Nama: Rahmita
NIM:23030630015
Kelas: Matematika B 2023

EMT untuk Perhitungan Aljabar

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

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

Contoh pertama

Menyederhanakan bentuk aljabar:

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

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

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

Menjabarkan:

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

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

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

Latihan membuat contoh soal dan penyelesainnya
latex:(3x^(-2)+5y^4)(-4x^3-2y^(-6))

```
>$&showev('expand((3*x^(-2)+5*y^4)*(-4*x^3-2*y^(-6))))
```

$$expand\left(\left(-\frac{2}{y^6}-4\,x^3\right)\left(5\,y^4+\frac{3}{x^2}\right)\right)=-20\,x^3\,y^4-\frac{10}{y^2}-\frac{6}{x^2\,y^6}-12\,x$$

Latihan membuat contoh soal dan penyelesainnya
latex:(8x^(-1)-6y^3)(5x^2+y^(-4))

```
>$&showev('expand((8*x^(-1)-6*y^3)*(5*x^2+y^(-4))))
```

$$expand\left(\left(\frac{1}{y^4}+5\,x^2\right)\,\left(\frac{8}{x}-6\,y^3\right)\right)=-30\,x^2\,y^3-\frac{6}{y}+\frac{8}{x\,y^4}+40\,x$$

Latihan membuat contoh soal dan penyelesainnya
latex:(2x^(-5)+7y^2)(-9x^3-y^(-7))

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

$$expand\left(\left(-\frac{1}{y^7}-9\,x^3\right)\,\left(7\,y^2+\frac{2}{x^5}\right)\right)=-63\,x^3\,y^2-\frac{7}{y^5}-\frac{2}{x^5\,y^7}-\frac{18}{x^2}$$

Latihan membuat contoh soal dan penyelesainnya
latex:5x(3x-4)+2x(3x+5)

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

$$2\,x\,(3\,x+5)+5\,x\,\,(3\,x-4)$$

Latihan membuat contoh soal dan penyelesainnya
latex:(2x+3y)^2

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

$$(3\,y+2\,x)^2$$

Baris Perintah

Baris perintah Euler terdiri dari satu atau beberapa perintah Euler diikuti dengan titik koma ";" atau koma ",". Titik koma mencegah pencetakan hasilnya. 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, %+2*pi*r*h // Ingat tanda % menyatakan hasil perhitungan terakhir sebelumnya
```

```
  50.2654824574
  100.530964915
```

Baris perintah dijalankan sesuai urutan yang ditekan pengguna kembali. Jadi, Anda mendapatkan nilai baru setiap kali Anda menjalankan baris kedua.

```
>x := 1;
>x := cos(x) // nilai cosinus (x dalam radian)
```

```
  0.540302305868
```

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

```
  0.857553215846
```

jika dua garis dihubungkan dengan "..." kedua garis akan selalu dijalankan secara bersamaan.

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

```
  1.41666666667
  1.41421568627
  1.41421356237
```

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

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

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

A line starting with %% will be completely invisible.

```
  81
```

Euler mendukung loop di baris perintah, asalkan cocok ke dalam satu baris atau multi-baris. Tentu saja, pembatasan ini tidak berlaku dalam program. Untuk informasi lebih lanjut lihat pendahuluan 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 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!
```

Kupikir begitu!
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. Di baris kosong, bantuan untuk bantuan jendela akan di tampilkan. 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. Mencobamengklik dua kali perintah exp di bawah pada baris perintah.
>exp(log(2.5))

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

```
  2.5
```

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

latihan membuat contoh soal dan penyelesaiannya
r:=3;h:=7;pi*r^2*h

```
>r:=3; h:=7; V:=pi*r^2*h
```

```
  197.920337176
```

latihan membuat contoh soal dan penyelesaiannya
r:=3; h:=7; V:=pi*r^2*h; A:=2*pi*r*h; V,A

```
>pi*r^2*h, %+2*pi*r*h
```

```
  197.920337176
  329.867228627
```

latihan membuat contoh soal dan penyelesaiannya
x:=1;
x:=sin(x)// nilai sinus (x dalam radian)

```
>x:=1; x:=sin(x)//nilai sinus(x dalam radian)
```

```
  0.841470984808
```

latihan membuat contoh soal dan penyelesaiannya
x:=1, 5; a:=5; x:=(x+a/x)/2

```
>x:=1, 5; a:=5; x:=(x+a/x)/2, x:=(x+a/x)/2, x:=(x+a/x)/2
```

```
  1
  3
  2.33333333333
  2.2380952381
```

latihan membuat contoh soal dan penyelesaiannya
x:=3*7; x:=x*2

```
>x:=3*7; x:=x*2
```

```
  42
```

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

Untuk menyetel variabel, gunakan "=" atau ":=". Demi kejelasan, pendahuluan ini menggunakan bentuk yang terakhir. Spasi tidak penting. Tapi jarak antar perintah diharapkan.

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

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

```
  30.65625
```

EMT menggunakan sintaks pemrograman untuk ekspresi. Untuk masuk

lateks: e^2 \cdot \frac(\frac{1}{3+4 \log(0.6)}+\frac{1}{7} \kanan)

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

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

```
8.77908249441
```

Untuk menghitung ekspresi rumit seperti

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

Anda harus memasukkannya dalam formulir baris.

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

```
23.2671801626
```

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

Hasil perhitungan ini berupa bilangan floating point. Ini secara default dicetak dengan akurasi sekitar 12 digit. Di baris perintah berikut, kita juga mempelajari 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, harus berisi tanda kurung untuk memaksakan urutan eksekusi yang benar. Jika ragu, memasang braket adalah ide yang bagus. Perhatikan bahwa 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 Euler meliputi

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

dan masih banyak lagi.

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

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

Beberapa perintah memiliki alias, mis. Untuk log.

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

```
  2
  0.5
```

```
>sin(30°)
```

```
  0.5
```

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

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

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

Latihan membuat contoh soal dan penyelesaiannya
a:=3; b:=4; 1/2*a*b^2

```
>a:=3; b:=4; 1/2*a*b^2
```

```
  24
```

Latihan membuat contoh soal dan penyelesaiannya
cos60*pi/180), tan (45*pi/180)

```
>cos (60* pi/180), tan(45*pi/180)
```

```
  0.5
  1
```

Latihan membuat contoh soal dan penyelesaiannya
4^2^3, (4^2)^3, 4^(2^3)

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

```
  65536
  4096
  65536
```

Latihan membuat contoh soal dan penyelesaiannya
(cos(pi/3)+1)^2*(sin(pi/3)+1)^3

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

```
  14.6195893444
```

Latihan membuat contoh soal dan penyelesaiannya
1/5+1/8, fraction %

```
>1/5+1/8, fraction %
```

```
  0.325
  13/40
```

Bilangan Nyata

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

```
>longest 1/3
```

```
       0.3333333333333333
```

Representasi ganda internal membutuhkan 8 byte.

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

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

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

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

Latihan membuat contoh soal dan penyelesaiannya
longest 2/9

```
>longest 2/9
```

```
0.2222222222222222
```

Latihan membuat contoh soal dan penyelesaiannya
printdual (2/9)

```
>printdual (2/9)
```

```
1.11000111000111000111000111000111000111000111000111100*2^-3
```

Latihan membuat contoh soal dan penyelesaiannya
printhex (2/9)

```
>printhex (2/9)
```

```
3.8E38E38E38E38*16^-1
```

Latihan membuat contoh soal dan penyelesaiannya
longest pi/7

```
>longest pi/7
```

```
        0.4487989505128276
```

Latihan membuat contoh soal dan penyelesaiannya
printdual(pi/7)

```
>printdual(pi/7)
```

```
  1.1100101110010001111100111011101110111010000101000000*2^-2
```

Strings

Sebuah string di Euler didefinisikan dengan "...".

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

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

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

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

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

Fungsi cetak juga mengkonversi angka menjadi string. Ini bisa memakan waktu a jumlah digit dan jumlah tempat (0 untuk keluaran padat), dan optimal a satuan.

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

```
  Golden Ratio : 1.61803
```

Ada string khusus none yang tidak dicetak. Itu dikembalikan oleh beberapa fungsi, ketika hasilnya tidak menjadi masalah. (Ini dikembalikan secara otomatis, jika fungsi tidak memiliki pernyataan return.)

```
>none
```

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

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

```
  1234.5
```

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

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

```
  affe
  charlie
  bravo
```

Vektor string kosong dilambangkan dengan [tidak ada]. Vektor string dapat berupa digabungkan

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

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

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

String Unicode dapat digabungkan seperti string lainnya.

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

```
 = 45°
```

I

Di komentar, entitas yang sama seperti , dll. dapat digunakan. Ini mungkin merupakan alternatif cepat untuk Lateks. (Detail lebih lanjut 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 grafik keluarf().

```
>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 diwakili dengan 1=true atau 0=false di Euler.
% String dapat dibandingkan, seperti halnya angka.

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

```
  0
  1
```

”dan” adalah operator ”&&” dan ”atau” adalah operator ”||”, seperti dalam bahasa C. (Kata ”dan” dan ”atau” hanya dapat digunakan dalam kondisi ”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 nonzeros() untuk mengekstrak elemen tertentu dari a vektor. Dalam contoh ini, kita menggunakan kondisi isprime(n).

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

```
[2,  3,  5,  7,  9,  11,  13,  15,  17,  19,  21,  23,  25,  27,  29,
31,  33,  35,  37,  39,  41,  43,  45,  47,  49,  51,  53,  55,  57,
59,  61,  63,  65,  67,  69,  71,  73,  75,  77,  79,  81,  83,  85,
87,  89,  91,  93,  95,  97,  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]
```

Latihan membuat contoh soal dan penyelesaiannya
u" ="+60+u""

```
>u"&theta; ="+60+u"&deg;"
```

=60°

Latihan membuat contoh soal dan penyelesaiannya
E:=[12, 22, 45,69]; E[nonzeros(E<50)]

```
>E:=[12, 22, 45,69]; E[nonzeros(E<50)]
```

[12, 22, 45]

Latihan membuat contoh soal dan penyelesaiannya
(2:40)>20, nonzeros(%)

```
>(2:40)>20, nonzeros(%)
```

```
[0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
0,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
1,  1,  1]
[20,  21,  22,  23,  24,  25,  26,  27,  28,  29,  30,  31,  32,  33,
34,  35,  36,  37,  38,  39]
```

Latihan membuat contoh soal dan penyelesaiannya
M:=4|6:2:20

```
>M:=4|6:2:20
```

```
  [4,  6,  8,  10,  12,  14,  16,  18,  20]
```

Latihan membuat contoh soal dan penyelesaiannya
S:=8|1:4:25

```
>S:=8|1:4:25
```

```
  [8,  1,  5,  9,  13,  17,  21,  25]
```

Format Keluaran

Format keluaran default EMT mencetak 12 digit. Untuk memastikan bahwa kami melihat defaultnya, kami mengatur ulang formatnya.

```
>defformat; pi
```

```
  3.14159265359
```

Secara internal, EMT menggunakan standar IEEE untuk angka ganda dengan sekitar 16 angka desimal. Untuk melihat jumlah digit selengkapnya, gunakan perintah"format terpanjang", atau kita menggunakan operator "terpanjang" untuk menampilkan hasilnya
format terpanjang.

```
>longest pi
```

```
        3.141592653589793
```

Berikut adalah representasi heksadesimal internal dari bilangan ganda.

```
>printhex(pi)
```

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

Format keluaran 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 "format terpendek", "format pendek", "format panjang" berfungsi 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" juga mengatur format skalar.

```
>longestformat; pi
```

```
  3.141592653589793
```

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

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

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

Tetapi format output EMT dapat diatur dengan cara yang fleksibel.

```
>longestformat; pi,
```

```
  3.141592653589793
```

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

```
    3.14159
```

Standarnya adalah defformat().

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

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

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

```
        4.934802200544679
```

Ada juga operator singkat 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 secara tepat. Kesalahannya bertambah sedikit, seperti yang Anda lihat pada 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 menyadarinya. Untuk kenyamanan, keluaran 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
```

Latihan membuat contoh soal dan penyelesaiannya
defformat; 123.6789

```
>defformat; 123.6789
```

```
  123.6789
```

Latihan membuat contoh soal dan penyelesaiannya
fraction 1/2+1/3+1/4+1/5

```
>fraction 1/2+1/3+1/4+1/5
```

```
  77/60
```

Latihan membuat contoh soal dan penyelesaiannya
e:=exp(1); longest e^pi

```
>e:=exp(1); longest e^pi
```

```
        23.14069263277926
```

Latihan membuat contoh soal dan penyelesaiannya
format (12,5); 1/3, pi, sin(1)

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

```
      0.33333
      3.14159
      0.84147
```

Latihan membuat contoh soal dan penyelesaiannya
format (15,6); log (10)

```
>format (15,6); log (10)
```

```
        2.302585
```

Ekspresi

String atau nama dapat digunakan untuk menyimpan ekspresi matematika, yang dapat dievaluasi dengan EMT. Untuk ini, gunakan tanda kurung setelah ekspresi. Jika Anda ingin menggunakan string sebagai ekspresi, gunakan konvensi untuk menamainya "fx" atau "fxy" dll. Ekspresi lebih diutamakan daripada fungsi.

Variabel global dapat digunakan dalam evaluasi.

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

```
  Syntax error in expression, or unfinished expression!
  Error in:
  =r:=2; fx:="pi*r^2"; longest fx() ...
  ^
```

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)
```

```
       -0.919536
```

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

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

```
       36.000000
```

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

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

```
       45.000000
```

Sebagai referensi, kami mencatat 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)
```

```
       45.000000
```

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

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

```
       12.000000
```

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

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

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

Bentuk ekspresi khusus memungkinkan variabel apa pun sebagai parameter yang tidak disebutkan namanya untuk mengevaluasi ekspresi, bukan hanya "x", "y", dll. Untuk ini, mulailah ekspresi dengan "@(variabel) ...".

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

```
  @(a,b) a^2+b^2
       41.000000
```

Hal ini memungkinkan untuk memanipulasi ekspresi dalam variabel lain untuk fungsi EMT yang memerlukan ekspresi dalam "x".

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

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

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

```
       -0.475000
```

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

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

```
       -0.425000
```

```
>fx := (a*sin(x)^2)fx(0.5,a=4.5)
```

```
       -2.142030
```

Sebuah ekspresi tidak perlu bersifat simbolis. Hal 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 memperhatikan bahwa ada perbedaan sintaksis antara sintaksis asli Maxima dan sintaksis default ekspresi simbolik di EMT.

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

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

```
>$&44!
```

$$26582715747884487680436258110146158903196385280000000000$$

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

lateks: 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 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 yang disediakan oleh penulis program tersebut.

Anda akan mengetahui bahwa cara berikut juga bisa dilakukan.

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

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

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

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

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

120

Dengan begitu Anda bisa menggunakan solusi suatu persamaan di persamaan lain.

Ekspresi simbolik dicetak oleh Maxima dalam bentuk 2D. Alasannya adalah adanya tanda simbolis khusus pada string tersebut.

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

Untuk mencetak ekspresi simbolik dengan LaTeX, gunakan $ di depan & (atau Anda dapat menghilangkan &) sebelum perintah. Jangan jalankan perintah Maxima dengan $, jika Anda belum menginstal LaTeX.

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

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

Ekspresi simbolik diurai oleh Euler. Jika Anda memerlukan sintaksis kompleks dalam satu ekspresi, Anda dapat mengapit ekspresi tersebut di "...". Menggunakan lebih dari sekadar ekspresi sederhana bisa saja dilakukan, namun sangat tidak disarankan.

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


Untuk kelengkapannya, kami perhatikan bahwa ekspresi simbolik dapat digunakan program, tetapi harus diapit tanda petik. Apalagi jumlahnya jauh lebih banyak efektif untuk memanggil Maxima pada waktu kompilasi jika memungkinkan.

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

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

Sekali lagi, % mengacu pada hasil sebelumnya.

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

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

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

Ekspresi simbolik dapat digunakan dalam ekspresi simbolik lainnya.

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

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

Input langsung dari perintah Maxima juga tersedia. Mulai baris perintah dengan "::". Sintaks Maxima disesuaikan dengan sintaks EMT (disebut "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 sintaksis asli Maksimum. Anda dapat melakukan ini 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 perintah berikut sisi kanan &= dievaluasi sebelum ditugaskan ke Fx.

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

```
     5
125 E
```

$$125\, e^5$$

```
18551.64488782208
```

```
>fx(5)
```

```
31104000.000000 644204000.000000 23066015625.000000 195503918625.000000
```

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

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))
```

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

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

```
>tex(fx)
```

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

Ekspresi simbolik dapat dievaluasi seperti halnya ekspresi numerik.

```
>fx(0.5)
```

```
        0.433013        0.586302        0.838525        1.038328
```

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

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

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

Penugasannya juga bisa bersifat simbolis.

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

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

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

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

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

Bandingkan dengan perintah numerik "solve" di Euler, yang memerlukan nilai awal, dan opsional nilai target.

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

```
        1.561553
```

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

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

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

```
       -3.236068        1.236068
```

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

Untuk mendapatkan solusi simbolik tertentu, seseorang dapat menggunakan "dengan" 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 simbolis dapat memiliki bendera, yang menunjukkan perlakuan khusus di Maxima. Beberapa flag dapat digunakan sebagai perintah juga, yang lainnya tidak. Bendera 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(%)
```

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

Latihan membuat contoh soal dan penyelesaiannya
::: factor(30!)

```
>::: factor(30!)
```

```
                  26  14  7  4   2   2
2^26 3^14 5^7 7^4 11^2 13^2 17 19 23 29
```

Latihan membuat contoh soal dan penyelesaiannya
fx(9)

```
>fx(9)
```

```
  3761479876608.000000 880099259770368.000000 551620188439453120.000000 25844572032135069696.000000
```

Latihan membuat contoh soal dan penyelesaiannya
&(fx with x=25)-(fx with x=10), &float(%)

```
>&(fx with x=25)-(fx with x=10), &float(%)
```

$$15625\ E^{25} - 1000\ E^{10}$$

```
1.125076530120191e+15
```

Latihan membuat contoh soal dan penyelesaiannya
&factor(6!)

```
>&factor(6!)
```

720

Latihan membuat contoh soal dan penyelesaiannya
gx&= (x^2+3)/(x^3+2); $&fx

```
>gx&= (x^2+3)/(x^3+2); $&fx
```

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

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 simbolik. Fungsi satu baris numerik didefinisikan oleh ":=".

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

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

```
>f(2):
```

```
Error in return result.
Try "trace errors" to inspect local variables after errors.
f:
    useglobal; return x*sqrt(x^2+1):
Error in:
f(2): ...
    ^
```

Fungsi ini juga dapat digunakan untuk vektor, mengikuti bahasa matriks Euler, karena ekspresi yang digunakan dalam fungsi tersebut divektorkan.

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

```
Error in return result.
Try "trace errors" to inspect local variables after errors.
f:
    useglobal; return x*sqrt(x^2+1):
Error in:
f(0:0.1:1) ...
          ^
```

Fungsi dapat diplot. Daripada ekspresi, kita hanya perlu memberikan nama fungsinya.

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

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

```
Error in return result.
f:
    useglobal; return x*sqrt(x^2+1):
secant:
    y0=f$(x0,args())-y; y1=f$(x1,args())-y;
Try "trace errors" to inspect local variables after errors.
solve:
    if eps then return secant(f$,a,b,y;args(),eps=eps);
```

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

Anda masih dapat memanggil fungsi bawaan sebagai "_...", jika fungsi tersebut ada di inti Euler.

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

```
0.707107
```

Sebaiknya kita menghilangkan redefinisi dosa ini.

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

```
0.707107
```

Latihan membuat contoh soal dan penyelesaiannya
function f(x) := x*sqrt(x^2+1)

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

Latihan membuat contoh soal dan penyelesaiannya
f(4)

```
>f(4)
```

```
       16.492423
```

Latihan membuat contoh soal dan penyelesaiannya
g(0:0.2:2)

```
>g(0:0.2:2)
```

```
  Unexpected "(". Index () not allowed in strict mode!
  In Euler files, use relax to avoid this.
  Error in:
  g(0:0.2:2) ...
            ^
```

Latihan membuat contoh soal dan penyelesaiannya
solve(”f”,4,y=4)

```
>solve("f",4,y=4)
```

```
        1.879130
```

Latihan membuat contoh soal dan penyelesaiannya
function overwrite tan(x) := _tan(x°); tan(45)

```
>function overwrite tan(x) := _tan(x°); tan(45)
```

Parameter Bawaan

Fungsi numerik dapat memiliki parameter default.

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

Menghilangkan parameter ini akan menggunakan nilai default.

```
>f(4)
```

```
       16.000000
```

Menyetelnya akan menimpa nilai default.

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

```
       80.000000
```

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

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

```
       16.000000
```

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

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

```
       24.000000
```

Namun parameter yang ditetapkan mengesampingkan nilai global.

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

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

```
20.000000
```

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

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

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

Fungsi simbolik dapat digunakan dalam ekspresi simbolik.

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

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

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

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

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

```
      207.124419      211.128005      213.608878      214.438560
```

Mereka dapat digunakan untuk mendefinisikan fungsi atau ekspresi simbolik lainnya.

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

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

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

```
Cannot use vectors for conditions, use all(...)!
Maybe you need to vectorize the function with "map".
secant:
    if x2~=x1; break; endif;
Try "trace errors" to inspect local variables after errors.
solve:
    if eps then return secant(f$,a,b,y;args(),eps=eps);
```

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

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

```
Cannot use vectors for conditions, use all(...)!
Maybe you need to vectorize the function with "map".
secant:
    if x2~=x1; break; endif;
Try "trace errors" to inspect local variables after errors.
solve:
    if eps then return secant(f$,a,b,y;args(),eps=eps);
```

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

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

```
>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)
```

```
      625.000000
```

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

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

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

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

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

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

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

$$x^3-x$$

Dengan &= fungsinya bersifat 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 tertentu

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

yang tidak dapat dievaluasi secara simbolis.

Jika kita mendefinisikan ulang fungsi tersebut dengan kata kunci “peta” maka dapat digunakan untuk vektor x. Secara internal, fungsi ini 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)
```

```
  Real 1 x 5 matrix

       -0.783431       -0.410816        0.000000      ...
```

Fungsi dapat memiliki nilai default untuk parameter.

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

Sekarang fungsinya bisa dipanggil dengan atau tanpa parameter ”base”.

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

```
        2.000000
        6.700000
```

Selain itu, dimungkinkan untuk menggunakan parameter yang ditetapkan.

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

```
        2.000000        2.000000        2.000000        2.000000
```

Seringkali, kita ingin menggunakan fungsi untuk vektor di satu tempat, dan untuk elemen individual di tempat lain. Hal 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 ini dapat digunakan untuk variabel simbolik.
Namun fungsinya juga dapat digunakan untuk vektor numerik.

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

```
       17.000000
```

Ada juga fungsi yang murni simbolik, 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

Namun 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)
```

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

Latihan membuat contoh soal dan penyelesaiannya
f(4,a:=10)

```
>f(4,a:=10)
```

```
  Function f needs at least 2 arguments!
  Use: f (x, y)
  Error in:
  f(4,a:=10) ...
            ^
```

Latihan membuat contoh soal dan penyelesaiannya
g(6+g(2))

```
>g(6+g(2))
```

```
     2735.841433      2741.570965      2743.419300      2743.753000
```

Latihan membuat contoh soal dan penyelesaiannya
v=[6,8]; f(v)

```
>v=[6,8]; f(v)
```

```
  Function f needs at least 2 arguments!
  Use: f (x, y)
  Error in:
  v=[6,8]; f(v) ...
               ^
```

Latihan membuat contoh soal dan penyelesaiannya
mylog(100), mylog(4^3.7,2)

```
>mylog(100), mylog(4^3.7,2)
```

```
        2.000000
        7.400000
```

Latihan membuat contoh soal dan penyelesaiannya
f(5,20)

```
>f(5,20)
```

```
  2396134608750.000000
```

Untuk meringkas

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

Memecahkan Ekspresi

Ekspresi dapat diselesaikan secara numerik dan simbolis.

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

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

```
        1.414214
```

Ini juga berfungsi untuk ekspresi simbolik. Ambil fungsi berikut.

```
>$&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])
```

$$\left[\left[x=-\frac{c\,e}{b\,(d-5)-a\,e},y=\frac{c\,(d-5)}{b\,(d-5)-a\,e}\right]\right]$$

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

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

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

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

```
Function csolve not found.
Try list ... to find functions!
Error in:
csolve(px,1,y=2), px(%) ...
                ^
```

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

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

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

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

```
>longest sol()
```

```
      -0.6180339887498949       1.618033988749895
```

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

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

$$0$$

Penyelesaian 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])
```

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

The function f() can see global variables. But often we want to use local parameters.

$$a^x - x^a = 0.1$$

with 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 parameternya (cara lainnya adalah parameter titik koma).

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

```
        2.541163
```

Ini juga berfungsi dengan ekspresi. Namun 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)
```

```
        2.541163
```

Latihan membuat contoh soal dan penyelesaiannya
$&solve(x^3=3,x)

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

$$\left[x=\frac{3^{\frac{5}{6}}\,i-3^{\frac{1}{3}}}{2},x=\frac{-3^{\frac{5}{6}}\,i-3^{\frac{1}{3}}}{2},x=3^{\frac{1}{3}}\right]$$

Latihan membuat contoh soal dan penyelesaiannya
$&solve([x+y=5,xˆ3+3*y+x=10],[x,y])

```
>$&solve([x+y=5,xˆ3+3*y+x=10],[x,y])
```

$$[[x = 1.135939889088928\,i + 1.047275740771163, y = 3.952724259228837 - 1.135939889088928\,i], [x = 1.047275740771163$$

Latihan membuat contoh soal dan penyelesaiannya
$&xˆ2 with sol[1], $&expand(xˆ2-2*x with sol[2])

```
>$&xˆ2 with sol[1], $&expand(xˆ2-2*x with sol[2])
```

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

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

Latihan membuat contoh soal dan penyelesaiannya
solve({{”xˆa-aˆx”,a=4}},3,y=0.2)

```
>solve({{"xˆa-aˆx",a=4}},3,y=0.2)
```

```
2.020015
```

Latihan membuat contoh soal dan penyelesaiannya
longest sol(5)

```
>longest sol(5)
```

```
      -0.6180339887498949       1.618033988749895
```

Menyelesaikan Ketidaksamaan

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

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

```
        C:/Program Files/Euler x64/maxima/share/maxima/5.35.1/share/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]) // tidak memiliki penyelesaian
```

$$emptyset$$

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

$$universalset$$

```
>$&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]) // sistem pertidaksamaan
```

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

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

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

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

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

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

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

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

```
        [6 < x, x < 8, y < - 11] 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])
```

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

Latihan membuat contoh soal dan penyelesaiannya
$&fourier_elim([(x+5)/(x-8) <= 5],[x])

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

$$\left[x = \frac{45}{4}\right] \vee \left[\frac{45}{4} < x\right] \vee [x < 8]$$

Latihan membuat contoh soal dan penyelesaiannya
&fourier_elim([max(x,y) > 5, x 7, abs(y-2) > 10],[x,y])

```
>&fourier_elim([max(x,y) > 5, x # 7, abs(y-2) > 10],[x,y])
```

```
[5 < x, x < 7, y < - 8] or [7 < x, y < - 8] or [x < 7, 12 < y]
or [x = y, 12 < y] or [7 < x, x < y, 12 < y] or [y < x, 12 < y]
```

Latihan membuat contoh soal dan penyelesaiannya
$&fourier_elim(((x + y < 6) and x < 2) or (x - y >7),[x,y])

```
>$&fourier_elim(((x + y < 6) and x < 2) or  (x - y >7),[x,y])
```

$$[y + 7 < x] \vee [x < min\,(2, 6 - y)]$$

Latihan membuat contoh soal dan penyelesaiannya
$&fourier_elim((x + y < 6) and (x - y >7),[x,y])

```
>$&fourier_elim((x + y < 6) and (x - y >7),[x,y])
```

$$\left[y + 7 < x, x < 6 - y, y < -\frac{1}{2}\right]$$

Latihan membuat contoh soal dan penyelesaiannya
$&fourier_elim([xˆ2 - 4>0],[x]) // xˆ2-4 > 0

```
>$&fourier_elim([xˆ2 - 4>0],[x]) // xˆ2-4 > 0
```

$$[2 < x] \vee [x < -2]$$

Bahasa Matriks

Dokumentasi inti EMT berisi pembahasan rinci tentang bahasa matriks Euler.

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

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

```
       1.000000       2.000000
       3.000000       4.000000
```

Hasil kali matriks dilambangkan dengan titik.

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

```
       3.000000
       4.000000
```

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

```
       3.000000       4.000000
```

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

```
       -2.000000        1.000000
        1.500000       -0.500000
```

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

```
       11.000000
       25.000000
```

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

```
        1.000000        0.000000
        0.000000        1.000000
```

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

```
>A.A
```

```
        7.000000       10.000000
       15.000000       22.000000
```

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

```
        1.000000        4.000000
        9.000000       16.000000
```

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

```
       37.000000       54.000000
       81.000000      118.000000
```

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

```
       37.000000       54.000000
       81.000000      118.000000
```

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

```
        1.000000        1.000000
        1.000000        1.000000
```

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

```
         0.333333        0.666667
         0.750000        1.000000
```

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

```
        -2.000000
         2.500000
```

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

```
        -2.000000
         2.500000
```

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

```
         1.000000        0.000000
         0.000000        1.000000
```

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

```
        1.000000        0.000000
        0.000000        1.000000
```

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

```
        1.000000        4.000000
        9.000000       16.000000
```

Ini bukan hasil kali matriks, melainkan perkalian elemen demi elemen. Hal yang sama juga berlaku untuk vektor.

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

```
        9.000000
       16.000000
```

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

```
>2*A
```

```
       2.000000       4.000000
       6.000000       8.000000
```

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

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

```
       1.000000       4.000000
       3.000000       8.000000
```

Jika ini adalah vektor baris, maka diterapkan ke semua kolom A.

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

```
       2.000000       6.000000
       6.000000      12.000000
```

Kita dapat membayangkan perkalian ini seolah-olah vektor baris v telah diduplikasi untuk membentuk matriks yang berukuran sama dengan A.

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

```
       1.000000       2.000000
       1.000000       2.000000
```

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

```
       1.000000       4.000000
       3.000000       8.000000
```

Hal ini juga berlaku untuk dua vektor dimana yang satu adalah vektor baris dan yang lainnya adalah vektor kolom. Kita menghitung i*j untuk i,j dari 1 sampai 5. Caranya adalah dengan mengalikan 1:5 dengan transposenya. Bahasa matriks Euler secara otomatis menghasilkan tabel nilai.

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

```
  Real 5 x 5 matrix

       1.000000       2.000000       3.000000     ...
       2.000000       4.000000       6.000000     ...
       3.000000       6.000000       9.000000     ...
       4.000000       8.000000      12.000000     ...
       5.000000      10.000000      15.000000     ...
```

Sekali lagi, ingatlah bahwa ini bukan produk matriks!

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

```
       55.000000
```

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

```
       55.000000
```

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

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

```
  Real 1 x 10 matrix
       1.000000       1.000000       1.000000     ...
```

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

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

```
        5.000000
```

Euler memiliki operator perbandingan, seperti "==", yang memeriksa kesetaraan.
Kita mendapatkan vektor 0 dan 1, dimana 1 berarti benar.

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

```
  Real 1 x 10 matrix
        0.000000       0.000000       0.000000      ...
```

Dari vektor tersebut, "bukan nol" memilih elemen bukan nol.
Dalam hal ini, kita mendapatkan indeks semua elemen lebih besar dari 50.

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

```
        8.000000       9.000000      10.000000
```

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
```

```
       64.000000       81.000000      100.000000
```

Sebagai contoh, mari kita cari semua kuadrat bilangan 1 sampai 1000, yaitu 5 modulo 11 dan 3 modulo 13.

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

```
  Real 1 x 28 matrix

       4.000000       48.000000       95.000000      ...
```

EMT tidak sepenuhnya efektif untuk perhitungan bilangan bulat. Ia menggunakan floating point presisi ganda secara internal. Namun, seringkali hal ini sangat berguna.

Kita dapat memeriksa primalitasnya. Mari kita cari tahu, berapa banyak persegi ditambah 1 yang merupakan bilangan prima.

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

```
      112.000000
```

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

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

```
0.765761    0.401188    0.406347    0.267829
0.136730    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
```

```
1.000000    4.000000
2.000000    1.000000
2.000000    2.000000
3.000000    2.000000
```

Indeks ini dapat digunakan untuk mengatur elemen ke nilai tertentu.

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

```
0.765761    0.401188    0.406347    0.000000
0.000000    0.000000    0.495975    0.952814
0.548138    0.000000    0.444255    0.539246
```

Fungsi mset() juga dapat mengatur elemen pada indeks ke entri 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
```

And it is possible to get the elements in a vector.

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

```
        0.267829       0.136730       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       4.000000       0.765761       1.000000
        0.136730       1.000000       0.952814       4.000000
        0.006085       2.000000       0.548138       1.000000
```

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
```

Namun 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)
```

```
         1.000000        1.000000
         2.000000        4.000000
         3.000000        1.000000
        -0.765761       -0.952814       -0.548138
```

Latihan membuat contoh soal dan penyelesaiannya
A=[4,3;2,1]

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

```
        4.000000        3.000000
        2.000000        1.000000
```

Latihan membuat contoh soal dan penyelesaiannya
b=[2;1]

```
>b=[2;1]
```

```
        2.000000
        1.000000
```

Latihan membuat contoh soal dan penyelesaiannya
inv(A) //inverse A

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

```
       -0.500000        1.500000
        1.000000       -2.000000
```

Latihan membuat contoh soal dan penyelesaiannya
A.b //perkalian matriks

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

```
       11.000000
        5.000000
```

Latihan membuat contoh soal dan penyelesaiannya
A.A

```
>A.A
```

```
       22.000000      15.000000
       10.000000       7.000000
```

Fungsi Matriks Lainnya (Matriks Bangunan)

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

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

```
       1.000000       2.000000       3.000000
       1.000000       2.000000       3.000000
```

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

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

```
  Real 3 x 5 matrix

       0.032444       0.053417       0.595713     ...
       0.839160       0.175552       0.396988     ...
       0.025757       0.658585       0.629832     ...
```

Jika jumlah barisnya tidak sama, matriks yang lebih pendek diisi dengan 0.

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

```
>A|1
```

```
  Real 3 x 5 matrix

        0.032444        0.053417        0.595713      ...
        0.839160        0.175552        0.396988      ...
        0.025757        0.658585        0.629832      ...
```

Dimungkinkan untuk membuat matriks vektor baris dan kolom.

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

```
        1.000000        2.000000        3.000000
        1.000000        2.000000        3.000000
```

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

```
        1.000000        1.000000
        2.000000        2.000000
        3.000000        3.000000
```

Tujuan utamanya adalah untuk menafsirkan ekspresi vektor untuk vektor kolom.

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

```
        1.000000        1.000000
        2.000000        4.000000
        3.000000        9.000000
```

Untuk mendapatkan ukuran A, kita bisa menggunakan fungsi berikut.

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

```
        2.000000
        4.000000
        2.000000        4.000000
        4.000000
```

Untuk vektor, ada panjang().

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

```
        9.000000
```

Masih banyak fungsi lain yang menghasilkan matriks.

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

```
        1.000000        1.000000
        1.000000        1.000000
```

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

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

```
  Real 1 x 5 matrix

        6.000000        6.000000        6.000000      ...
```

Matriks bilangan acak juga dapat dihasilkan dengan acak (distribusi seragam) atau normal (distribusi Gauß).

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

```
        0.665660        0.831835
        0.977000        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
```

```
       1.000000       2.000000       3.000000
       4.000000       5.000000       6.000000
       7.000000       8.000000       9.000000
```

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

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

Let us test.

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

```
  Real 1 x 15 matrix

       1.000000       2.000000       3.000000     ...
```

The function multdup() duplicates elements of a vector.

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

```
  Real 1 x 15 matrix

        1.000000       1.000000       1.000000     ...
  Real 1 x 7 matrix

        1.000000       1.000000       2.000000     ...
```

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

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

```
  Real 1 x 5 matrix

        5.000000       4.000000       3.000000     ...
```

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

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

```
  Real 1 x 5 matrix
        2.000000        3.000000        4.000000      ...
```

A special function is drop(v,i), which removes the elements with the indices in i from the vector v.

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

```
  Real 1 x 10 matrix
       10.000000       11.000000       13.000000      ...
```

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

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

```
Real 1 x 15 matrix

      2.000000       3.000000       5.000000     ...
Real 1 x 11 matrix

      0.000000       5.000000       0.000000     ...
Real 1 x 11 matrix

      2.000000       3.000000       5.000000     ...
```

Seperti yang Anda lihat, tidak ada salahnya 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]))
```

```
Real 1 x 8 matrix

      1.000000       2.000000       3.000000     ...
```

Ada beberapa fungsi khusus untuk mengatur diagonal atau menghasilkan matriks diagonal.

Kita mulai dengan matriks identitas.

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

```
Real 5 x 5 matrix

      1.000000       0.000000       0.000000     ...
      0.000000       1.000000       0.000000     ...
      0.000000       0.000000       1.000000     ...
      0.000000       0.000000       0.000000     ...
      0.000000       0.000000       0.000000     ...
```

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

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

```
Real 5 x 5 matrix

      1.000000       0.000000       0.000000     ...
      1.000000       1.000000       0.000000     ...
      0.000000       2.000000       1.000000     ...
      0.000000       0.000000       3.000000     ...
      0.000000       0.000000       0.000000     ...
```

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

Berikut adalah fungsi yang mengembalikan matriks tri-diagonal.

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

```
Real 5 x 5 matrix

      2.000000      3.000000      0.000000     ...
      1.000000      2.000000      3.000000     ...
      0.000000      1.000000      2.000000     ...
      0.000000      0.000000      1.000000     ...
      0.000000      0.000000      0.000000     ...
```

Diagonal suatu matriks juga dapat diekstraksi dari matriks tersebut. Untuk mendemonstrasikannya, kami menyusun ulang vektor 1:9 menjadi matriks 3x3.

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

```
      1.000000      2.000000      3.000000
      4.000000      5.000000      6.000000
      7.000000      8.000000      9.000000
```

Sekarang kita dapat mengekstrak diagonalnya.

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

```
      1.000000      5.000000      9.000000
```

Misalnya. Kita dapat membagi matriks dengan diagonalnya. Bahasa matriks menjaga agar vektor kolom d diterapkan pada matriks baris demi baris.

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

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

Latihan membuat contoh soal dan penyelesaiannya
v=2:3; v_v

```
>v=2:3; v_v
```

```
2.000000  3.000000
2.000000  3.000000
```

Latihan membuat contoh soal dan penyelesaiannya
A=random(2,4); A|v

```
>A=random(2,4); A|v
```

```
  Real 2 x 6 matrix

        0.208566        0.220144        0.855399      ...
        0.259286        0.181379        0.293642      ...
```

Latihan membuat contoh soal dan penyelesaiannya
A|1

```
>A|1
```

```
  Real 2 x 5 matrix

        0.208566        0.220144        0.855399      ...
        0.259286        0.181379        0.293642      ...
```

Latihan membuat contoh soal dan penyelesaiannya
v;v

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

```
        2.000000        3.000000
        2.000000        3.000000
```

Latihan membuat contoh soal dan penyelesaiannya
v',v'

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

```
2.000000        2.000000
3.000000        3.000000
```

Vektorisasi

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

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

```
        1.000000        1.414214        1.732051
```

So you can easily create a table of values. This is one way to plot a function (the alternative uses an expression).

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

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

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

lateks: s = t^3-t

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

```
  Real 1 x 21 matrix

        0.000000        0.171000        0.288000      ...
```

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

Misalnya, vektor kolom dikali vektor baris diperluas ke matriks, jika operator diterapkan. Berikut ini, v' adalah vektor yang dialihkan (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, ini sangat berbeda dengan perkalian matriks. Hasil kali matriks dilambangkan dengan titik "." di EMT.

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

```
       55.000000
```

Secara default, vektor baris dicetak dalam format ringkas.

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

```
        1.000000        2.000000        3.000000        4.000000
```

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

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

```
        5.000000
       25.000000
```

Untuk mengubah urutan matriks kita menggunakan apostrof.

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

```
1.000000
2.000000
3.000000
4.000000
```

Jadi kita dapat menghitung matriks A dikalikan vektor b.

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

```
30.000000
70.000000
```

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

```
>v'.v
```

```
1.000000    2.000000    3.000000    4.000000
2.000000    4.000000    6.000000    8.000000
3.000000    6.000000    9.000000   12.000000
4.000000    8.000000   12.000000   16.000000
```

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

```
>v.v'
```

```
       30.000000
```

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

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

```
       30.000000
```

Operator dan fungsi mematuhi bahasa matriks Euler.

Berikut ringkasan peraturannya.

- Suatu fungsi yang diterapkan pada vektor atau matriks diterapkan pada setiap elemen.

- Operator yang mengoperasikan dua matriks dengan ukuran yang sama diterapkan secara berpasangan pada elemen-elemen matriks.

- Jika kedua matriks mempunyai dimensi yang berbeda, keduanya diekspansi secara wajar sehingga mempunyai ukuran yang sama.

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

Berikut ini adalah kasus sederhana dengan operator ^.

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

```
       1.000000       4.000000       9.000000
```

Ini kasus yang lebih rumit. Vektor baris dikalikan vektor kolom memperluas keduanya dengan cara menduplikasi.

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

```
       1.000000       2.000000       3.000000
       2.000000       4.000000       6.000000
       3.000000       6.000000       9.000000
```

Perhatikan bahwa perkalian skalar menggunakan perkalian matriks, bukan *!

```
>v.v'
```

```
       14.000000
```

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

```
sum,prod menghitung jumlah dan hasil kali baris
cumsum,cumprod melakukan hal yang sama secara kumulatif
menghitung nilai ekstrem setiap baris
extreme mengembalikan vektor dengan informasi ekstrem
diag(A,i) mengembalikan diagonal ke-i
setdiag(A,i,v) menyetel diagonal ke-i
id(n) matriks identitas
det(A) determinannya
charpoly(A) polinomial karakteristik
nilai eigen(A) nilai eigen
```

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

```
       1.000000       4.000000       9.000000
      14.000000
       1.000000       5.000000      14.000000
```

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

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

```
       1.000000       2.000000       3.000000       4.000000
  Real 1 x 5 matrix

       1.000000       3.000000       5.000000     ...
```

Untuk menggabungkan matriks dan vektor terdapat operator "|" Dan "_".

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

```
  Real 1 x 5 matrix

        1.000000       2.000000       3.000000     ...
        1.000000       2.000000       3.000000
        1.000000       1.000000       1.000000
```

Elemen-elemen matriks disebut dengan "A[i,j]".

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

```
        6.000000
```

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

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

```
        6.000000
        7.000000       8.000000       9.000000
```

Indeks juga dapat berupa vektor baris dari indeks. : menunjukkan semua indeks.

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

```
        2.000000        4.000000
        2.000000
        5.000000
        8.000000
```

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

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

```
        2.000000        3.000000
        5.000000        6.000000
        8.000000        9.000000
```

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

```
>A{4}
```

```
        4.000000
```

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

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

```
  Real 1 x 9 matrix

       1.000000       2.000000       3.000000     ...
  Real 1 x 9 matrix

       1.000000       2.000000       3.000000     ...
```

Untuk menggunakan matriks pada tabel, mari kita atur ulang ke format default, dan hitung tabel nilai sinus dan kosinus. Perhatikan bahwa sudut dinyatakan 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 sebuah matriks, yang setiap barisnya merupakan tabel t^i untuk satu i. Artinya, matriks tersebut memiliki elemen lateks: a_{i,j} = t_j^i, \quad 1 \le j \le 101, \quad 1 \le i \le n

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

Integrasi numerik integral() hanya berfungsi untuk batas interval skalar. Jadi kita perlu memvektorkannya.

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

Kata kunci ”peta” membuat vektorisasi fungsi tersebut. Fungsinya sekarang akan berfungsi untuk vektor bilangan.

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

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

Latihan membuat contoh soal dan penyelesaiannya
sqrt(2:5)

```
>sqrt(2:5)
```

```
  [1.41421,  1.73205,  2,  2.23607]
```

Latihan membuat contoh soal dan penyelesaiannya
t=-1:1.2:2; s=tˆ2-t

```
>t=-1:1.2:2; s=t^2-t
```

```
  [2,  -0.16,  0.56]
```

Latihan membuat contoh soal dan penyelesaiannya
shortest (2:5)*(2:5)'

```
>shortest (2:5)*(2:5)'
```

```
 4  6  8 10
 6  9 12 15
 8 12 16 20
10 15 20 25
```

Latihan membuat contoh soal dan penyelesaiannya
(2:5).(2:5)'

```
>(2:5).(2:5)'
```

```
  54
```

Latihan membuat contoh soal dan penyelesaiannya
v:=[2,2]; v.v', %^4

```
>v:=[2,2]; v.v', %^4
```

```
  8
4096
```

Sub-Matriks dan Elemen Matriks

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
```

We can access a complete line of a matrix.

```
>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 1xn dan mxn, tentukan semua kolom menggunakan indeks kedua yang kosong.

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

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

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

Di sini kita menginginkan baris pertama dan kedua A.

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

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

Kita bahkan dapat menyusun ulang A menggunakan vektor indeks. 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 juga berfungsi dengan kolom.

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
```

Alternatifnya, biarkan indeks pertama kosong.

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

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

Kita juga bisa mendapatkan baris terakhir A.

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

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

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

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

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

Kita juga dapat memberikan nilai pada baris A.

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

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

We can even assign to a sub-matrix if it has the proper size.

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

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

Kita bahkan dapat mengatur sub-matriks jika ukurannya sesuai.

```
>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, bergantung pada pengaturan sistem. Standarnya adalah pesan kesalahan. Namun perlu diingat 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] ...
    ^
```

Latihan membuat contoh soal dan penyelesaiannya
A=[10,20,30;40,50,60;70,80,90]

```
>A=[10,20,30;40,50,60;70,80,90]
```

```
            10            20            30
            40            50            60
            70            80            90
```

Latihan membuat contoh soal dan penyelesaiannya
A[3]

```
>A[3]
```

```
  [70,  80,  90]
```

Latihan membuat contoh soal dan penyelesaiannya
v=5:7; v[3]

```
>v=5:7; v[3]
```

```
  7
```

Latihan membuat contoh soal dan penyelesaiannya
A[2,]

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

```
  [40,  50,  60]
```

Latihan membuat contoh soal dan penyelesaiannya
A[[1,2]]

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

```
             10              20              30
             40              50              60
```

Menyortir dan Mengacak

Fungsi sort() mengurutkan vektor baris.

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

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

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

Mari kita mengacak sebuah vektor.

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

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

Indeks berisi urutan 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
```

Seperti yang Anda lihat, posisi entri ganda agak acak.

```
>ind
```

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

Fungsi unik mengembalikan daftar elemen unik vektor yang diurutkan.

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

```
  [5,  1,  4,  2,  5,  8,  5,  2,  7,  10]
  [1,  2,  4,  5,  7,  8,  10]
```

Ini juga berfungsi untuk vektor string.

```
>unique(s)
```

```
  a
  aa
  d
  e
```

Latihan membuat contoh soal dan penyelesaiannya
sort([10,40,30,50,20])

```
>sort([10,40,30,50,20])
```

```
  [10,  20,  30,  40,  50]
```

Latihan membuat contoh soal dan penyelesaiannya
v=shuffle(11:20)

```
>v=shuffle(11:20)
```

```
  [13,  15,  17,  16,  19,  12,  18,  11,  20,  14]
```

Latihan membuat contoh soal dan penyelesaiannya
{vs,ind}=sort(v); v[ind]

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

```
  [11,  12,  13,  14,  15,  16,  17,  18,  19,  20]
```

Latihan membuat contoh soal dan penyelesaiannya
S=[15, 25,35]; unique(S)

```
>S=[15, 25,35]; unique(S)
```

```
  [15,  25,  35]
```

Latihan membuat contoh soal dan penyelesaiannya
sort([5,4,3,2,1])

```
>sort([5,4,3,2,1])
```

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

Aljabar Linier

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

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

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

```
      -4
     4.5
```

Contoh lain, kita membuat matriks berukuran 200x200 dan jumlah baris-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 tepat.

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

```
4.627409566637652e-13
```

Jika sistem tidak mempunyai 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
```

Penentu matriks ini adalah 0.

```
>det(A)
```

```
0
```

Latihan membuat contoh soal dan penyelesaiannya
A=[10,20;30,40]; b=[5;6]; A\b

```
>A=[10,20;30,40]; b=[5;6]; A\b
```

```
-0.4
0.45
```

Latihan membuat contoh soal dan penyelesaiannya
A=normal(100,100); b=sum(A); longest totalmax(abs(inv(A).b-2))

```
>A=normal(100,100); b=sum(A); longest totalmax(abs(inv(A).b-2))
```

```
1.000000000000067
```

Latihan membuat contoh soal dan penyelesaiannya
A=[10,20,30;40,50,60;70,80,90]

```
>A=[10,20,30;40,50,60;70,80,90]
```

```
10  20  30
40  50  60
70  80  90
```

Latihan membuat contoh soal dan penyelesaiannya
det(A)

```
>det(A)
```

```
0
```

Latihan membuat contoh soal dan penyelesaiannya
A=[-10,-20;-30,-40]; b=[5;6]; A\b

```
>A=[-10,-20;-30,-40]; b=[5;6]; A\b
```

```
            0.4
          -0.45
```

Latihan membuat contoh soal dan penyelesaiannya
A=[10,20;30,40]; b=[1;2]; A\b

```
>A=[10,20;30,40]; b=[1;2]; A\b
```

```
              0
           0.05
```

Matriks Simbolik

Maxima memiliki matriks simbolik. Tentu saja Maxima dapat digunakan untuk permasalahan aljabar linier sederhana seperti itu. Kita dapat mendefinisikan matriks untuk Euler dan Maxima dengan &:=, lalu menggunakannya dalam ekspresi simbolik. Bentuk [...] yang biasa untuk mendefinisikan matriks dapat digunakan di Euler untuk mendefinisikan matriks simbolik.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Untuk mengekstrak vektor eigen tertentu memerlukan 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 sama 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 deretan matriks simbolik berfungsi sama seperti matriks numerik.

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

$$[1, a]$$

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

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

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

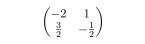
Ada fungsi simbolik di Maxima untuk membuat vektor dan matriks. Untuk ini, lihat dokumentasi Maxima atau tutorial tentang Maxima di EMT.

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

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

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

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



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

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

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

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

Perhatikan, bahwa dengan &:= matriks B telah didefinisikan sebagai simbolik dalam ekspresi simbolik dan 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]
```

Or symbolically. See the tutorial about Maxima for details on this.

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

$$\left[\left[\frac{15-3\sqrt{33}}{2},\frac{3\sqrt{33}+15}{2},0\right],[1,1,1]\right]$$

```
>$&A=[1,2;1,2];eigenvalues(A)
```

```
Space between commands expected!
Found: eigenvalues(A) (character 101)
You can disable this in the Options menu.
Error in:
 $&A=[1,2;1,2];eigenvalues(A) ...
              ^
```

Nilai Numerik dalam Ekspresi simbolik

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

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

```
                1        3.14159
                4              5
```

Atau secara simbolis. Lihat tutorial tentang Maxima untuk detailnya.

```
>$&A
```

$$\begin{pmatrix} 1 & \frac{1146408}{364913} \\ 4 & 5 \end{pmatrix}$$

Untuk menghindari hal ini, ada fungsi "mxmset(variabel)".

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

$$\begin{pmatrix} 1 & 3.141592653589793 \\ 4 & 5 \end{pmatrix}$$

Maxima can also compute with floating point numbers, and even with big floating numbers with 32 digits. The evaluation is much slower, however.

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

1.414213562373095

The precision of the big floating point numbers can be changed.

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

```
        3.14159265358979323846264338327950288419716939937510582097494\
 459230781640628620899862803482534211706 8b0
```

A numerical variable can be used in any symbolic expressions using "@var".
Note that this is only necessary, if the variable has been defined with ":=" or "=" as a numerical variable.

```
>B:=[1,pi;3,4]; $&det(@B)
```

$$-5.424777960769379$$

Latihan membuat contoh soal dan penyelesaiannya
A &:= [2,5;4,5]

```
>A &:= [2,5;4,5]
```

$$\begin{matrix} 2 & 5 \\ 4 & 5 \end{matrix}$$

Latihan membuat contoh soal dan penyelesaiannya
$&A

```
>$&A
```

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

Latihan membuat contoh soal dan penyelesaiannya
mxmset(A); $&A

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

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

Latihan membuat contoh soal dan penyelesaiannya
$&bfloat(sqrt(2)), $&float(sqrt(2))

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

$$1.414213562373095$$

Latihan membuat contoh soal dan penyelesaiannya
&fpprec:=100; &bfloat(pi)

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

```
          3.14159265358979323846264338327950288419716939937510582097494\
 459230781640628620899862803482534211706 8b0
```

Demo - Suku Bunga

Di bawah ini, kami menggunakan Euler Math Toolbox (EMT) untuk menghitung 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 sebesar 5.000 (katakanlah dalam dolar).

```
>K=5000
```

```
  5000
```

Sekarang kami mengasumsikan tingkat bunga 3% per tahun. Mari kita tambahkan satu tarif sederhana dan hitung hasilnya.

```
>K*1.03
```

```
  5150
```

Euler juga akan memahami sintaks berikut.

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

```
  5150
```

Tetapi lebih mudah menggunakan faktor tersebut

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

```
  1.03
  5150
```

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

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

```
  6719.58189672
```

Untuk keperluan kita, kita dapat mengatur formatnya menjadi 2 digit setelah titik desimal.

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

```
      6719.58
```

Mari kita cetak yang dibulatkan menjadi 2 digit dalam satu 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 ke 9? Untuk ini, bahasa matriks Euler sangat membantu. Anda tidak perlu menulis satu perulangan, tetapi cukup masuk

```
>K*q^(0:10)
```

```
  Real 1 x 11 matrix

      5000.00      5150.00      5304.50      5463.64      ...
```

Bagaimana keajaiban ini terjadi? 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 di Euler dapat diterapkan pada vektor elemen demi 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 nilainya.

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

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

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

Mari kita bandingkan kedua hasil tersebut, 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 mengulanginya selama bertahun-tahun. Euler memberikan banyak solusi untuk ini.

Cara termudah adalah fungsi iterate, 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 vektor tertentu, 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. Ingatlah 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
```

Latihan membuat contoh soal dan penyelesaiannya
VKr = [10, 20, 30, 40]; VK = [50, 60, 70, 80]; VKr[-1], VK[-1]

```
>VKr = [10, 20, 30, 40]; VK = [50, 60, 70, 80]; VKr[-1], VK[-1]
```

```
        40.00
        80.00
```

Latihan membuat contoh soal dan penyelesaiannya
VKr = [15, 25, 35, 45]; VKr[2], VKr[1:3]

```
>VKr = [15, 25, 35, 45]; VKr[2], VKr[1:3]
```

```
        25.00
        15.00       25.00       35.00
```

Latihan membuat contoh soal dan penyelesaiannya
VKr = [12, 22, 32, 42]; VKr'

```
>VKr = [12, 22, 32, 42]; VKr'
```

```
   12.00
   22.00
   32.00
   42.00
```

Latihan membuat contoh soal dan penyelesaiannya
K= 300

```
>K= 300
```

```
  300.00
```

Latihan membuat contoh soal dan penyelesaiannya
K*1.5

```
>K*1.5
```

```
  450.00
```

Perbedaannya sangat kecil.

Memecahkan Persamaan

Sekarang kita mengambil fungsi yang lebih maju, yang menambahkan tingkat uang tertentu setiap tahunnya.

```
>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 uangnya berkurang. Jelasnya, jika kita hanya mendapat bunga sebesar 150 pada tahun pertama, namun menghapus 200, kita kehilangan uang setiap tahunnya.

Bagaimana kita dapat menentukan berapa tahun uang tersebut akan bertahan? Kita harus menulis satu lingkaran untuk ini. Cara termudah adalah dengan melakukan 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
```

Alasannya adalah bukan nol (VKR<0) mengembalikan vektor indeks i, dengan VKR[i]<0, dan min menghitung indeks minimal.

Karena vektor selalu dimulai dengan indeks 1, maka 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 mengetahui nilainya dalah 0 setelah 50 tahun. Berapa tingkat bunganya?

Ini adalah pertanyaan yang hanya bisa dijawab secara numerik. Di bawah ini, kami akan melakukannya mendapatkan rumus yang diperlukan. Maka Anda akan melihat bahwa tidak ada yang mudah rumus untuk tingkat bunga. Namun untuk saat ini, kami menargetkan solusi numerik.

Langkah pertama adalah mendefinisikan fungsi yang melakukan iterasi sebanyak n kali. Kami tambahkan 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

lateks: x_{n+1} = x_n \cdot \kiri(1+ \frac{P}{100}\kanan) + R

Namun 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.

Apalagi kami hanya tertarik pada nilai terakhir. Jadi kita ambil indeks [-1].

Mari kita coba tes.

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

```
       -19.83
```

Now we can solve our problem.

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

```
         3.15
```

The solve routine solves expression=0 for the variable x. The answer is 3.15% per year. We take the start value of 3% for the algorithm. The solve() function always needs a start value.

We can use the same function to solve the following question: How much can we remove per year so that the seed capital is exhausted after 20 years assuming an interest rate of 3% per year.

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

```
      -336.08
```

```
>solve("f(500,x,3,2)",-100)
```

```
      -261.31
```

```
>R=300; iterate("onepay",4000,20)
```

```
  Real 1 x 21 matrix
       4000.00      4420.00      4852.60      5298.18      ...
```

Perhatikan bahwa Anda tidak dapat menyelesaikan jumlah tahun, karena fungsi kami mengasumsikan n sebagai nilai bilangan bulat.

Solusi Simbolis Masalah Suku Bunga

Kita dapat menggunakan bagian simbolis dari Euler untuk mempelajari masalahnya. Pertama kita mendefinisikan fungsi onepay() kita secara simbolis.

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

$$R + q\,K$$

Sekarang kita dapat mengulanginya.

```
>$&op(op(op(op(K)))), $&expand(%)
```

$$q^3\,R + q^2\,R + q\,R + R + q^4\,K$$

Kami melihat sebuah pola. Setelah n periode yang kita miliki
lateks: K_n = q^n K + R (1+q+\ldots+q^{n-1}) = q^n K + \frac{q^n-1}{q-1} R
Rumusnya adalah rumus 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 tanda "simpsum" untuk menguranginya menjadi hasil bagi. Mari kita membuat fungsi untuk ini.

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

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

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

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

```
-19.82504734650985
-19.82504734652684
```

Sekarang kita dapat menggunakannya untuk menanyakan waktu n. Kapan modal kita habis? Perkiraan awal kami adalah 30 tahun.

```
>solve("fs(5000,-330,3,x)",30)
```

```
20.51
```

Jawaban ini mengatakan akan menjadi negatif setelah 21 tahun.

Kita juga dapat menggunakan sisi simbolis Euler untuk menghitung rumus 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). Rumusnya jelas

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

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

Biasanya rumus ini diberikan dalam bentuk

lateks: i = \frac{P}{100}

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

$$\frac{((i+1)^n-1)\ R}{i}+(i+1)^n\ K=Kn$$

Kita dapat menyelesaikan nilai R secara simbolis.

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

$$\left[R = \frac{i\,Kn - i\,(i+1)^n\,K}{(i+1)^n - 1}\right]$$

Seperti yang Anda lihat dari rumusnya, fungsi ini mengembalikan kesalahan floating point untuk i=0. Euler tetap merencanakannya.

Tentu saja, kami memiliki batasan berikut.

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

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

Yang jelas, tanpa bunga kita harus membayar kembali 10 bunga 500.

Persamaan ini juga dapat diselesaikan untuk n. Kelihatannya lebih bagus, jika kita menerapkannya penyederhanaan untuk itu.

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

$$R + q\,K$$