah diurutkan. Jadi kita harus mengurutkan `x` sebelum dapat menggunakannya.

```
xs=sort(x);
```

Kemudian kita memplot distribusi empiris dan beberapa batang kepadatan ke dalam satu plot. Alih-alih plot batang untuk distribusi, kali ini kami menggunakan plot gigi gergaji.

```
figure(2,1); ... figure(1); plot2d("empdist",-4,4,xs); ... figure(2); plot2d(histo(x,v=-
4:0.2:4,ibar)); ... figure(0):
```

Plot sebaran mudah dilakukan di Euler dengan plot titik biasa. Grafik berikut ini menunjukkan bahwa  $X$  dan  $X+Y$  berkorelasi positif secara jelas.

```
x=normal(1,100); plot2d(x,x+rotright(x), points,style=".."):
```

Sering kali, kita ingin membandingkan dua sampel dari distribusi yang berbeda.

Hal ini dapat dilakukan dengan plot kuantil-kuantil.

Untuk pengujian, kami mencoba distribusi student-t dan distribusi eksponensial.

```
x=randt(1,1000,5); y=randnormal(1,1000,mean(x),dev(x)); ... plot2d("x",r=6,style="—",
,y1="normal",xl="student-t", vertical); ... plot2d(sort(x),sort(y), points,color=red,style="x", add):
```

Plot ini dengan jelas menunjukkan bahwa nilai yang terdistribusi normal cenderung lebih kecil pada ujung yang ekstrim.

Jika kita memiliki dua distribusi dengan ukuran yang berbeda, kita dapat memperluas distribusi yang lebih kecil atau memperkecil distribusi yang lebih besar. Fungsi berikut ini bagus untuk keduanya. Fungsi ini mengambil nilai median dengan persentase antara 0 dan 1.

```
function medianexpand (x,n) := median(x,p=linspace(0,1,n-1));
Mari kita bandingkan dua distribusi yang sama.
x=random(1000); y=random(400); ... plot2d("x",0,1,style="-"); ...
plot2d(sort(medianexpand(x,400)),sort(y), points,color=red,style="x", add):
```

Regresi dan Korelasi

Regresi linier dapat dilakukan dengan fungsi polyfit() atau berbagai fungsi kecocokan.

Sebagai permulaan, kita mencari garis regresi untuk data univariat dengan 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
```

Kami ingin membandingkan kecocokan tanpa bobot dan dengan bobot. Pertama, koefisien dari kecocokan linier.

```
p=polyfit(x,y,1)
[0.733333, 0.812121]
Sekarang, koefisien dengan bobot yang menekankan nilai terakhir.
w = "exp(-(x-10)^2/10)"; pw = polyfit(x,y,1,w = w(x))
[4.71566, 0.38319]
```

Kami menempatkan semuanya ke dalam satu plot untuk titik-titik dan garis regresi, dan untuk bobot yang digunakan.

```
figure(2,1); ... figure(1); statplot(x,y,"b",xl="Regression"); ... plot2d("evalpoly(x,p)", add,color=blue,style="b"); ...
plot2d("evalpoly(x,pw)",5,10, add,color=red,style="-"); ... figure(2);
plot2d(w,1,10, filled,style="r",fillcolor=red,xl=w); ... figure(0):
```

Untuk contoh lain, kita membaca survei tentang siswa, usia mereka, usia orang tua mereka, dan jumlah saudara kandung dari sebuah file.

Tabel ini berisi "m" dan "f" pada kolom kedua. Kita menggunakan variabel tok2 untuk mengatur terjemahan yang tepat dan bukannya membiarkan readtable() mengumpulkan terjemahan.

```
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
```

Bagaimana usia saling bergantung satu sama lain? Kesan pertama datang dari scatterplot berpasangan.

```
scatterplots(tablecol(MS,3:5),hd[3:5]):
```

Jelas bahwa usia ayah dan ibu saling bergantung satu sama lain. Mari kita tentukan dan plot garis regresinya.

```
cs:=MS[,4:5]'; ps:=polyfit(cs[1],cs[2],1)
```

```
[17.3789, 0.740964]
```

Ini jelas merupakan model yang salah. Garis regresinya adalah  $s = 17 + 0,74t$ , di mana  $t$  adalah usia ibu dan  $s$  adalah usia ayah. Perbedaan usia mungkin sedikit bergantung pada usia, tetapi tidak terlalu banyak.

Sebaliknya, kami menduga fungsi seperti  $s = a + t$ . Kemudian  $a$  adalah rata-rata dari  $s-t$ . Ini adalah perbedaan usia rata-rata antara ayah dan ibu.

```
da:=mean(cs[2]-cs[1])
```

```
3.65
```

Mari kita plotkan ini ke dalam satu scatter plot.

```
plot2d(cs[1],cs[2], points); ... plot2d("evalpoly(x,ps)",color=red,style=".", add);
... plot2d("x+da",color=blue, add):
```

Berikut ini adalah plot kotak dari kedua usia tersebut. Ini hanya menunjukkan, bahwa usia keduanya berbeda.

```
boxplot(cs,["mothers","fathers"]):
```

Sangat menarik bahwa perbedaan dalam median tidak sebesar perbedaan dalam mean.

```
median(cs[2])-median(cs[1])
```

```
1.5
```

Koefisien korelasi menunjukkan korelasi positif.

```
correl(cs[1],cs[2])
```

```
0.7588307236
```

Korelasi peringkat adalah ukuran untuk urutan yang sama dalam kedua vektor. Korelasi ini juga cukup positif.

```
rankcorrel(cs[1],cs[2])
```

```
0.758925292358
```

Membuat Fungsi baru

Tentu saja, bahasa EMT dapat digunakan untuk memprogram fungsi baru.

Misalnya, kita mendefinisikan fungsi kemiringan.

di mana  $m$  adalah rata-rata dari  $x$ .

```
function skew (x:vector) ...
```

```
m:=mean(x); return sqrt(cols(x))*sum((x-m)^3)/(sum((x-m)^2))^(3/2);endfunction <
/pre > SepertiyangAndalihat,kitadapatdengandemudahmenggunakanbahasamatriksuntukmendapatkanimpl
```

```
data=normal(20); skew(normal(10))
```

```
0.781590591673
```

Berikut ini adalah fungsi lain, yang disebut koefisien kemencengan Pearson.

```
function skew1 (x) := 3(mean(x)-median(x))/dev(x)
```

```
skew1(data)
```

```
-0.311449531843
```

Simulasi Monte Carlo

Euler dapat digunakan untuk mensimulasikan kejadian acak. Kita telah melihat contoh sederhana di atas. Berikut ini adalah contoh lainnya, yang mensimulasikan 1000 kali pelemparan 3 dadu, dan menanyakan distribusi dari jumlah tersebut.

```
ds:=sum(intrandom(1000,3,6)); fs=getmultiplicities(3:18,ds)
```

```
[7, 12, 27, 54, 106, 108, 119, 126, 122, 93, 65, 53, 35, 11, 6]
```

Kita bisa merencanakan ini sekarang.

```
columnsplot(fs,lab=3:18):
```

Untuk menentukan distribusi yang diharapkan tidaklah mudah. Kami menggunakan rekursi tingkat lanjut untuk hal ini.

Fungsi berikut ini menghitung jumlah cara angka k dapat direpresentasikan sebagai jumlah n angka dalam rentang 1 hingga m. Fungsi ini bekerja secara rekursif dengan cara yang jelas.

```
function map countways (k; n, m) ...
```

```
if n==1 then return k; k=1 k=m else sum=0; loop 1 to m; sum=sum+countways(k-
,n-1,m); end; return sum; end; endfunction i/prej
```

Berikut ini adalah hasil dari tiga lemparan dadu.

```
countways(5:25,5,5)
```

```
[1, 5, 15, 35, 70, 121, 185, 255, 320, 365, 381, 365, 320, 255, 185, 121, 70, 35,
15, 5, 1]
```

```
cw=countways(3:18,3,6)
```

```
[1, 3, 6, 10, 15, 21, 25, 27, 27, 25, 21, 15, 10, 6, 3, 1]
```

Kami menambahkan nilai yang diharapkan ke plot.

```
plot2d(cw/631000, add); plot2d(cw/631000, points, add) :
```

Untuk simulasi lainnya, deviasi nilai rata-rata dari n variabel acak berdistribusi normal 0-1 adalah  $1/\sqrt{n}$ .

```
longformat; 1/sqrt(10)
```

```
0.316227766017
```

Mari kita periksa hal ini dengan sebuah simulasi. Kami menghasilkan 10.000 kali 10 vektor acak.

```
M=normal(10000,10); dev(mean(M)')
```

```
0.316930486807
```

```
plot2d(mean(M)', distribution):
```

Median dari 10 bilangan acak berdistribusi normal 0-1 memiliki deviasi yang lebih besar.

```
dev(median(M)')
```

```
0.370577124758
```

Karena kita dapat dengan mudah menghasilkan jalan acak, kita dapat mensimulasikan proses Wiener. Kami mengambil 1000 langkah dari 1000 proses. Kami kemudian memplot deviasi standar dan rata-rata dari langkah ke-n dari proses-proses ini bersama dengan nilai yang diharapkan dalam warna merah.

```
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):
```

Tes

Tes adalah alat yang penting dalam statistik. Dalam Euler, banyak tes yang diterapkan. Semua tes ini mengembalikan kesalahan yang kita terima jika kita menolak hipotesis nol.

Sebagai contoh, kita menguji lemparan dadu untuk distribusi yang seragam. Pada 600 lemparan, kita mendapatkan nilai berikut, yang kita masukkan ke dalam uji chi-kuadrat.

```
chitest([90,103,114,101,103,89],dup(100,6)')
```

```
0.498830517952
```



Uji chi-square juga memiliki mode, yang menggunakan simulasi Monte Carlo untuk menguji statistik. Hasilnya seharusnya hampir sama. Parameter `gt;p` menginterpretasikan vektor `y` sebagai vektor probabilitas.

```
chitest([90,103,114,101,103,89],dup(1/6,6)', p, montecarlo)
0.488
```

Kesalahan ini terlalu besar. Jadi kita tidak bisa menolak distribusi seragam. Ini tidak membuktikan bahwa dadu kita adil. Tetapi kita tidak dapat menolak hipotesis kita.

Selanjutnya kita buat 1000 lemparan dadu dengan menggunakan generator bilangan acak, dan lakukan pengujian yang sama.

```
n=1000; t=random([1,n6]); chitest(count(t6,6),dup(n,6)')
0.869751903716
```

Mari kita uji nilai rata-rata 100 dengan uji-t.

```
s=200+normal([1,100])10; ... ttest(mean(s),dev(s),100,200)
0.419195594555
```

Fungsi `ttest()` membutuhkan nilai rata-rata, deviasi, jumlah data, dan nilai rata-rata untuk diuji.

Sekarang mari kita periksa dua pengukuran untuk mean yang sama. Kita tolak hipotesis bahwa kedua pengukuran tersebut memiliki nilai rata-rata yang sama, jika hasilnya `lt; 0,05`.

```
tcomparedata(normal(1,10),normal(1,10))
0.023252371527
```

Jika kita menambahkan bias pada satu distribusi, kita akan mendapatkan lebih banyak penolakan. Ulangi simulasi ini beberapa kali untuk melihat efeknya.

```
tcomparedata(normal(1,10),normal(1,10)+2)
0.000495022043408
```

Pada contoh berikut, kita membuat 20 lemparan dadu secara acak sebanyak 100 kali dan menghitung jumlah dadu yang muncul. Rata-rata harus ada  $20/6 = 3,3$  mata dadu.

```
R=random(100,20); R=sum(R6j=1)'; mean(R)
3.5
```

Sekarang kita bandingkan jumlah satu dengan distribusi binomial. Pertama, kita memplot distribusi angka satu.

```
plot2d(R,distribution=max(R)+1,even=1,style="
/"):
t=count(R,21);
```

Kemudian kami menghitung nilai yang diharapkan.

```
n=0:20; b=bin(20,n)(1/6)^n(5/6)^(20-n)100;
```

Kami harus mengumpulkan beberapa angka untuk mendapatkan kategori yang cukup besar.

```
t1=sum(t[1:2])—t[3:7]—sum(t[8:21]); ... b1=sum(b[1:2])—b[3:7]—sum(b[8:21]);
```

Uji chi-square menolak hipotesis bahwa distribusi kita adalah distribusi binomial, jika hasilnya `lt;0,05`.

```
chitest(t1,b1)
0.854492240929
```

Contoh berikut ini berisi hasil dari dua kelompok orang (laki-laki dan perempuan, katakanlah) yang memberikan suara untuk satu dari enam partai.

```
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
```

Kami ingin menguji independensi suara dari jenis kelamin. Uji tabel  $\chi^2$  *melakukan hal ini. Hasilnya terlampirkan.*

```
tabletest(A)
0.990701632326
```

Berikut ini adalah tabel yang diharapkan, jika kita mengasumsikan frekuensi pemungutan suara yang diamati.

```
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
```

Kita dapat menghitung koefisien kontingensi yang telah dikoreksi. Karena koefisien ini sangat dekat dengan 0, kami menyimpulkan bahwa pemungutan suara tidak bergantung pada jenis kelamin.

```
contingency(A)
0.0427225484717
```

Beberapa Tes Lainnya

Selanjutnya kita menggunakan analisis varians (uji F) untuk menguji tiga sampel data yang terdistribusi secara normal dengan nilai rata-rata yang sama. Metode ini disebut ANOVA (analisis varians). Dalam Euler, fungsi `varanalysis()` digunakan.

```
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
```

Ini berarti, kami menolak hipotesis nilai rata-rata yang sama. Kami melakukan ini dengan probabilitas kesalahan sebesar 1,3

Ada juga uji median, yang menolak sampel data dengan distribusi rata-rata yang berbeda dengan menguji median dari sampel gabungan.

```
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
```

Uji lain tentang kesetaraan adalah uji peringkat. Uji ini jauh lebih tajam daripada uji median.

```
ranktest(a,b)
0.00199969612469
```

Dalam contoh berikut ini, kedua distribusi memiliki rata-rata yang sama.

```
ranktest(random(1,100),random(1,50)3-1)
0.311936703478
```

Sekarang mari kita coba mensimulasikan dua perawatan a dan b yang diterapkan pada orang yang berbeda.

```

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];
Uji signum memutuskan, apakah a lebih baik daripada 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
Mari kita coba dua pengujian lagi dengan menggunakan rangkaian yang
dihasilkan.
wilcoxon(normal(1,20),normal(1,20)-1)
0.00556450704119
wilcoxon(normal(1,20),normal(1,20))
0.0726998875325
Bilangan Acak
Berikut ini adalah tes untuk generator bilangan acak. Euler menggunakan
generator yang sangat bagus, jadi kita tidak perlu mengharapkan adanya masalah.
Pertama, kita akan membangkitkan sepuluh juta bilangan acak dalam [0,1].
n:=10000000; r:=random(1,n);
Selanjutnya, kami menghitung jarak antara dua angka yang kurang dari
0,05.
a:=0.05; d:=differences(nonzeros(rja));
Terakhir, kami memplot berapa kali, setiap jarak yang terjadi, dan mem-
bandingkannya dengan nilai yang diharapkan.
m=getmultiplicities(1:100,d); plot2d(m); ... plot2d("n(1-a)^(x-1)a^2", color =
red, add) :
Menghapus data.
remvalue n;
Pengantar untuk Pengguna Proyek R
Jelas, EMT tidak bersaing dengan R sebagai sebuah paket statistik. Namun,
ada banyak prosedur dan fungsi statistik yang tersedia di EMT juga. Jadi EMT
dapat memenuhi kebutuhan dasar. Bagaimanapun, EMT hadir dengan paket
numerik dan sistem aljabar komputer.
Buku ini diperuntukkan bagi Anda yang sudah terbiasa dengan R, tetapi
perlu mengetahui perbedaan sintaks EMT dan R. Kami mencoba memberikan
gambaran umum mengenai hal-hal yang jelas dan kurang jelas yang perlu Anda
ketahui.
Selain itu, kami juga membahas cara-cara untuk bertukar data di antara
kedua sistem tersebut.
Perhatikan bahwa ini adalah pekerjaan yang sedang berlangsung.
Sintaks Dasar
Hal pertama yang Anda pelajari dalam R adalah membuat sebuah vektor.
Dalam EMT, perbedaan utamanya adalah bahwa operator : dapat mengambil
ukuran langkah. Selain itu, operator ini memiliki daya ikat yang rendah.

```

```
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, 9]
```

Fungsi `c()` tidak ada. Anda dapat menggunakan vektor untuk menggabungkan beberapa hal.

Contoh berikut ini, seperti banyak contoh lainnya, berasal dari “Interoduksi ke R” yang disertakan dengan proyek R. Jika Anda membaca PDF ini, Anda akan menemukan bahwa saya mengikuti alurnya dalam tutorial ini.

```
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]
```

Operator titik dua dengan ukuran langkah EMT digantikan oleh fungsi `seq()` dalam R. Kita dapat menulis fungsi ini dalam 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]
```

Fungsi `rep()` dari R tidak ada dalam EMT. Untuk input vektor, dapat dituliskan sebagai berikut.

```
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]
```

Perhatikan bahwa “=” atau “:=” digunakan untuk penugasan. Operator “-gt;” digunakan untuk unit dalam EMT.

```
125km - " miles"
77.6713990297 miles
```

Operator “lt;” untuk penugasan menyesatkan, dan bukan ide yang baik untuk R. Berikut ini akan membandingkan `a` dan `-4` dalam EMT.

```
a=2; a[-4]
0
```

Dalam R, “`alt;-4lt;3`” bisa digunakan, tetapi “`alt;-4lt;-3`” tidak. Saya juga mengalami ambiguitas yang sama di EMT, tetapi saya mencoba untuk menghilangkannya.

EMT dan R memiliki vektor dengan tipe boolean. Tetapi dalam EMT, angka 0 dan 1 digunakan untuk merepresentasikan salah dan benar. Dalam R, nilai benar dan salah tetap dapat digunakan dalam aritmatika biasa seperti dalam EMT.

```
xj5,
[0, 0, 1, 0, 0] [0, 0, 3.1, 0, 0]
```

EMT melempar kesalahan atau menghasilkan NAN tergantung pada flag “kesalahan”.

```
errors off; 0/0, isNAN(sqrt(-1)), errors on;
NAN 1
```

String sama saja dalam R dan EMT. Keduanya berada di lokal saat ini, bukan di Unicode.

Dalam R ada paket-paket untuk Unicode. Dalam EMT, sebuah string dapat berupa string Unicode. Sebuah string Unicode dapat diterjemahkan ke pengkodean lokal dan sebaliknya. Selain itu, u“...” dapat berisi entitas HTML.

```
u"169; Reneacut; Grothmann"
```

© René Grothmann

Berikut ini mungkin atau mungkin tidak ditampilkan dengan benar pada sistem Anda sebagai A dengan titik dan tanda hubung di atasnya. Hal ini tergantung pada jenis huruf yang Anda gunakan.

```
chartoutf([480])
```

Penggabungan string dilakukan dengan “+” atau “—”. Ini dapat menyertakan angka, yang akan dicetak dalam format saat ini.

```
”pi = ”+pi
```

```
pi = 3.14159265359
```

Pengindeksan

Sebagian besar waktu, ini akan bekerja seperti pada R.

Tetapi EMT akan menginterpretasikan indeks negatif dari bagian belakang vektor, sementara R menginterpretasikan  $x[n]$  sebagai  $x$  tanpa elemen ke- $n$ .

```
x, x[1:3], x[-2]
```

```
[10.4, 5.6, 3.1, 6.4, 21.7] [10.4, 5.6, 3.1] 6.4
```

Perilaku R dapat dicapai dalam EMT dengan `drop()`.

```
drop(x,2)
```

```
[10.4, 3.1, 6.4, 21.7]
```

Vektor logika tidak diperlakukan secara berbeda dengan indeks di EMT, berbeda dengan R. Anda harus mengekstrak elemen-elemen yang bukan nol terlebih dahulu di 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]
```

Sama seperti di R, vektor indeks dapat berisi pengulangan.

```
x[[1,2,2,1]]
```

```
[10.4, 5.6, 5.6, 10.4]
```

Namun pemberian nama untuk indeks tidak dimungkinkan dalam EMT. Untuk paket statistik, hal ini mungkin sering diperlukan untuk memudahkan akses ke elemen-elemen vektor.

Untuk meniru perilaku ini, kita dapat mendefinisikan sebuah fungsi sebagai berikut.

```
function sel (v,i,s) := v[indexof(s,i)]; ... s=[”first”,”second”,”third”,”fourth”];
sel(x,[”first”,”third”],s)
```

Trying to overwrite protected function sel! Error in: function sel (v,i,s) := v[indexof(s,i)]; ... ... [10.4, 3.1]

Tipe Data

EMT memiliki lebih banyak tipe data yang tetap dibandingkan R. Jelas, dalam R terdapat vektor yang berkembang. Anda bisa mengatur sebuah vektor numerik kosong  $v$  dan memberikan sebuah nilai pada elemen  $v[17]$ . Hal ini tidak mungkin dilakukan dalam EMT.

Hal berikut ini sedikit tidak efisien.

```
v=[]; for i=1 to 10000; v=v—i; end;
```

EMT sekarang akan membuat vektor dengan  $v$  dan  $i$  yang ditambahkan pada tumpukan dan menyalin vektor tersebut kembali ke variabel global  $v$ .

Semakin efisien mendefinisikan vektor.

```
v=zeros(10000); for i=1 to 10000; v[i]=i; end;
```

Untuk mengubah jenis tanggal di EMT, Anda dapat menggunakan fungsi seperti `complex()`.

```
complex(1:4)
[1+0i , 2+0i , 3+0i , 4+0i]
```

Konversi ke string hanya dapat dilakukan untuk tipe data dasar. Format saat ini digunakan untuk penggabungan string sederhana. Tetapi ada fungsi-fungsi seperti `print()` atau `frac()`.

Untuk vektor, Anda dapat dengan mudah menulis fungsi Anda sendiri.

```
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 i/pre; 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]
```

Untuk komunikasi dengan Maxima, ada sebuah fungsi `convertmxm()`, yang juga dapat digunakan untuk memformat vektor untuk output.

```
convertmxm(1:10)
[1,2,3,4,5,6,7,8,9,10]
```

Untuk LaTeX, perintah `tex` dapat digunakan untuk mendapatkan perintah LaTeX.

```
tex([1,2,3])
[1, 2, 3]
```

#### Faktor dan Tabel

Pada pengantar R terdapat sebuah contoh dengan apa yang disebut faktor.

Berikut ini adalah daftar wilayah dari 30 negara bagian.

```
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"];
```

Asumsikan, kita memiliki pendapatan yang sesuai di setiap negara bagian.

```
incomes = [60, 49, 40, 61, 64, 60, 59, 54, 62, 69, 70, 42, 56, ... 61, 61, 61,
58, 51, 48, 65, 49, 49, 41, 48, 52, 46, ... 59, 46, 58, 43];
```

Sekarang, kita ingin menghitung rata-rata pendapatan di wilayah tersebut.

Sebagai sebuah program statistik, R memiliki fungsi `factor()` dan `tapply()` untuk hal ini.

EMT dapat melakukan hal ini dengan mencari indeks dari wilayah-wilayah di dalam daftar unik dari wilayah-wilayah tersebut.

```
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, 2, 8, 5, 1, 2, 7, 7, 1]
```

Pada titik ini, kita dapat menulis fungsi perulangan kita sendiri untuk melakukan berbagai hal untuk satu faktor saja.

Atau kita dapat meniru fungsi `tapply()` dengan cara berikut.

```
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 <
/pre > Inisedikittidakefisien,karenamenghitungwilayahunikuntuksetiapit,tetapiberfungsi.
tappl(auterr,"mean",austates,incomes)
[44.5, 57.333333333333, 55.5, 53.6, 55, 60.5, 56, 52.25]
Perhatikan bahwa ini bekerja untuk setiap vektor wilayah.
tappl(["act", "nsw"], "mean", austates, incomes)
```

[44.5, 57.3333333333]

Sekarang, paket statistik EMT mendefinisikan tabel seperti halnya di R. Fungsi `readtable()` dan `writetable()` dapat digunakan untuk input dan output.

Jadi kita dapat mencetak rata-rata pendapatan negara di wilayah dengan cara yang ramah.

```
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
```

Kita juga dapat mencoba meniru perilaku R sepenuhnya.

Faktor-faktor tersebut harus disimpan dengan jelas dalam sebuah koleksi dengan jenis dan kategorinya (negara bagian dan wilayah dalam contoh kita). Untuk EMT, kita menambahkan indeks yang telah dihitung sebelumnya.

```
function makef (t) ...
```

Factor data Returns a collection with data t, unique data, indices. See: `tappl u=sort(unique(t)); return t,u,indexofsorted(u,t); endfunction i/pre` `statef=makef(austates);`

Sekarang elemen ketiga dari koleksi ini akan berisi indeks.

```
statef[3]
```

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

Sekarang kita dapat meniru `tappl()` dengan cara berikut. Ini akan mengembalikannya sebagai kumpulan data tabel dan judul kolom.

```
function tappl (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]); end if; end; return x, uf; endfunction < /pre >
```

*Wedidnotaddmuchtypecheckinghere.Theonlyprecautionconcernscategories(factors)withnodata.Butoneshould*

This table can be printed as a table with `writetable()`.

```
writetable(tappl(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
```

Larik

EMT hanya memiliki dua dimensi untuk array. Tipe datanya disebut matriks. Akan lebih mudah untuk menulis fungsi untuk dimensi yang lebih tinggi atau pustaka C untuk ini.

R memiliki lebih dari dua dimensi. Dalam R, larik adalah sebuah vektor dengan sebuah bidang dimensi.

Dalam EMT, sebuah vektor adalah sebuah matriks dengan satu baris. Ini bisa dibuat menjadi sebuah matriks dengan `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
```

Ekstraksi baris dan kolom, atau sub-matriks, sama seperti di R.

```
X[,2:3]
```

```
2 3 7 8 12 13 17 18
```

Namun, dalam R dimungkinkan untuk mengatur daftar indeks tertentu dari vektor ke suatu nilai. Hal yang sama juga dapat dilakukan dalam EMT hanya dengan sebuah perulangan.

```
function setmatrixvalue (M, i, j, v) ...
```

```
loop 1 to max(length(i),length(j),length(v)) M[i,j] = v; end; endfunction
i/pre Kami mendemonstrasikan hal ini untuk menunjukkan bahwa matriks
```

dilewatkan dengan referensi di EMT. Jika Anda tidak ingin mengubah matriks asli M, Anda perlu menyalinnya dalam fungsi.

```
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
```

Hasil kali luar dalam EMT hanya dapat dilakukan di antara vektor. Hal ini otomatis karena bahasa matriks. Satu vektor harus berupa vektor kolom dan vektor baris.

```
(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
```

Dalam pengantar PDF untuk R ada sebuah contoh, yang menghitung distribusi ab-cd untuk a, b, c, d yang dipilih dari 0 sampai n secara acak. Solusinya dalam R adalah membentuk sebuah matriks 4 dimensi dan menjalankan table() di atasnya.

Tentu saja, ini bisa dicapai dengan sebuah perulangan. Tetapi perulangan tidak efektif dalam EMT atau R. Dalam EMT, kita bisa menulis perulangan dalam C dan itu adalah solusi tercepat.

Tetapi kita ingin meniru perilaku R. Untuk ini, kita perlu meratakan perkalian ab dan membuat sebuah matriks ab-cd.

```
a=0:6; b=a'; p=flatten(ab); q=flatten(p-p'); ... u=sort(unique(q)); f=getmultiplicities(u,q);
... statplot(u,f,"h"):
```

Selain kelipatan yang tepat, EMT dapat menghitung frekuensi dalam vektor.

```
getfrequencies(q,-50:10:50)
[0, 23, 132, 316, 602, 801, 333, 141, 53, 0]
```

Cara yang paling mudah untuk memplot ini sebagai distribusi adalah sebagai berikut.

```
plot2d(q,distribution=11):
```

Tetapi juga memungkinkan untuk menghitung jumlah dalam interval yang dipilih sebelumnya. Tentu saja, berikut ini menggunakan getfrequencies() secara internal.

Karena fungsi histo() mengembalikan frekuensi, kita perlu menskalakannya sehingga integral di bawah grafik batang adalah 1.

```
x,y=histo(q,v=-55:10:55); y=y/sum(y)/differences(x); ... plot2d(x,y, bar,style="/"):
```

Daftar

EMT memiliki dua jenis daftar. Yang pertama adalah daftar global yang dapat diubah, dan yang kedua adalah jenis daftar yang tidak dapat diubah. Kita tidak peduli dengan daftar global di sini.

Tipe daftar yang tidak dapat diubah disebut koleksi dalam EMT. Ia berperilaku seperti struktur dalam C, tetapi elemen-elemennya hanya diberi nomor dan tidak diberi nama.

```
L="Fred","Flintstone",40,[1990,1992]
```

```
Fred Flintstone 40 [1990, 1992]
```

Saat ini elemen-elemen tersebut tidak memiliki nama, meskipun nama dapat ditetapkan untuk tujuan khusus. Elemen-elemen tersebut diakses dengan angka.

```
(L[4])[2]
```

```
1992
```

Input dan Output File (Membaca dan Menulis Data)



Anda mungkin sering ingin mengimpor matriks data dari sumber lain ke EMT. Tutorial ini akan menjelaskan kepada Anda tentang berbagai cara untuk melakukan hal tersebut. Fungsi yang sederhana adalah `writematrix()` dan `readmatrix()`.

Mari kita tunjukkan bagaimana cara membaca dan menulis sebuah vektor real ke sebuah file.

```
a=random(1,100); mean(a), dev(a),
0.50079 0.29283
```

Untuk menulis data ke sebuah berkas, kita menggunakan fungsi `writematrix()`.

Karena pengenalan ini kemungkinan besar berada di sebuah direktori, di mana pengguna tidak memiliki akses tulis, kita menulis data ke direktori home pengguna. Untuk notebook sendiri, hal ini tidak diperlukan, karena file data akan ditulis ke dalam direktori yang sama.

```
filename="test.dat";
```

Sekarang kita tuliskan vektor kolom `a'` ke dalam file. Hal ini akan menghasilkan satu angka pada setiap baris file.

```
writematrix(a',filename);
```

Untuk membaca data, kita menggunakan `readmatrix()`.

```
a=readmatrix(filename)';
```

Dan hapus file tersebut.

```
fileremove(filename);
```

```
mean(a), dev(a),
```

```
0.50079 0.29283
```

Fungsi `writematrix()` atau `writetable()` dapat dikonfigurasi untuk bahasa lain.

Sebagai contoh, jika Anda memiliki sistem bahasa Indonesia (titik desimal dengan koma), Excel Anda membutuhkan nilai dengan koma desimal yang dipisahkan oleh titik koma dalam file csv (defaultnya adalah nilai yang dipisahkan dengan koma). File “test.csv” berikut ini akan muncul di folder current Anda.

```
filename="test.csv"; ... writematrix(random(5,3),file=filename,separator=",");
```

Anda sekarang dapat membuka file ini dengan Excel Indonesia secara langsung.

```
fileremove(filename);
```

Terkadang kita memiliki string dengan token seperti berikut ini.

```
s1:="f m m f m m m f f f m m f"; ... s2:="f f f m m f f";
```

Untuk menandai ini, kita mendefinisikan vektor token.

```
tok:=["f", "m"]
```

```
f m
```

Kemudian kita dapat menghitung berapa kali setiap token muncul dalam string, dan memasukkan hasilnya ke dalam tabel.

```
M:=getmultiplicities(tok, strtokens(s1))-... getmultiplicities(tok, strtokens(s2));
```

Tulis tabel dengan tajuk token.

```
writetable(M, labc=tok, labr=1:2, wc=8)
```

```
f m 1 6 7 2 5 2
```

Untuk statika, EMT dapat membaca dan menulis tabel.

```
file="test.dat"; open(file,"w"); ... writeln("A,B,C"); writematrix(random(3,3));
... close();
```

File terlihat seperti ini.

```
printfile(file)
A,B,C 0.8988055216141367,0.7302744388921779,0.7931856542922466 0.1917462336357822,0.1282312860245
0.3417257196487329,0.7469069247539568,0.7009360406452619
```

Fungsi readtable() dalam bentuknya yang paling sederhana dapat membaca ini dan mengembalikan sebuah koleksi nilai dan baris judul.

```
L=readtable(file, list);
```

Koleksi ini dapat dicetak dengan writetable() ke buku catatan, atau ke sebuah file.

```
writetable(L,wc=10,dc=5)
A B C 0.89881 0.73027 0.79319 0.19175 0.12823 0.2201 0.34173 0.74691
0.70094
```

Matriks nilai adalah elemen pertama dari L. Perhatikan bahwa mean() dalam EMT menghitung nilai rata-rata dari baris-baris matriks.

```
mean(L[1])
0.80742 0.18003 0.59652
```

File CSV

Pertama, mari kita tulis sebuah matriks ke dalam sebuah file. Untuk keluarannya, kami membuat file di direktori kerja saat ini.

```
file="test.csv"; ... M=random(3,3); writematrix(M,file);
Berikut ini adalah isi file ini.
printfile(file)
0.9436089565077617,0.7258961441466456,0.970113946338969 0.5492308273374198,0.4226400092823435,0.27
0.04924212823881809,0.7158403456427294,0.6061594232560839
```

CSV ini dapat dibuka di sistem bahasa Inggris ke Excel dengan klik dua kali. Jika Anda mendapatkan file seperti itu pada sistem Jerman, Anda perlu mengimpor data ke Excel dengan memperhatikan titik desimal.

Namun, titik desimal juga merupakan format default untuk EMT. Anda dapat membaca sebuah matriks dari sebuah file dengan readmatrix().

```
readmatrix(file)
0.94361 0.7259 0.97011 0.54923 0.42264 0.2734 0.049242 0.71584 0.60616
```

Dimungkinkan untuk menulis beberapa matriks ke dalam satu file. Perintah open() dapat membuka file untuk menulis dengan parameter "w". Standarnya adalah "r" untuk membaca.

```
open(file,"w"); writematrix(M); writematrix(M'); close();
Matriks-matriks tersebut dipisahkan oleh sebuah baris kosong. Untuk mem-
baca matriks, buka file dan panggil readmatrix() beberapa kali.
open(file); A=readmatrix(); B=readmatrix(); A==B, close();
1 0 0 0 1 0 0 0 1
```

Di Excel atau spreadsheet serupa, Anda dapat mengekspor matriks sebagai CSV (nilai yang dipisahkan dengan koma). Pada Excel 2007, gunakan "save as" dan "format lain", lalu pilih "CSV". Pastikan, tabel saat ini hanya berisi data yang ingin Anda ekspor.

Berikut ini adalah contohnya.

```

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

```

Seperti yang Anda lihat, sistem Jerman saya menggunakan titik koma sebagai pemisah dan koma desimal. Anda dapat mengubahnya di pengaturan sistem atau di Excel, tetapi tidak perlu untuk membaca matriks ke dalam EMT.

Cara termudah untuk membaca ini ke dalam Euler adalah `readmatrix()`. Semua koma digantikan oleh titik dengan parameter `gt;comma`. Untuk CSV bahasa Inggris, hilangkan saja parameter ini.

```

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

```

Mari kita plot ini.

```
plot2d(M'[1],M'[2:3], points,color=[red,green]');
```

Ada beberapa cara yang lebih mendasar untuk membaca data dari file. Anda dapat membuka file dan membaca angka baris demi baris. Fungsi `getvectorline()` akan membaca angka dari sebuah baris data. Secara default, fungsi ini mengharapkan sebuah titik desimal. Tetapi fungsi ini juga dapat menggunakan koma desimal, jika Anda memanggil `setdecimaldot(",")` sebelum menggunakan fungsi ini.

Fungsi berikut ini adalah contohnya. Fungsi ini akan berhenti pada akhir file atau baris kosong.

```

function myload (file) ...
open(file); M=[]; repeat until eof(); v=getvectorline(3); if length(v);0 then
M=M_v; elsebreak; endif; end; return M; close(file); endfunction < /pre > myload(file)
0.94361 0.7259 0.97011 0.54923 0.42264 0.2734 0.049242 0.71584 0.60616

```

Anda juga dapat membaca semua angka dalam file tersebut dengan `getvector()`.

```

open(file); v=getvector(10000); close(); redim(v[1:9],3,3)
0.94361 0.7259 0.97011 0.54923 0.42264 0.2734 0.049242 0.71584 0.60616

```

Dengan demikian, sangat mudah untuk menyimpan vektor nilai, satu nilai di setiap baris dan membaca kembali vektor ini.

```

v=random(1000); mean(v)
0.49495
writematrix(v',file); mean(readmatrix(file)')
0.49495

```

Menggunakan Tabel

Tabel dapat digunakan untuk membaca atau menulis data numerik. Sebagai contoh, kita menulis tabel dengan judul baris dan kolom ke file.

```

file="test.tab"; M=random(3,3); ... open(file,"w"); ... writetable(M,separator=",",labc=["one","two","
... close(); ... printfile(file)

```

```
one,two,three 0.91, 0.38, 0.78 0.01, 0.62, 0.63 0.71, 0.2, 0.82
```

File ini dapat diimpor ke Excel.

Untuk membaca file di EMT, kita menggunakan `readtable()`.

```

M,headings=readtable(file, clabs); ... writetable(M,labc=headings)
one two three 0.91 0.38 0.78 0.01 0.62 0.63 0.71 0.2 0.82

```

Menganalisis Garis

Anda bahkan dapat mengevaluasi setiap baris dengan tangan. Misalkan, kita memiliki baris dengan format berikut.

```

line="2020-11-03,Tue,1'114.05"
2020-11-03,Tue,1'114.05

```

Pertama, kita dapat memberi tanda pada garis tersebut.

```

vt=strtokens(line)
2020-11-03 Tue 1'114.05

```

Kemudian, kita dapat mengevaluasi setiap elemen garis dengan menggunakan evaluasi yang sesuai.

```

day(vt[1]), ... indexof(["mon","tue","wed","thu","fri","sat","sun"],tolower(vt[2])),
... strrepl(vt[3],"","")()
7.3816e+05 2 1114

```

Dengan menggunakan ekspresi reguler, Anda dapat mengekstrak hampir semua informasi dari sebuah baris data.

Anggaplah kita memiliki baris dokumen HTML berikut ini.

```

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>

```

Untuk mengekstrak ini, kita menggunakan ekspresi reguler, yang mencari

- tanda kurung tutup gt;, - setiap string yang tidak mengandung tanda kurung dengan sub-pencocokan "...",
- kurung pembuka dan kurung penutup menggunakan solusi terpendek,
- sekali lagi, semua string yang tidak mengandung tanda kurung,
- dan sebuah kurung pembuka lt;.

Ekspresi reguler agak sulit untuk dipelajari tetapi sangat kuat.

```

pos,s,v=stxrfnd(line,"(<]+) < .+? (<]+) < ");

```

Hasilnya adalah posisi kecocokan, string yang cocok, dan vektor string untuk sub-cocokan.

```

for k=1:length(vt); vt[k](), end;
1145.5 5.6

```

Berikut ini adalah fungsi yang membaca semua item numerik antara lt;tdgt; dan lt;/tdgt;.

```

function readtd (line) ...
v=[]; cp=0; repeat pos,s,v=stxrfnd(line,"<td.*?(<.+?)</td>",cp); until pos==0;
if length(vt)>0 then v=v—vt[1]; endif; cp=pos+strlen(s); end; return v; end-
function j/pre< readtd(line+"<td non-numerical</td>")
1145.45 5.6 -4.5 non-numerical

```

Membaca dari Web

Situs web atau file dengan URL dapat dibuka di EMT dan dapat dibaca baris demi baris.

Dalam contoh, kita membaca versi saat ini dari situs EMT. Kami menggunakan ekspresi reguler untuk memindai "Versi ..." dalam judul.

```

function readversion () ...

```

```

urlopen("http://www.euler-math-toolbox.de/Programs/Changes.html"); re-
peat until urleof(); s=urlgetline(); k=strfind(s,"Version ",1); if k<0 then substring(s,k,strfind(s,"i",k)-
1), break; endif; end; urlclose(); endfunction i/prej readversion

```

Version 2024-01-12

Masukan dan Keluaran Variabel

Anda dapat menulis variabel dalam bentuk definisi Euler ke file atau ke baris perintah.

```

writevar(pi,"mypi");
mypi = 3.141592653589793;

```

Untuk pengujian, kami membuat file Euler di direktori kerja EMT.

```

file="test.e"; ... writevar(random(2,2),"M",file); ... printfile(file,3)
M = [.. 0.2497656416288016, 0.775763913961096; 0.4610794937199713,
0.6500435412180149];

```

Sekarang kita dapat memuat file tersebut. Ini akan mendefinisikan matriks M.

```

load(file); show M,
M = 0.24977 0.77576 0.46108 0.65004

```

Sebagai catatan, jika writevar() digunakan pada sebuah variabel, maka ia akan mencetak definisi variabel dengan nama variabel tersebut.

```

writevar(M); writevar(inch)
M = [.. 0.2497656416288016, 0.775763913961096; 0.4610794937199713,
0.6500435412180149]; inch= 0.0254;

```

Kita juga dapat membuka file baru atau menambahkan ke file yang sudah ada. Dalam contoh ini, kami menambahkan ke file yang telah dibuat sebelumnya.

```

open(file,"a"); ... writevar(random(2,2),"M1"); ... writevar(random(3,1),"M2");
... close();
load(file); show M1; show M2;
M1 = 0.981 0.999 0.82766 0.54879 M2 = 0.54617 0.91751 0.19975
Untuk menghapus file, gunakan fileremove().
fileremove(file);

```

Sebuah vektor baris dalam sebuah file tidak membutuhkan koma, jika setiap angka berada dalam baris baru. Mari kita buat file seperti itu, dengan menulis setiap baris satu per satu dengan writeln().

```

open(file,"w"); writeln("M = ["); ... for i=1 to 5; writeln(" "+random());
end; ... writeln("];"); close(); ... printfile(file)
M = [0.665296499408 0.83789814693 0.851469427987 0.0564923737207 0.612061825127
];
load(file); M
[0.6653, 0.8379, 0.85147, 0.056492, 0.61206]

```

Latihan

1. misalkan punya vektor  $x=[2,4,6,8,10]$ 
    - a. buat vektor yang menggabungkan vektor  $x$ , angka 0 dan vektor  $x$  lagi
    - b. tentukan apakah setiap elemen vektor  $x$  lebih besar dari 5 (hasil logika 1 untuk benar dan 0 untuk salah)
- ```

x:=[2,4,6,8,10]; [x,0,x]

```

```

[2, 4, 6, 8, 10, 0, 2, 4, 6, 8, 10]
x 5,
[0, 0, 1, 1, 1] [0, 0, 6, 8, 10]
2. tentukan matriks x dengan elemen-elemen yang berurutan dari 1 hingga
20 dan susunlah elemen tersebut menjadi matriks berukuran 5x4.
shortformat; X=redim(1:20,5,4)
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
3. seorang analis memiliki data penjualan harian selama 5 hari (150,200,250,300,350)
yang disimpan dalam bentuk vektor sebagai berikut:
a. mean
b. deviasi standar
a=[150,200,250,300,350]
[150, 200, 250, 300, 350]
mean(a)
250
dev(a)
79.057
4. Hitung rata-rata, median, dan modus dari data berikut:
data = [72, 65, 85, 78, 92, 83, 75, 88, 70, 82]
tentukan:
a. mean
b. deviasi standar
c. box plot
data=[72,65,85,78,92,83,75,88,70,82]
[72, 65, 85, 78, 92, 83, 75, 88, 70, 82]
mean(data)
79
dev(data)
8.52447456836
boxplot(data)
5. Diketahui data nilai ujian siswa: [75, 80, 85, 90, 92, 88, 82, 78]. Buat
grafik distribusi normal untuk data tersebut.
tentukan:
a. median
b. mean
c. deviasi standar
nilai = [75, 80, 85, 90, 92, 88, 82, 78];
median(nilai)
83.5
mean(nilai)
83.75
dev(nilai)
6.01783064852
6. Hitung kuartil pertama (Q1) dan kuartil ketiga (Q3) dari data berikut:
[6, 3, 9, 5, 8, 2, 7]
quantile([6, 3, 9, 5, 8, 2, 7], 0.25)

```

```

4
quantile([6, 3, 9, 5, 8, 2, 7], 0.75)
7.5
7. Hitung jangkauan dari data berikut: [18, 22, 15, 20, 19]
max([18, 22, 15, 20, 19]) - min([18, 22, 15, 20, 19])
7
8. Hitung kuartil pertama (Q1) dari data berikut: [7, 9, 6, 8, 10, 5]
quantile([7,9,6,8,10,5],0.25)
6.25
9. misalkan punya vektor x=[1,3,5,7,9,11,13]
a. buat vektor yang menggabungkan vektor x, angka 0 dan vektor x lagi
b. tentukan apakah setiap elemen vektor x lebih besar dari 5 (hasil logika 1
untuk benar dan 0 untuk salah)
x:=[1,3,5,7,9,11]; [x,0,x]
[1, 3, 5, 7, 9, 11, 0, 1, 3, 5, 7, 9, 11]
x 5,
[0, 0, 0, 1, 1, 1] [0, 0, 0, 7, 9, 11]
10. tentukan matriks x dengan elemen-elemen yang berurutan dari 1 hingga
36 dan susunlah elemen tersebut menjadi matriks berukuran 6x6.
shortformat; X=redim(1:36,6,6)
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
29 30 31 32 33 34 35 36

```