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.)

KEZIA DINI PRAKOSA 24030130017

Contoh Perhitungan

Menyederhanakan bentuk aljabar:

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

$$4x^{-2}y^4 \times -3x^2y^{-7}$$

$$(5x^4 + 3y^2) + (3x^4)$$

Hasil yang ditampilkan adalah penyederhanaan bentuk aljabar dengan perkalian dimana bisa saja hasilnya adalah

$$-12x^0 \times y^-3$$

```
>$&4*x^(-2)*y^4*-3*x^2*y^(-7)
```

$$-\frac{12}{y^3}$$

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

$$-\frac{63\,y^5}{x^{10}}$$

```
>$&(5*x^4+3*y^2)+(3*x^4)
```

$$3\,y^2+8\,x^4$$

Menjabarkan:

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

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

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

Hasil yang ditampilkan dari perintah expand adalah penguraian dari suatu ekspresi aljabar dengan membentuknya menjadi bentuk terpanjang dari hasil operasi perkalian seperti di contoh

Baris Perintah

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

Baris perintah berikut ini hanya akan mencetak hasil dari ekspresi, bukan penugasan atau perintah format:

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

```
16.7551608191
```

Pemisahan Perintah

Perintah harus dipisahkan dengan tanda kosong atau spasi. Baris perintah berikut ini menampilkan dua hasil:

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

```
50.2654824574
100.530964915
```

Baris perintah dieksekusi sesuai urutan saat pengguna menekan enter. Jadi, anda akan mendapatkan nilai beru setiap kali menjalankan baris kedua.

```
>x := 1, y:=3
```

```
1
3
```

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

```
0.654289790498
```

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

```
0.540302305868
-0.9899924966
```

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

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

```
1.41666666667
1.41421568627
1.41421356237
```

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

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

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

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

```
81
```

Euler mendukung perulangan di baris perintah, selama mereka masuk ke dalam suatu perulangan garis atau multi-garis. Dalam program, pembatasan ini tidak berlaku, dari kursus. Untuk iformasi 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-baris. Pastikan baris diakhiri dengan "..."

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

```
1.41421356237
```

Struktur bersyarat juga berfungsi.

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

```
Thought so!
```

Saat anda menjalankan perintah, kursor dapat berada di posisi mana pun dalam perintah garis. Anda dapat mengklik ke bagian komentar diatas perintah untuk pergi ke printah.

Saat anda menggerakan kursor di sepanjang garis, pasangan pembuka dan penutup dari kurung atau kurung akan disorot. Juga, perhatikan barus status. Setelah kurung buka dari fungsi sqrt(), beris status akan menampilkan teks bantuan untuk fungsi tersebut. Jalankan perintah dengan tombol return.

```
>sqrt(sin(10°)/cos(20°))
```

```
0.429875017772
```

You can double click on any command to open the help for this command. Try double clicking the exp command below in the command line.
Untuk mengetahui bantuan untuk perintah terbaru, buka jendela bantuan dengan F1. Di sana, Anda dapat memasukkan teks untuk dicari. Di baris kosong, bantuan jendela akan ditampilkan. Anda dapat menekan escape (ESC) untuk menghapus garis, atau menutup jendela bantuan.
Anda dapat mengklik dua kali perintah exp di bawah ini di barus perintah.

```
>exp(log(2.5))
```

```
2.5
```

Anda bisa menyalin dan menempel di Euler. Gunakan Ctrl-C dan Ctrl-V untuk hal ini. Untuk menandai teks, seret mouse atau gunakan shift bersama dengan tombol kursor apapun. Yang lebih lagi, anda dapat menyalin tanda kurung yang disorot.

Basis sintaks

Euler mengenal fungsi-fungsi matematika biasa. Seperti yang telah terlihat di atas, fungsi trigonometri bekerja dalam radian atau derajat. Untuk mengubah ke derajat, tambahkan simbol derajat (dengan tombol F7) pada nilai, atau gunakan fungsi rad(x). Fungsi akar kuadrat disebut sqrt dalam Euler. Tentu saja, xˆ(1/2) juga dapat digunakan.

Untuk menetapkan variabel, gunakan tanda "=" atau ":=". Demi kejelasan, pengenalan ini menggunakan bentuk terakhir. Spasi tidak berpengaruh. Namun spasi antar perintah diharapkan ada.
eberapa perintah dalam satu baris dipisahkan dengan "," atau ";". Tanda titik koma ; menekan (menyembunyikan) output dari perintah. Pada akhir baris perintah, tanda "," diasumsikan jika ; tidak ada.

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

```
30.65625
```

EMT menggunakan sintaks pemrograman untuk ekspresi. Memasuki

lateks: eˆ2 \cdot \left(\frac{1}{3+4 \log(0.6)}+\frac{1}{7} \right)

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

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

```
8.77908249441
```

Untuk menghitung ekspresi yang rumit seperti

lateks: \left(\frac{\frac17 + \frac18 + 2}{\frac13 + \frac12}\right)^2 \pi

Anda harus memasukkannya dalam bentuk baris.

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

```
23.2671801626
```

Tempatkan tanda kurung dengan hati-hati pada sub-ekspresi yang harus dihitung terlebih dahulu. EMT membantu Anda dengan menyoroti ekspresi yang ditutup oleh tanda kurung penutup. Anda juga harus menuliskan nama pi untuk huruf Yunani p.
asil dari perhitungan ini adalah bilangan desimal (floating point). Secara bawaan, hasil ditampilkan dengan ketelitian sekitar 12 digit.

Pada baris perintah berikut, kita juga belajar bagaimana cara merujuk pada hasil sebelumnya dalam baris yang sama.

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

```
0.47619047619
10/21
```

Sebuah perintah Euler dapat berupa ekspresi atau perintah primitif. Ekspresi terdiri atas operator dan fungsi. Jika diperlukan, ekspresi harus mengandung tanda kurung untuk memaksa urutan eksekusi yang benar.

Jika ragu, menggunakan tanda kurung adalah ide yang baik. EMT menampilkan tanda kurung buka dan tutup saat mengedit baris perintah.

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

```
14.4978445072
```

Operator numerik dalam Euler meliputi:

+ tanda plus (unary) atau operator penjumlahan
- tanda minus (unary) atau operator pengurangan
*, /
. perkalian matriks
a^b perpangkatan untuk a positif atau b bilangan bulat (a**b) juga berlaku)
n! operator faktorial

dan banyak lagi.

Berikut beberapa fungsi yang mungkin Anda perlukan (masih banyak lainnya):

sin, cos, tan, atan, asin, acos, rad, deg
log, exp, log10, sqrt, logbase
bin, logbin, logfac, mod, floor, ceil, round, abs, sign
conj, re, im, arg, conj, real, complex
beta, betai, gamma, complex
gamma, ellrf, ellf, ellrd, elle

bitand, bitor, bitxor, bitnot

Beberapa perintah memiliki alias, misalnya ln untuk log.

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

```
2
0.5
```

```
>sin(30°)
```

```
0.5
```

Pastikan selalu menggunakan tanda kurung bulat () bila ada keraguan terhadap urutan eksekusi!
ontoh berikut tidak sama dengan (2^3)^4. Perintah 2^3^4 secara bawaan diartikan sebagai 2^(3^4) (beberapa sistem numerik melakukan sebaliknya).

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

```
2.41785163923e+24
4096
2.41785163923e+24
```

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

```
>longest 1/7
```

```
      0.1428571428571428
```

Representasi ganda internal

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

```
  1.0101010101010101010101010101010101010101010101010101*2^-2
```

Representasi dalam heksadesimal:

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

```
  5.5555555555554*16^-1
```

String

String merupakan kumpulan karakter (teks) yang ditulis di dalam tanda kutip ganda "..."

```
>"Ini merupakan string"
```

```
Ini merupakan string
```

Menggabungkan String

```
>"Luas dari lingkaran dengan jari-jari "+3+" cm adalah "+pi*9+" cm^2"
```

```
Luas dari lingkaran dengan jari-jari 3 cm adalah 28.2743338823 cm^2
```

String bisa berisi huruf, angka, maupun simbol. Contoh berikut merupakan operasi dasar menggunakan string. Kita dapat menggabungkan string dengan angka maupun kumpulan karakter lain menggunakan tanda + maupun | sebagai operator penggabung.

Fungsi Print

```
>"Angka Pi (7 digit) adalah " +print(pi,7,0)
```

```
Angka Pi (7 digit) adalah 3.1415927
```

Fungsi print digunakan untuk mengubah angka jadi string dengan format tertentu yaitu berapa banyak digit yang ditampilkan.

Format umum: print(nilai, jumlah_digit, jumlah_tempat)
nilai: angka yang mau ditampilkan (contoh: pi, golden ratio)
jumlah_digit: berapa banyak digit desimal yang mau ditampilkan (di contoh 7 digit)
jumlah_tempat: 0 jika ingin tanpa spasi tambahan

Nilai None

```
>v:= [none]; v| ["a", "b"]
```

```
a
b
```

Nilai khusus bernama none digunakan ketika suatu fungsi tidak menghasilkan apapun. none ini bisa dikatakan sebagai ”kosong” atau ”tidak ada hasil” yang nantinya tidak akan menampilkan apa-apa.

Dalam contoh, dalam vektor string yang digabung dengan [”a”,”b”] hasil yang ditampilkan hanya ”a” dan ”b” karena none di awal dianggap kosong.

Vektor String

```
>"127.9"()
```

```
127.9
```

Kalau string berisi angka dan akan digunakan sebagai anka sungguhan yang bisa dihitung, kita perlu evaluasi angka tersebut dengan memberikan tanda kurung () di belakang string.

```
>"127.9"() +3
```

```
130.9
```

Ketika angka sudah di evaluasi, angka bisa digunakan untuk menghitung. Seperti pada contoh, angka 127.9 +3 menghasilkan output 130.9 karena angka 127.9 sudah dievaluasi menjadi angka sungguhan.

```
>"127.9" +3
```

```
127.93
```

Jika angka tidak di evaluasi lalu ditambahkan +3 di belakangnya, output yang dihasilkan bukan berupa hitungan namun hanya menggabung teks seperti pada contoh di awal.

```
>w:= ["math","euler","string"]
```

```
math
euler
string
```

Untuk membuat list string, gunakan format [...] dengan pemisah antar elemen berupa "," tanda koma

```
>v:= ["9","4","3"]
```

```
9
4
3
```

Vektor string juga dapat berupa angka

```
>x:= v|v
```

```
9
4
3
9
4
3
```

Vektor string dapat digabungkan dengan tanda |

Unicode

```
>u"&pi; = "+print(pi,2,0),
```

```
 = 3.14
```

Huruf u pada depan perintah string merupakan syarat untuk menulis format unicode. String Unicode ini digunakan untuk menampilkan simbol-simbol khusus misalnya huruf yunani, simbol derajat, dan lain-lain. Dengan menggunakan u”..” euler mengetahui bahwa isi dari string bisa mengandung Unicode atau entiTA HTML.

```
>w=strtochar (u"&alpha;&beta;&pi;")
```

```
[945,  946,  960]
```

strtochar() merupakan fungsi yang digunakan untuk mengubah string unicode menjadi daftar kode angka. Jadi setiap huruf akan memiliki kode unik berupa angka.
Artinya, alpha memiliki kode 945, beta kodenya 946, dan pi kodenya 960

```
>chartoutf(w)
```

chartoutf() merupakan fungsi kebalikan dari strtochar(). chartoutf() digunakan untuk mengubah daftar kode angka unicode kembali menjadi string. Jadi kalau strtochar() menjadikan huruf menjadi kode angka unicode, chartoutf() mengembalikan kode angka unicode menjadi huruf lagi.

```
>u"&#931;"
```

Nilai Boolean

Nilai boolean sebagai logika benar atau salah di representasikan dalam euler dengan 1= true (benar) dan 0=false (salah.

```
>0>1, "car"<"plane"
```

```
0
1
```

Dalam nilai boolean, string juga dapat dibandingan seperti halnya angka. Pada contoh, 0>1 bernilai 0(false) karena 0 tidak lebih dari 1. Lalu string yang dibandingkan adalah mobil < pesawat bernilai 1(true) karena secara alfabet, car dianggap lebih kecil dari plane.

Operator Boolean

```
>4>pi && pi>2
```

```
1
```

&& merupakan operator boolean yang berarti dan atau dalam logika himpunan disebut konjungsi yang kedua pernyataannya benar agar hasilnya 1(true). Pada contoh hasilnya 1 karena pi 3.14 lebih kecil dari 4 dan pi lebih besar dari 2.

```
>pi>5 || pi<1, pi>1 || pi<2
```

```
0
1
```

Selanjutnya || merupakan operator logika yang menyatakan "atau" yang akan bernilai salah apabila kedua pernyataannya salah dan akan bernilai benar apabila sekurang-kurangnya satu dari kedua pernyataan bernilai benar.

Boolean dengan Vektor

```
>(1:9)>pi, nonzeros(%)
```

```
[0,  0,  0,  1,  1,  1,  1,  1,  1]
[4,  5,  6,  7,  8,  9]
```

Kalau bandingkan dengan vektor ataumatriks, perbandingan akan berlaku untuk setap elemen. Pada contoh, elemennya dari 1 sampai 9 > pi (3.14) menghasilkan 0 untuk 1,2,3 dan 1 untuk 4,5,6,7,8,9.

Kalau ingin tahu elemen mana yang bernilai true, gunakan perintah nonzeros(%).

```
>N= 2 |3:20
```

```
[2,  3,  4,  5,  6,  7,  8,  9,  10,  11,  12,  13,  14,  15,  16,  17,
18,  19,  20]
```

Kita dapat menggunakan fungsi bukan nol untuk mengekstreak elemen tertentu dari bentuk vektor. Pada contoh, akan diekstrak bilangan 2 atau semua bilangan dari 3 sampai 20.

```
>N[nonzeros(isprime(N))]
```

```
[2,  3,  5,  7,  11,  13,  17,  19]
```

Lalu untuk menyaring N agar menyisakan bilangan prima, bisa menggunakan perintah [nonzeros (isprime(N))]

Format Keluaran

Format keluaran berarti komputer menyimpan angka dalam bentuk biner dengan standar sekitar 16 digit desimal namun ketika ditampilkan ke layar, default EMT dapat menampilkannya dengan 12 digit.

Default format

```
>defformat; pi
```

```
3.14159265359
```

defformat digunakan untuk mengatur ulang format ke output default yaitu EMT yang mencetak 12 digit.

longest format

```
>longest pi
```

```
      3.141592653589793
```

Jika ingin menampilkan menggunakan standar IEEE dengan sekitar 16 digit, dapat digunakan perintah longest atau format terpanjang untuk menampilkan hasil dengan format terpanjang.

Format Heksadesimal

```
>printhex(pi)
```

```
3.243F6A8885A30*16^0
```

printhex() merupakan cara komputer menyimpan pi yaitu dalam bentuk biner atau heksadesimal. Nilai pi bukan disimpan sebagai 3.1415.. namun dalam representasi biner.

Mengatur tampilan format

```
>format (12,4); 1/6, sin(45), pi
```

```
      0.1667
      0.8509
      3.1416
```

Format tampilan dapat diatur seperti contoh berikut. Perintah format(12,4) artinya 12= lebar total (soasi yang digunakan) dan 4=jumlah angka di belakang koma.

```
>format(12); 1/6
```

```
0.166666666667
```

Ini merupakan standar format yaitu kembali ke default 12 digit.

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

```
  0.66    0.2   0.89
  0.28   0.53   0.31
  0.44    0.3   0.28
```

membuat matriks acak 3x3 dengan angka antara 0 sampai 1 yaitu format default EMT. shortestformat disini memastikan agar EMT menampilkan dengan format sependek mungkin.

```
>shortformat; pi, 22/7
```

```
3.1416
3.1429
```

shortformat digunakan untuk menampilkan hasil dengan lebih ringkas namun tetap mempertahankan beberapa digit.

```
>longformat; pi, 22/7
```

```
3.14159265359
3.14285714286
```

longformat mirip default, yaitu menampilkan lebih banyak digit.

```
>longestformat; 1/7, pi
```

```
0.1428571428571428
3.141592653589793
```

longestformat digunakan untuk menampilkan semua 16 digit IEEE

```
>setscalarformat (4); pi
```

```
3.142
```

mengubah format skalar yang defaultnya 12 menjadi 4 sehingga hanya menampilkan 4 digit.

```
>goodformat(3); 22/7, pi
```

```
3.143
3.142
```

fungsi dari goodformat(length) adalah menampilkan angka dengan jumlah digit tertentu yang rapi dengan length adalah jumlah digit setelah koma.

Format Pecahan

```
>fracformat(10); 0.125
```

```
      1/8
```

fracformat(length) digunakan untuk menampilkan dalam bentuk pecahan rasional dengan length sebagai batas maksimum penyebut.

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

```
11/6
```

fraction merupakan format pecahan pada EMT. Pada contoh, EMT secara otomatis menjumlahkan ke dalam bentuk pecahan rasional.

Masalah angka desimal pada komputer

```
>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
```

komputer menyimpan angka dalam biner (basis 2) sehingga beberapa angka desimal yang terlihat sederhana tidak bisa ditulis dengan tepat dalam biner.

Jika pakai format biasa, hasil yang ditampilkan adalah 0 agar tampilan rapi dan tidak membingungkan.

```
>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, menggunakan tanda kurung setelah ekspresi. Jika menggunakan string sebagai ekspresi, gunakan konvensi untuk menamainya "fx" atau "fxy" dll. Ekspresi mengambil didahulukan dari fungsi.

Variabel global dapat digunakan dalam evaluasi.

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

```
        12.56637061435917
```

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

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

```
  -0.919535764538
```

Ekspresi akan selalu menggunakan variabel global, bahkan jika ada a variabel dalam fungsi dengan nama yang sama. (Jika tidak, evaluasi ekspresi dalam fungsi dapat memberikan hasil yang sangat membingungkan yang disebut fungsi.)

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

```
36
```

Jika ingin menggunakan nilai lain untuk "at" daripada nilai global yang diperlukan tambahkan "at=nilai".

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

```
45
```

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

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

```
45
```

Ekspresi dalam x sering digunakan seperti fungsi. Perhatikan bahwa mendefinisikan fungsi dengan nama yang sama seperti simbol global ekspresi menghapus variabel ini untuk menghindari kebingungan antara simbolik ekspresi dan fungsi.

```
>f &= 5*x;
>function f(x):= 6*x;
>f(2)
```

```
12
```

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", %(4,5)
```

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

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

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

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

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

```
-0.475
```

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

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

```
-0.425
```

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

Matematika Simbolik

EMT melakukan matematika simbolis dengan bantuan Maxima.Untuk detailnya, mulailah dengan mengikuti tutorial, atau telusuri referensi untuk Maxima. Pakar di Maxima harus dicatat bahwa ada perbedaan sintaks antara yang asli sintaks Maxima dan sintaks default ekspresi simbolik di EMT.
Matematika simbolik terintegrasi secara mulus ke dalam Euler dengan &. Ekspresi apa pun dimulai dengan & adalah ekspresi simbolis. Itu dievaluasi dan dicetak oleh Maksimal.
Pertama-tama, Maxima memiliki aritmatika "tak terbatas" yang dapat menangani angka yang sangat besar.

```
>$&44!
```

$$2658271574788448768043625811014615890319638528000000000$$

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

$$C(44,10) = \frac{44!}{34! \cdot 10!}$$

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

$$2481256778$$

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

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

$$2481256778$$

Untuk mempelajari lebih lanjut tentang fungsi tertentu, klik dua kali padanya. Misalnya, coba klik dua kali pada "&binomial" di baris perintah sebelumnya. Ini membuka dokumentasi Maxima sebagai disediakan oleh penulis program itu.
Anda akan mengetahui bahwa yang berikut ini juga berfungsi.

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

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

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

Jika Anda ingin mengganti x dengan nilai tertentu, gunakan ”with”.

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

$$120$$

Dengan begitu dapat menggunakan solusi persamaan dalam persamaan lain.
Ekspresi simbolik dicetak oleh Maxima dalam bentuk 2D. Alasan untuk ini adalah spesial bendera simbolis dalam string.
Seperti yang akan dilihat pada contoh sebelumnya dan berikut, jika telah menginstal Latex dapat mencetak ekspresi simbolis dengan Lateks. Jika tidak, perintah berikut akan mengeluarkan pesan kesalahan.
Untuk mencetak ekspresi simbolis dengan LaTeX, gunakan $ di depan & (atau dapat menghilangkan &) sebelum perintah. Jangan jalankan perintah Maxima dengan $, jika tidak memiliki LaTeX terpasang.

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

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

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

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

$$25$$

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

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

$$4\,\left(x+1\right)^3$$

Sekali lagi,% mengacu pada hasil sebelumnya. Untuk mempermudah kita menyimpan solusi kevariabel simbolik. Variabel simbolik didefinisikan dengan "&=".

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

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

Ekspresi simbolik dapat digunakan dalam ekspresi simbolik lainnya.

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

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

Input langsung dari perintah Maxima juga tersedia. Mulai baris perintah dengan "::". Sintaks Maxima disesuaikan dengan sintaks EMT (disebut "compatibilitymode").

```
>&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 seorang ahli dalam Maxima, mungkin ingin menggunakan sintaks asli dari Maxima. Dapat melakukannya dengan ":::".

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

```
 2
g
```

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

```
 3  x
x  E
```

$$x^3\,e^x$$

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

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

```
       5
125 E
```

$$125\,e^5$$

```
18551.64488782208
```

```
>fx(5)
```

```
18551.6448878
```

Untuk evaluasi ekspresi dengan nilai variabel tertentu, dapat menggunakan operator "with".
Baris perintah berikut juga menunjukkan bahwa Maxima dapat mengevaluasi dan 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))
```

$$x\,\left(x^2+6\,x+6\right)\,e^{x}$$

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

```
>tex(fx)
```

```
x^3\,e^{x}
```

Ekspresi simbolik dapat dievaluasi seperti ekspresi numerik.

```
>fx(0.5)
```

```
0.206090158838
```

Dalam ekspresi simbolik, ini tidak berfungsi, karena Maxima tidak mendukung dia. Sebagai gantinya, gunakan sintaks "with" (bentuk yang lebih bagus dari peerintah at(...) dari Maxima).

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

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

Tugas juga bisa simbolis.

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

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

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

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

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

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

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

```
1.56155281281
```

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

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

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

```
[-3.2360679775,  1.2360679775]
```

$$\left[x=-3.23606797749979 , x=1.23606797749979\right]$$

Untuk mendapatkan solusi simbolis teetentu, sesorang dapat menggunakan "with" dan indeks.

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

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

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

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

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

$$2$$

Ekspresi simbolik dapat memiliki tanda, yang menunjukkan perlakuan khusus dalam maxima. Beberapa flag dapar digunakan sebagai perintah juga, yang lain tidak. Flag adalah ditambahkan dengan "|" (bentuk yang lebih bagus dari "ev(...,flags)").

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

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

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

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

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

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

Kita bisa menggunakan kode angka untuk memunculkan suatu simbol huruf yunani.

Fungsi

Dalam EMT,fungsi adalah program yang didefinisikan dengan perintah "fungsi". Ini bisa menjadi fungsi satu baris atau fungsi multibaris. Fungsi satu baris dapat berupa numerik atau simbolis. Fungsi satu baris numerik adalah ditentukan oleh ":=".

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

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

```
>f(2)
```

```
4.472135955
```

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

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

```
[0,  0.100498756211,  0.203960780544,  0.313209195267,  0.430813184571,
0.559016994375,  0.699714227381,  0.854458893101,  1.02449987799,
1.21082616424,  1.41421356237]
```

Fungsi dapat diplot. Alih-alih ekspresi, kita hanya perlu menyediakan nama fungsi. Berbeda dengan ekspresi simbolik atau numerik, nama fungsi harus disediakan dalam sebuah string.

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

```
0.786151377757
```

Secara default, jika Anda perlu menimpa fungsi bawaan, Anda harus menambahkan kata kunci "timpa". Menimpa fungsi bawaan berbahaya dan dapat menyebabkan masalah untuk fungsi lain tergantung pada fungsi tersebut.
Anda masih dapat memanggil fungsi bawaan sebagai "_...", jika itu adalah fungsi di inti Euler

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

```
0.707106781187
```

Lebih baik kita menghapus definisi ulang sin ini.

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

```
0.707106781187
```

Parameter Default

Fungsi numerik dapat memiliki parameter default.

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

Menghilangkan parameter ini menggunakan nilai default.

```
>f(3)
```

```
18
```

Menyetelnya akan menimpa nilai default

```
>f(3,4)
```

```
36
```

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

```
>f(3,a=2)
```

```
18
```

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

```
>function f(x) := a*x^2
>a=5; f(3)
```

```
45
```

Tetapi parameter yang ditetapkan menimpa nilai global.

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

```
>f(3,a:=4)
```

```
36
```

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

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

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

Symbolic functions can be used in symbolic expressions.

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

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

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

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

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

```
178.635099908
```

Mereka dapat digunakan untuk mendefinisikan fungsi atau ekspresi simbolis lainnya.

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

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

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

```
0.703467422498
```

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

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

```
0.703467422498
```

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

$$\left(2\,x-1\right)^{n}$$

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

$$(x+2)^n$$

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

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

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

625

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

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

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

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

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

$$\left(x+2\right)^3\,\left(2\,x-1\right)^4$$

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

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

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

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

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

$$x^3 - x$$

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

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

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

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

lateks: f(x) = \int_1^x t^t \, dt,

yang tidak dapat dinilai secara simbolis.

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

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

```
[-0.783430510712,  -0.410815648254,  0,  0.676863278799,  2.05044623453]
```

Fungsi dapat memiliki nilai default untuk parameter.

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

Now the function can be called with or without a parameter ”base”.

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

```
2
6.7
```

Selain itu, dimungkinkan untuk menggunakan parameter yang ditetapkan.

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

```
2
```

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

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

$$y^2 - x\,y + y + x^2$$

Fungsi simbolik seperti itu dapat digunakan untuk variabel simbolik.

Tetapi fungsi tersebut 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+I*y)^4), $&lapl(%,x,y)
```

$$0$$

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

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

$$10\left(y^2+x\right)^3\left(9\,y^2+x+2\right)$$

Untuk meringkas

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

Memecahkan

Ekspresi

Ekspresi dapat diselesaikan secara numerik dan simbolis.

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

```
>solve("x^3-5",1)
```

```
1.70997594668
```

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

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

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

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

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

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

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

$$\left[\left[x=\frac{b\,f-c\,e}{b\,d-a\,e},y=\frac{c\,d-a\,f}{b\,d-a\,e}\right]\right]$$

```
>px &= 3*x^6+x^3+2^x-5; $&px
```

$$2^{x}+3\,x^6+x^3-5$$

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=4), px(%)
```

```
1.10610336809
4
```

```
>$&solve(x^3=5,x)
```

$$\left[x=\frac{\sqrt{3}\,5^{\frac{1}{3}}\,i-5^{\frac{1}{3}}}{2} , x=\frac{-\sqrt{3}\,5^{\frac{1}{3}}\,i-5^{\frac{1}{3}}}{2} , x=5^{\frac{1}{3}}\right]$$

```
>$&solve(x^3-5,x)
```

$$\left[x=\frac{\sqrt{3}\,5^{\frac{1}{3}}\,i-5^{\frac{1}{3}}}{2} , x=\frac{-\sqrt{3}\,5^{\frac{1}{3}}\,i-5^{\frac{1}{3}}}{2} , x=5^{\frac{1}{3}}\right]$$

```
>$&solve(a*x^3+b*x+c=0,x)
```

$$\left[x=\left(-\frac{\sqrt{3}\,i}{2}-\frac{1}{2}\right)\,\left(\frac{\sqrt{\frac{27\,a\,c^2+4\,b^3}{a}}}{2\,3^{\frac{3}{2}}\,a}-\frac{c}{2\,a}\right)^{\frac{1}{3}}-\frac{\left(\frac{\sqrt{3}\,i}{2}-\frac{1}{2}\right)\,b}{3\,a\,\left(\frac{\sqrt{\frac{27\,a\,c^2+4\,b^3}{a}}}{2\,3^{\frac{3}{2}}\,a}-\frac{c}{2\,a}\right)^{\frac{1}{3}}},x=\left(\frac{\sqrt{3}\,i}{2}-\frac{1}{2}\right)\,\left(\frac{\sqrt{\frac{27\,a\,c^2+4\,b^3}{a}}}{2\,3^{\frac{3}{2}}\,a}-\frac{c}{2\,a}\right)^{\frac{1}{3}}-\right.$$

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

$$\left[\left[x=\frac{b\,f-c\,e}{b\,d-a\,e},y=\frac{c\,d-a\,f}{b\,d-a\,e}\right]\right]$$

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

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

Sekarang kita mencari titik, di mana polinomialnya adalah 2. Dalam solve(), nilai target default y=0 dapat diubah dengan variabel yang ditetapkan.
Kami menggunakan y=2 dan memeriksa dengan mengevaluasi polinomial pada hasil sebelumnya.

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

```
0.966715594851
2
```

.

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

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

The easiest way to get the numerical values is to evaluate the solution numerically just like an expression.

```
>longest sol()
```

```
-0.6180339887498949       1.618033988749895
```

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

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

$$0$$

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

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

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

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

lateks: aˆx-xˆa = 0.1

dengan a=3.

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

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

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

```
2.54116291558
```

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

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

```
2.54116291558
```

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

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

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

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

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

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

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

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

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

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

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

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

$$emptyset$$

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

$$universalset$$

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

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

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

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

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

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

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

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

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

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

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

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

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

```
        [6 < x, x < 8, y < - 11] or [8 < x, y < - 11]
 or [x < 8, 13 < y] or [x = y, 13 < y] or [8 < x, x < y, 13 < y]
 or [y < x, 13 < y]
```

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

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

Bahasa Matriks

Dokumentasi inti EMT berisi diskusi terperinci tentang bahasa matriks Euler.
Vektor dan matriks dimasukkan dengan tanda kurung diku, elemen dipindahkan oleh koma, baris dipisahkan denan titik koma.

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

```
              1               2
              3               4
```

Produk matriks dilambangkan dengan titik.

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

```
              3
              4
```

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

```
  [3,  4]
```

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

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

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

```
          11
          25
```

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

```
           1             0
           0             1
```

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

```
>A.A
```

```
           7            10
          15            22
```

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

```
             1             4
             9            16
```

Hasil yang ditunjukkan merupakan hasil perpangkatan elemen per elemen.

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

```
            37            54
            81           118
```

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

```
            37            54
            81           118
```

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

```
             1             1
             1             1
```

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

```
        0.333333      0.666667
            0.75             1
```

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

```
              -2
             2.5
```

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

```
              -2
             2.5
```

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

```
               1             0
               0             1
```

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

```
            1             0
            0             1
```

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

```
            1             4
            9            16
```

Ini bukan produk matriks, tetapi sebuah perkalian elemen demi elemen.
Sama halnya bekerja dengan vektor.

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

```
            9
           16
```

Jika salah satu operan yang dihitung berupa vektor atau skalar, maka nilainya akan diperluas secara otomatis sesuai cara yang alami.

```
>2*A
```

```
            2             4
            6             8
```

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

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

```
            1             4
            3             8
```

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

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

```
            2             6
            6            12
```

Satu bisa membayangkan perkalian ini seolah-olah vektor baris v telah digandakan untuk membentuk sebuah matriks dengan ukuran sama seperti A.

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

```
              1             2
              1             2
```

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

```
              1             4
              3             8
```

Ini juga berlaku untuk dua vektor di mana salah satunya adalah vektor baris dan lainnya adalah vektor kolom. Kami menghitung i*j untuk i,j dari 1 hingga 5. Triknya adalah mengalikan 1:5 dengan transposnya. Bahasa mattriks 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 < 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 dpat menghitung jumlah elemen yang memnuhi syarat tertentu kondisi 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, "nonzeros" memilih elemen yang bukan nol.
Dalam hal ini,kita mendapatkan indeks dari semua elemen yang 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 yang sesuai nilai 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, yang adalah 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 kotak ditambah 1 adalah bilangan prima.

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

```
 112
```

Fungsi nonzeros() bekerja hanya untuk vektor. Untuk matriks, digunakan fungsi 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 bisa 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 itu mungkin untuk mendapatkan elemen dalam sebuah vektor.

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

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

Fungsi lain yang berguna adalah extrema, yang mengembalikan nilai minimum dan nilai maksimum 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 dapt mengekstrak indek 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 sebuah matriks, kita dapat menumpuk satu matriks di atas matriks lainnya. Jika keduanya tidak memiliki jumlahkolom 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 akan diisi dengan 0. Ada pengecualian untuk aturan ini. Bilangan real yang melekat 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

It is possible to make a matrix of row and column vectors.

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

1	2	3
1	2	3

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

1	1
2	2
3	3

Tujuan utama dari ini adalah untuk interpretasi vektor dari ekpresi per kolom vektor.

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

```
              1             1
              2             4
              3             9
```

Untuk mendapatkan ukuran dari A, kita bisa menggunakan fungsi yang tertera.

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

```
  2
  4
  [2,  4]
  4
```

Untuk vektor, terdapat length().

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

```
  9
```

Ada banyak fungsi lain yang menghasilkan matriks.

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

```
              1               1
              1               1
```

Ini juga dapat digunakan dengan parameter. Untuk mendapatkan sebuah vektor dengan bilangan selain 1, gunakan fungsi tertera.

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

```
  [6,  6,  6,  6,  6]
```

Matriks dnegan bilangan acak juga dapat dimunculkan dengan acak (uniform distribution) atau normal (Gauß distribution).

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

```
        0.66566        0.831835
          0.977        0.544258
```

Berikut ini juga fungsi lain yang berguna, yang merestrukturisasi elemen sebuah matriks ke 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 yang tertera, kita bisa menggunakan ini dan fungsi dup untuk menulis sebuah fungsi rep(), yang mengulang sebuah vektor sebanyak n.

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

Mari kita uji.

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

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

Fungsi multdup() menduplikasi elemen sebuah 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() membalikkan urutan baris atau kolom matriks. Yaitu, fumgsi 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 mneghilangkan elemen dengan indeks di i dari vektor v.

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

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

Perhatikan bahwa vektor i di drop(v,i) megacu pada indeks elemen di v, bukan nilai elemen. Jika anda ingin meghapus elemen, anda perlu menemukan elemennya terlebih dahulu. Fungsi indexof(v,x) dapat digunakan untuk menemukan elemen x dalam vektor yang diurutkan 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 dapat dilihat, tidak ada salahnya memasukkan indeks di luar jangkauan (seperti 0), indeks ganda, atau indeks yang tidak disortir.

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

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

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

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

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

Lalu kita atir 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 kita tidak mengubah matriks A.Kita mendapatkan matriks baru sebagai hasil dari setdiag().

Berikut adalah fungsi yang mengembalikan matriks tri-diagonal.

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

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

Diagonal suatu matriks juga dapat diekstraksi dari matriks tersebut. Untuk mendemonstrasikan ini, kita merestrukturasi vektor 1:9 ke matriks 3x3.

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

1	2	3
4	5	6
7	8	9

Sekarang kita dapat mengekstrak diagonalnya.

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

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

Misalnya kita dapat membagi matriks dengan diagonalnya sendiri. Bahasa Matriks memastikan bhwa vektor kolom d diterapkanke baris matriks dengan baris.

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

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

Hampir semua fungsi di euler juga berfungso untuk input matriks dan vektor kapanpun ini masuk akal. Misalnya, fungsi sqrt() menghitung akar kuadrat dari semua elemen dari vektor atau matriks.

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

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

Jadi anda bisa dengan mudah membuat tabel ini. Ini adalah satu cara untuk merencanakan function (alternatifnya menggunakan ekspresi)

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

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

Dalam contoh berikut, kita membangkitkan sebuah vektor nilai t[i] dengan spasi 0.1 dari -1 ke 1. Kemudian kita membangkitkan sebuah vektor dari nilai-nilai fungsi

$$s = t^3 - t$$

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

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

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

Misalnya, jika sebuah vektor diterapkan, vektor kolom dikalikan dengan vektor baris menjadi matriks. 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 snagat berbeda dari produk matriks. Matriks produk dilambangkan dengan titik "." di EMT.

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

```
55
```

Secara default, vektor baris dicetak dalam format ringkas.

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

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

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

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

```
5
25
```

Untuk mentranspos kita gunakan 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 vv'

```
>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 sebuah vektor 1x1, yang bekerja seperti bilangan real.

```
>v.v'
```

```
30
```

Ada juga norma fungsi (bersama dengan banyak fungsi lainnya dari Aljabar Linier).

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

```
30
```

Operator dan fungsi mematuhi bahasa matriks Euler.

Berikut adalah ringkasan aturannya.
-Fungsi yang diterapkan ke vektor atau matriks diterapkan ke setiap elemen.
-Operator yang beroperasi pada dua matriks dengan ukuran yang sama diterapkan secara berpasangan ke elemen matriks.
-Jika kedua matriks memiliki dimensi yang berbeda, keduanya diperluas dalam acara yang masuk akal, sehingga mereka memiliki ukuran yang sama.

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

Berikut ini adalah kasus sederhana dengan operator^.

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

```
[1,  4,  9]
```

Ini adalah kasus yang lebih rumit. Vektor baris dikali vektor kolom mengembang keduanya dengan menduplikasi.

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

```
            1             2             3
            2             4             6
            3             6             9
```

Perhatikan bahwa produk skalar menggunakan produk matriks, bukan*!

```
>v.v'
```

```
14
```

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

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

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

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

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

```
>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
```

Elemen 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 dapat berupa vektor baris dari indeks. : menunjukkan semua indeks.

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

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

Bentuk singkat untuk: adalah menghilangkan indeks sepenuhnya.

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

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

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

```
>A{4}
```

```
4
```

Sebuah matriks juga dapat diratakan, menggunakan fungsi redim(). Ini adalah 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
```

Menggunakan bahasa matriks, kita dapat menghasilkan beberapa tabel dari beberapa berfungsi sekaligus.

Dalam contoh berikut, kita menghitung t[j]ˆi untuk i dari 1 hingga n. Kami mendapatkan matriks, di mana setiap baris adalah tabel tˆi untuk satu i. Yaitu, matriks memiliki elemen

$$a_{i,j} = t_j^i, \quad 1 \leq j \leq 101, \quad 1 \leq i \leq n$$

Sebuah fungsi yang tidak bekerja untuk input vektor harus ”vectorized”. Ini dapat dicapai dengan kata kunci ”map” dalam definisi fungsi. Kemudian fungsi akan dievaluasi untuk setiap elemen dari parameter vektor.
Integrasi numerik integrate() 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 angka.

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

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

Untuk mengakses elemen matriks, gunakan notasi braket.

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

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

Kita dapat mengakses baris matriks yang lengkap.

```
>A[2]
```

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

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

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

```
 2
```

Untuk memastikan, Anda mendapatkan baris pertama untuk matriks 1xn dan mxn, menentukan semua kolom menggunakan indeks kedua yang kosong.

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

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

Jika indeks adalah vektor indeks, Euler akan mengembalikan baris matriks yang sesuai.
Di sini kita menginginkan baris pertama dan kedua dari A

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

```
             1              2              3
             4              5              6
```

Kita bahkan dapat menyusun ulang A menggunakan vektor indeks. Lebih tepatnya, kami melakukannya 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 sub matriks A ke beberapa nilai. Ini sebenarnya mengubah matriks A yang tersimpan.

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

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

Kita juga dapat menetapkan nilai ke baris A

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

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

Kita 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 kesalahan pesan, tergantung pada pengaturan sistem. Standarnya adalah kesalahan pesan. Ingat, bagaimanapun, bahwa indeks negatif dapat digunakan untuk mengakses elemen matriks yang dihitung dari akhir

```
>A[4]
```

```
A is not a variable!
Error in:
A[4] ...
    ^
```

Penyortiran dan Pengacakan

Fungsi sort() mengurutkan vektor baris.

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

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

Seringkali kita perlu mengetahui indeks dari vektor yang diurutkan dalam vektor asli. Ini dapat digunakan untuk menyusun ulang vektor lain di cara yang sama.

Mari kita mengocok sebuah vektor.

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

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

Indeks berisi urutan yang tepat dari v.

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

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

Ini juga berfungsi untuk vektor string.

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

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

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

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

As you see, the position of double entries is somewhat random.

```
>ind
```

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

The function unique returns a sorted list of unique elements of a vector.

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

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

This works for string vectors too.

```
>unique(s)
```

```
a
aa
d
e
```

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

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

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

```
             -4
            4.5
```

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

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

```
    8.790745908981989e-13
```

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

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

1	2	3
4	5	6
7	8	9

Determinan matriks ini adalah 0.

```
>det(A)
```

0

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

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

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

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

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

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

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

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

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

Seperti semua variabel simbolik, matriks ini dapat digunakan di lain ekspresi simbolis.

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

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

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

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

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

$$\left[\left[a-1 , a+1\right] , \left[1 , 1\right]\right]$$

Untuk mengekstrak vektor eigen tertentu perlu pengindeksan yang cermat.

```
>$&eigenvectors([a,1;1,a]), &%[2][1][1]
```

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

```
[1, - 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.

```
>$&A with [a=4,b=5]
```

$$\begin{pmatrix}1 & 4\\ 5 & 2\end{pmatrix}$$

Akses ke baris matriks simbolik berfungsi seperti numerik matriks.

```
>$&A[1]
```

$$[1, a]$$

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

```
>&A[1,1]:=t+1; $&A
```

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

Ada fungsi simbolis di Maxima untuk membuat vektor dan matriks. Untuk ini, lihat dokumentasi Maxima atau ke tutorial tentang Maxima di EMT.

```
>v &= makelist(1/(i+j),i,1,3); $v
```

$$\left[\frac{1}{j+1}, \frac{1}{j+2}, \frac{1}{j+3}\right]$$

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

$$\begin{pmatrix}-2 & 1 \\ \frac{3}{2} & -\frac{1}{2}\end{pmatrix}$$

$$\begin{pmatrix}-2 & 1 \\ \frac{3}{2} & -\frac{1}{2}\end{pmatrix}$$

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

```
>$&invert(B)()
```

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

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

Perhatikan, bahwa dengan &:= matriks B telah didefinisikan sebagai simbolik dalam ekspresi simbolik dan sebagai numerik dalam ekspresi numerik. Jadi kita dapat menggunakannya di sini.

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

```
                      1                         0
                      0                         1
```

E.g. the eigenvalues of A can be computed numerically.

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

```
  [16.1168,  -1.11684,  0]
```

Misalnya nilai eigen dari A dapat dihitung secara numerik.

```
>$&eigenvalues(@A)
```

$$\left[\left[\frac{3-\sqrt{4\,a\,b+1}}{2} , \frac{\sqrt{4\,a\,b+1}+3}{2} \right] , \left[1 , 1 \right] \right]$$

Nilai Numerik dalam Ekspresi Simbolis

Ekspresi simbolis hanyalah string yang berisi ekspresi. Jika kita ingin mendefinisikan nilai baik untuk ekspresi simbolik maupun numerik ekspresi, kita harus menggunakan "&:=".

```
>A &:= [1,pi;4,5]
```

```
           1        3.14159
           4              5
```

Masih ada perbedaan antara numerik dan simbolik membentuk. Saat mentransfer matriks ke bentuk simbolis, pecahan perkiraan untuk real akan digunakan

```
>$&A
```

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

Untuk menghindari ini, ada fungsi "mxmset(variable)".

```
>mxmset(A); $&A
```

matrix([1,a],[b,2])

Maxima juga dapat menghitung dengan angka floating point, dan bahkan dengan besar angka mengambang dengan 32 digit. Evaluasi jauh lebih lambat, namun.

```
>$&bfloat(sqrt(2)), $&float(sqrt(2))
```

$$1.414213562373095$$

Ketepatan angka floating point besar dapat diubah.

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

```
        3.14159265358979323846264338327950288419716939937510582097494\
45923078164062862089986280348253421170 68b0
```

Variabel numerik dapat digunakan dalam ekspresi simbolis apa pun menggunakan "@var".

Perhatikan bahwa ini hanya diperlukan, jika variabel telah didefinisikan dengan ":=" or "=" 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% per tahun. Mari kita tambahkan satu tarif sederhana dan menghi
tung hasilnya.

```
>K*1.03
```

```
5150
```

Euler juga dapat memahami sintaks berikut.

```
>K+K*3%
```

```
5150
```

`Tetapi lebih mudah menggunakan faktornya`

```
>q=1+3%, K*q
```

```
1.03
5150
```

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

```
>K*q^10
```

```
6719.58189672
```

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

```
>format(12,2); K*q^10
```

```
      6719.58
```

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

```
>"Starting from " + K + "$ you get " + round(K*q^10,2) + "$."
```

```
  Starting from 5000$ you get 6719.58$.
```

Bagaimana jika kita ingin mengetahui hasil antara dari tahun 1 sampai tahun 9? Untuk ini, bahasa matriks Euler sangat membantu. Anda tidak perlu menulis loop, tetapi cukup masukkan

```
>K*q^(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 ke
```

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. Ini dikalikan dengan K, dan kita mendapatkan vektor nilai.

```
>VK=K*q^(0:10);
```

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

```
>function oneyear (K) := round(K*q,2)
```

```
Mari kita bandingkan kedua hasil, dengan dan tanpa pembulatan.
```

```
>longest oneyear(1234.57), longest 1234.57*q
```

```
           1271.61 
          1271.6071 
```

Sekarang tidak ada rumus sederhana untuk tahun ke-n, dan kita harus mengulang bertahun-tahun. Euler memberikan banyak solusi untuk ini.

Cara termudah adalah iterasi fungsi, yang mengulangi fungsi yang diberikan a berkali-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 desimal tetap tempat.

```
>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 elementer tentu 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 dapat menggunakan vektor indeks. Ingat bahwa 1:3 diproduksi 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 sanggat kecil.

Sekarang kita mengambil fungsi yang lebih maju, yang masing-masing menambahkan tingkat ulang tertentu tahun.

```
>function onepay (K) := K*q+R
```

Kita tidak perlu menentukan q atau R untuk definisi fungsi. Hanya jika kita menjalankan perintah, kita harus mendefinisikan nilai-nilai ini. Kami memilih R=200.

```
>R=200; iterate("onepay",5000,10)
```

```
Real 1 x 11 matrix

    5000.00     5350.00     5710.50     6081.82     ...
```

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, kami kehilangan uang setiap tahun.

Bagaimana kita bisa menentukan berapa tahun uang itu akan bertahan? Kami akan memiliki untuk 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     ...
```

Menggunakan bahasa matriks, kita dapat menentukan nilai negatif pertama dalam mengikuti cara.

```
>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. Ini dapat 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 nilai adalah 0 setelah 50 tahun. Apa yang akan menjadi tingkat bunga?

Ini adalah pertanyaan, yang hanya dapat dijawab secara numerik. Di bawah ini, kami akan mendapatkan formula yang diperlukan. Maka Anda akan melihat bahwa tidak ada yang mudah formula untuk tingkat bunga. Tapi untuk saat ini, kami bertujuan untuk solusi numerik.

Langkah pertama adalah mendefinisikan sebuah fungsi yang melakukan iterasi sebanyak n kali. Kita tambahkan semua parameter ke fungsi ini.

```
>function f(K,R,P,n) := iterate("x*(1+P/100)+R",K,n;P,R)[-1]
```

Iterasinya seperti diatas

$$x_{n+1} = x_n \cdot \left(1 + \frac{P}{100}\right) + R$$

Tapi kita tidak lagi menggunakan nilai global R dalam ekspresi kita. Fungsi seperti: iterate() memiliki trik khusus di Euler. Anda dapat meneruskan nilai-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 indeksnya [-1].

Mari kita coba tes

```
>f(5000,-200,3,47)
```

```
-19.83
```

Sekarang kita bisa memecahkan masalah kita.

```
>solve("f(5000,-200,x,50)",3)
```

```
3.15
```

Rutin pemecahan memecahkan ekspresi=0 untuk variabel x. Jawabannya adalah 3,15% per tahun. Kami mengambil nilai awal 3% untuk algoritma. Pemecahan() fungsi selalu membutuhkan nilai awal.

Kita dapat menggunakan fungsi yang sama untuk menyelesaikan pertanyaan berikut: Berapa banyak yang dapat kita? hapus per tahun sehingga modal benih habis setelah 20 tahun dengan asumsi tingkat bunga 3% per tahun.

```
>solve("f(5000,x,3,20)",-200)
```

```
-336.08
```

Perhatikan bahwa Anda tidak dapat memecahkan jumlah tahun, karena fungsi kami mengasumsikan n sebagai nilai integer.

Solusi Simbolik untuk Masalah Suku Bunga

Kita dapat menggunakan bagian simbolik dari Euler untuk mempelajari masalah. Pertama kita definisikan fungsi kita onepay() secara simbolis.

```
>function op(K) &= K*q+R; $&op(K)
```

Sekarang kita dapat mengulangi ini.

```
>$&expand(%)
```

$$\frac{100\,\left(\frac{P}{100}+1\right)^{n}\,R}{P}-\frac{100\,R}{P}+K\,\left(\frac{P}{100}+1\right)^{n}$$

Kita melihat sebuah pola. Setelah n periode yang kita miliki

$$K_n = q^n K + R(1+q+\ldots+q^{n-1}) = q^n K + \frac{q^n-1}{q-1} R$$

Rumusnya adalah rumus untuk jumlah geometri, yang diketahui Maxima.

```
>&sum(q^k,k,0,n-1); $& % = ev(%,simpsum)
```

$$\sum_{k=0}^{n-1}{q^{k}}=\frac{q^{n}-1}{q-1}$$

Ini agak rumit. Jumlahnya dievaluasi dengan bendera ”simpsum” untuk dikurangi ke hasil bagi.

Mari kita membuat fungsi untuk ini.

```
>function fs(K,R,P,n) &= (1+P/100)^n*K + ((1+P/100)^n-1)/(P/100)*R; $&fs(K,R,P,n)
```

$$\frac{100\,\left(\left(\frac{P}{100}+1\right)^{n}-1\right)\,R}{P}+K\,\left(\frac{P}{100}+1\right)^{n}$$

Fungsi melakukan hal yang sama seperti fungsi f kita sebelumnya. Tapi itu lebih efektif.

```
>longest f([5000, -200, 3, 47]); f([5000, -330, 3, x])
```

```
Function f needs at least 2 arguments!
Use: f (x, a)
Error in:
longest f([5000, -200, 3, 47]); f([5000, -330, 3, x]) ...
                              ^
```

```
>longest fs(5000, -200, 3, 47)
```

```
      -19.82504734652684
```

Sekarang kita dapat menggunakannya untuk menanyakan waktu n. Kapan modal kita habis? Kita tebakan awal 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 rumus untuk pembayaran.

Asumsikan kita mendapatkan pinjaman sebesar K, dan membayar n pembayaran sebesar R (dimulai setelah yang pertama tahun) menyisakan sisa hutang sebesar Kn (pada saat pembayaran terakhir). Itu rumus untuk ini jelas

```
>equ &= fs(K,R,P,n)=Kn; $&equ
```

$$fs\left(K, R, P, n\right) = Kn$$

Biasanya rumus ini diberikan dalam bentuk

$$i = \frac{P}{100}$$

```
>equ &= (equ with P=100*i); $&equ
```

$$fs\left(K , R , 100\,i , n\right)=Kn$$

Kita dapat memecahkan tingkat R secara simbolis.

```
>$&solve(equ,R)
```

$$\left[fs\left(K , R , 100\,i , n\right)=Kn \right]$$

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

Tentu saja, kami memiliki batas berikut.

```
>$&limit(R(5000,0,x,10),x,0)
```

$$\lim_{x\to 0} R\left(5000, 0, x, 10\right)$$

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

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

```
>fn &= solve(equ,n) | ratsimp; $&fn
```

$$\left[fs\left(K, R, 100\,i, n\right) = K n\right]$$

LATIHAN SOAL

1. Menyederhanakan

$$m^{-5} \times m^{5}$$

$$(y-1)^{-1}(y-1)^{5}$$

```
>$&m^(-5)*m^5
```

$$1$$

```
>$&(y-1)^(-1)*(y-1)^5
```

$$(y-1)^{4}$$

2. Mengubah fungsi desimal

$$7.6 \times 10^{5}$$

$$1.09 \times 10^{-7}$$

```
>7.6*10^5
```

```
760000
```

```
>1.09*10^(-7)
```

```
1.09e-07
```

3. Persamaan kuadrat dengan akar imajiner.
Persamaan kudrat

$$ax^2 + bx + c = 0$$

memiliki solusi

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Jika diksriminan kurang dari 0 maka akarnya berupa bilangan imajiner.

soal: cari akar-akar dari persamaan kuadrat

$$2x^2 + 3x + 7 = 0$$

lainnya,

$$3x^3 + 6x^2 - 27x - 54 = 0$$

```
>$&solve(2*x^2+3*x+7=0,x)
```

$$\left[x = \frac{-\sqrt{47}\,i - 3}{4}, x = \frac{\sqrt{47}\,i - 3}{4}\right]$$

```
>$&solve(3*x^3+6*x^2-27*x-54=0,x)
```

$$[x = -3, x = -2, x = 3]$$

$$\lim_{x \to 0} \frac{\sin x}{x}$$

```
>$&limit(sin(x)/x,x,0)
```

4. Limit

$$\lim_{x\to 0} \frac{\cos x}{2x-1}$$

```
>$&limit(cos(x)/2*x-1,x,0)
```

5. Turunan dan trigonometri
Penyelesaiian soal ini menggunakan aturan rantai:
jika f(x)=sin(g(x)) maka f'(x)= g'(x)cos(g(x))

$$f(x) = sin(x^2)$$

```
>$&diff(sin(x^2),x)
```

$$2\,x\,\cos x^2$$

6. Turunan dan fungsi trigonometri
latex f(x)=2sin(2x)

```
>$&diff(2*sin(2*x),x)
```

$$4\,\cos\left(2\,x\right)$$

7. Determinan dan Invers Matriks

$$M = \begin{bmatrix} 4 & 3 \\ 2 & 5 \end{bmatrix}$$

```
>M:= [4,3; 2,5] //mendefinisikan matriks
```

```
            4             3
            2             5
```

```
>det(M)
```

```
 14
```

Determinan matriks 2x2

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

dicari menggunakan det() di EMT atau manualnya dengan:

$$ac - bd$$