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^(-4)*y^5*-7*x^2*y^(-9)
```

$$-\frac{42}{x^2\,y^4}$$

Menjabarkan:

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

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

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

Baris Perintah

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

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

```
>r:=3; h:=4; pi*r^2*h/3
     37.6991118431
```

Perintah harus dipisahkan dengan spasi. Baris perintah berikut mencetak dua hasilnya.

```
>pi*2*r*h, %+2*pi*r*h // Ingat tanda % menyatakan hasil perhitungan
terakhir sebelumnya
     75.3982236862
     150.796447372
```

Baris perintah dieksekusi sesuai urutan pengguna menekan tombol enter. Jadi Anda akan mendapatkan nilai baru setiap kali Anda mengeksekusi baris kedua.

```
>x := 3;
>x := cos(x) // nilai cosinus (x dalam radian)
     -0.9899924966
>x := cos(x)
     0.548696133603
```

Jika dua baris dihubungkan dengan "..." kedua baris akan selalu dieksekusi secara bersamaan.

```
>x := 1.7; ...
 x := (x+2/x)/2, x := (x+2/x)/2, x := (x+2/x)/2,
     1.43823529412
     1.41441417058
     1.4142135766
```

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

Untuk melipat semua baris yang terdiri dari beberapa baris, tekan Ctrl+L. Kemudian baris-baris berikutnya hanya akan terlihat, jika salah satunya menjadi fokus. Untuk melipat satu baris yang terdiri dari beberapa baris, mulailah baris pertama dengan "%+ ".

```
>%+ x=4+5; ...
  // This line will not be visible once the cursor is off the line
```

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

```
     81
```

Euler mendukung perulangan dalam baris perintah, asalkan dapat dimasukkan ke dalam satu baris atau beberapa baris. Dalam program, pembatasan ini tentu saja tidak berlaku. Untuk informasi lebih lanjut, lihat pengantar berikut.

```
>x=1; for i=1 to 5; x := (x+2/x)/2, end; // menghitung akar 2
     1.5
     1.41666666667
     1.41421568627
     1.41421356237
     1.41421356237
```

Tidak apa-apa menggunakan beberapa 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 kondisional juga berfungsi.

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

```
     Thought so!
```

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

Saat Anda menggerakkan kursor di sepanjang baris, pasangan tanda kurung buka dan tutup akan disorot. Perhatikan juga baris status. Setelah tanda kurung buka fungsi sqrt(), baris status akan menampilkan teks bantuan untuk fungsi tersebut. Jalankan perintah dengan tombol return.

```
>sqrt(sin(10°)/cos(20°))
```

```
     0.429875017772
```

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

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

```
>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 tetikus atau gunakan shift bersamaan dengan tombol kursor apa pun. Selain itu, Anda dapat menyalin tanda kurung yang disorot.

Contoh Soal Baris Perintah

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

```
>a:= 2; (3*a^2)*(-7*a^4)
```

```
     -1344
```

Sintaksis Dasar

Euler mengetahui fungsi matematika yang umum. Seperti yang telah Anda lihat di atas, fungsi trigonometri bekerja dalam radian atau derajat. Untuk mengonversi ke derajat,

tambahkan simbol derajat (dengan tombol F7) ke nilai, atau gunakan fungsi rad(x). Fungsi akar kuadrat disebut sqrt di Euler. Tentu saja, x^(1/2) juga memungkinkan.

Untuk mengatur variabel, gunakan "=" atau ":=". Demi kejelasan, pengantar ini menggunakan bentuk yang terakhir. Spasi tidak menjadi masalah. Namun, spasi di antara perintah diharapkan.

Beberapa perintah dalam satu baris dipisahkan dengan "," atau ";". Titik koma menghilangkan keluaran perintah. Di akhir baris perintah, "," diasumsikan, jika ";" tidak ada.

```
>g:=9.81; t:=2.5; 1/2*g*t^2
      30.65625
```

EMT menggunakan sintaks pemrograman untuk ekspresi. Untuk memasukkan

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

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

```
>E^2*(1/(3+4*log(0.6))+1/7)
      8.77908249441
```

Untuk menghitung ekspresi rumit seperti

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

Anda perlu memasukkannya dalam bentuk baris.

```
>((1/7 + 1/8 + 2) / (1/3 + 1/2))^2 * pi
      23.2671801626
```

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

Hasil perhitungan ini adalah angka floating point. Secara default, angka ini dicetak dengan akurasi sekitar 12 digit. Pada baris perintah berikut, kita juga mempelajari cara 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 terdiri dari operator dan fungsi. Jika perlu, ekspresi harus berisi tanda kurung untuk memaksakan urutan

eksekusi yang benar. Jika ragu, sebaiknya gunakan tanda kurung. 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
*, /
. perkalian matriks
a^b pangkat untuk a positif atau integer b (a**b juga berfungsi)
n! operator faktorial

dan masih banyak lagi.

Berikut ini beberapa fungsi yang mungkin Anda perlukan. Masih banyak lagi.

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

Beberapa perintah memiliki alias, misalnya ln untuk log.

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

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

```
>2^3^4, (2^3)^4, 2^(3^4)
    2.41785163923e+24
    4096
    2.41785163923e+24
```

Bilangan Riil

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

```
>longest 1/7
         0.1428571428571428
```

Representasi ganda internal membutuhkan 8 byte.

```
>printdual(1/3)
    1.0101010101010101010101010101010101010101010101010101*2^-2
>printhex(1/3)
```

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

String

String dalam Euler didefinisikan dengan "...".

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

String dapat dirangkai dengan | atau dengan +. Ini juga berlaku untuk angka, yang dalam kasus tersebut diubah menjadi string.

```
>"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 mengonversi angka menjadi string. Fungsi ini dapat mengambil sejumlah digit dan sejumlah tempat (0 untuk keluaran padat), dan optimalnya satu unit.

```
>"Golden Ratio : " + print((1+sqrt(5))/2,5,0)
Golden Ratio : 1.61803
```

Ada string khusus none, yang tidak dicetak. String ini dikembalikan oleh beberapa fungsi, ketika hasilnya tidak penting. (Dikembalikan secara otomatis, jika fungsi tersebut tidak memiliki pernyataan return.)

```
>none
```

Untuk mengubah string menjadi angka, cukup evaluasi string tersebut. Ini juga berlaku untuk ekspresi (lihat di bawah).

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

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

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

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

```
>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 membuat string seperti itu, gunakan u"..." dan salah satu entitas HTML.

String Unicode dapat dirangkai seperti string lainnya.

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

I

Contoh Soal

Dalam komentar, entitas yang sama seperti alpha;, beta; dll. dapat digunakan. Ini mungkin merupakan alternatif cepat untuk Latex. (Rincian lebih lanjut pada komentar di bawah).

```
>a:=1; b:=2; c:=3; (a+b+c)^2
```
```
36
```

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

```
>v=strtochar(u"&Auml; is a German letter")
```
```
[196,  32,  105,  115,  32,  97,  32,  71,  101,  114,  109,  97,
110,
32,  108,  101,  116,  116,  101,  114]
```

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

```
>v[1]=strtochar(u"&Uuml;")[1]; chartoutf(v)
```
```
Ü is a German letter
```

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

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

Dimungkinkan juga untuk menggunakan entitas numerik.

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

Nilai Boolean

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

```
>2<1, "apel"<"banana"
```
```
0
1
```

"dan" adalah operator "&&" dan "atau" adalah operator "||", seperti dalam bahasa C. (Kata "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 sebuah vektor. Dalam contoh ini, kami menggunakan kondisional 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]
```

Format Keluaran

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

```
>defformat; pi
     3.14159265359
```

Secara internal, EMT menggunakan standar IEEE untuk angka ganda dengan sekitar 16 digit desimal. Untuk melihat jumlah digit lengkap, gunakan perintah "longestformat", atau kami menggunakan operator "longest" untuk menampilkan hasil dalam format terpanjang.

```
>longest pi
          3.141592653589793
```

Berikut adalah representasi heksadesimal internal dari angka 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
```

Format defaultnya adalah(12).

```
>format(12); 1/3
     0.333333333333
```

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

```
>shortestformat; random(5,7)
       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   0.53    0.5   0.17   0.26
  0.87   0.54   0.49    0.6   0.66   0.97   0.19
```

Format default untuk skalar adalah format(12). Namun, ini dapat diubah.

```
>setscalarformat(5); pi
     3.1416
```

Fungsi "longestformat" juga mengatur format skalar.

```
>longestformat; pi
     3.141592653589793
```

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

shortestformat shortformat longformat, longestformat

format(length,digits) goodformat(length)

fracformat(length)

defformat

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

Namun, format output EMT dapat diatur dengan cara yang fleksibel.

```
>longestformat; pi,
     3.141592653589793
>format(10,5); pi
       3.14159
```

Standarnya adalah defformat().

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

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

```
>longest pi^2/2
          4.934802200544679
```

Ada juga operator pendek untuk mencetak hasil dalam format pecahan. Kami telah 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 terwakili secara tepat. Kesalahannya bertambah sedikit, seperti yang Anda lihat dalam perhitungan berikut.

```
>longest 0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1-1
     -1.110223024625157e-16
```

Namun dengan "longformat" default, Anda tidak akan melihat hal ini. Demi kenyamanan, output angka yang sangat kecil adalah 0.

```
>0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1-1
     0
```

Ekspresi

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

Variabel global dapat digunakan dalam evaluasi.

```
>r:=2; fx:="pi*r^2"; longest fx()
           12.56637061435917
```

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

```
>fx:="a*sin(x)^2"; fx(5,a=-1)
     -0.919535764538
```

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

```
>at:=4; function f(expr,x,at) := expr(x); ...
 f("at*x^2",3,5) // computes 4*3^2 not 5*3^2
     36
```

Jika Anda ingin menggunakan nilai lain untuk "at" 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
```

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

```
>at:=4; function f(expr,x) := expr(x); ...
 f({{"at*x^2",at=5}},3)
     45
```

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

```
>f &= 5*x;
>function f(x) := 6*x;
>f(2)
     12
```

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

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

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

Bentuk khusus dari suatu ekspresi memperbolehkan variabel apa pun sebagai parameter tanpa nama untuk evaluasi ekspresi, bukan hanya "x", "y", dst. Untuk ini, awali ekspresi dengan "@(variabel) ...".

```
>"@(a,b) a^2+b^2", %(4,5)
     @(a,b) a^2+b^2
     41
```

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

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

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

```
>fx &= x^3-a*x;  ...
 a=1.2; fx(0.5)
     -0.475
```

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

```
>fx(0.5,a=1.1)
     -0.425
```

Suatu ekspresi tidak harus simbolis. Hal ini diperlukan, jika ekspresi tersebut memuat fungsi, yang hanya diketahui dalam kernel numerik, bukan dalam Maxima.

Matematika Simbolis

EMT mengerjakan matematika simbolis dengan bantuan Maxima. Untuk detailnya, mulailah dengan tutorial berikut, atau telusuri referensi untuk Maxima. Para ahli di Maxima harus memperhatikan bahwa terdapat perbedaan dalam sintaksis antara sintaksis asli Maxima dan sintaksis default ekspresi simbolis dalam EMT.

Matematika simbolis terintegrasi dengan mulus ke dalam Euler dengan &. Setiap ekspresi yang dimulai dengan & adalah ekspresi simbolis. Ekspresi tersebut dievaluasi dan dicetak oleh Maxima.

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

```
>$&44!
```

$$26582715747884487680436258110146158903196385280000000000$$

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

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

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

$$2481256778$$

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

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

$$2481256778$$

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

Anda akan mempelajari bahwa hal berikut juga berfungsi.

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

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

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

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

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

$$120$$

Dengan cara itu Anda dapat menggunakan solusi persamaan dalam persamaan lain.

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

Seperti yang telah Anda lihat pada contoh sebelumnya dan berikutnya, 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 tidak menginstal LaTeX.

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

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

Ekspresi simbolik diurai oleh Euler. Jika Anda memerlukan sintaksis yang kompleks dalam satu ekspresi, Anda dapat melampirkan ekspresi tersebut dalam "...". Menggunakan lebih dari satu ekspresi sederhana dimungkinkan, tetapi sangat tidak disarankan.

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

25

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

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

$$x^4+4\,x^3+6\,x^2+4\,x+1$$

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

Sekali lagi, % merujuk 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 perintah Maxima juga tersedia. Awali baris perintah dengan "::". Sintaks Maxima disesuaikan dengan sintaks EMT (disebut "mode kompatibilitas").

```
>&factor(30!)
```

265252859812191058636308480000000

```
>::: 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 Maxima. Anda dapat melakukannya dengan ":::".

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

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

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

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

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

$$125\, e^5$$

```
                                18551.64488782208
```

```
>fx(5)
     18551.6448878
```

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

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

```
>&(fx with x=10)-(fx with x=5), &float(%)
                                    10        5
                                1000 E   - 125 E


                                2.20079141499189e+7
```

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

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

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

```
>tex(fx)
```
```
x^3\,e^{x}
```

Ekspresi simbolik dapat dievaluasi seperti halnya ekspresi numerik.

```
>fx(0.5)
```
```
0.206090158838
```

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

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

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

Penugasan tersebut juga dapat bersifat simbolis.

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

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

Perintah solve memecahkan ekspresi simbolik untuk variabel dalam 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 secara opsional nilai target.

```
>solve("x^2+x",1,y=4)
```
```
1.56155281281
```

Nilai numerik dari solusi simbolik dapat dihitung dengan mengevaluasi hasil simbolik. Euler akan membaca ulang penugasan 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)
```

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

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

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

Untuk mendapatkan solusi simbolis yang spesifik, seseorang dapat menggunakan "with" dan indeks.

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

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

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

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]
```

$$[[x = 2, y = 1], [x = 1, y = 2]]$$

$$2$$

Ekspresi simbolik dapat memiliki tanda, yang menunjukkan perlakuan khusus di Maxima. Beberapa tanda dapat digunakan sebagai perintah juga, yang lainnya tidak. Tanda ditambahkan dengan "|" (bentuk yang lebih baik dari "ev(...,flags)")

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

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

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

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

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

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

Fungsi

Dalam EMT, fungsi adalah program yang didefinisikan dengan perintah "function". Fungsi ini dapat 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)
```

Sebagai gambaran umum, kami tampilkan semua definisi yang mungkin untuk fungsi satu baris. Suatu fungsi dapat dievaluasi seperti fungsi Euler bawaan lainnya.

```
>f(2)
     4.472135955
```

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

```
>f(0:0.1:1)
     [0,  0.100499,  0.203961,  0.313209,  0.430813,  0.559017,  0.699714,
     0.854459,  1.0245,  1.21083,  1.41421]
```

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

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

```
>solve("f",1,y=1)
     0.786151377757
```

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

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

```
>function overwrite sin (x) := _sin(x°) // redine sine in degrees
>sin(45)
     0.707106781187
```

Sebaiknya kita hilangkan pendefinisian ulang dosa ini.

```
>forget sin; sin(pi/4)
     0.707106781187
```

Parameter Default

Fungsi numerik dapat memiliki parameter default.

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

Mengabaikan parameter ini akan menggunakan nilai default.

```
>f(4)
```
```
      16
```

Mengaturnya akan menimpa nilai default.

```
>f(4,5)
```
```
      80
```

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

```
>f(4,a=1)
```
```
      16
```

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

```
>function f(x) := a*x^2
>a=6; f(2)
```
```
      24
```

Namun, parameter yang ditetapkan akan menggantikan nilai global.

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

```
>f(2,a:=5)
```
```
      20
```

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

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

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

Fungsi simbolik dapat digunakan dalam ekspresi simbolik.

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

$$x\,e^{-x} - e^{-x} + 3\,x^2$$

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

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

```
>g(5+g(1))
      178.635099908
```

Mereka dapat digunakan untuk mendefinisikan fungsi atau ekspresi simbolis lainnya.

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

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

```
>solve(&g(x),0.5)
      0.703467422498
```

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

```
>solve(&g,0.5)
      0.703467422498
>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(%)
```

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

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

```
>P(3,4)
      625
>$&P(x,4)+ Q(x,3), $&expand(%)
```

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

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

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

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

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

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

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

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

$$16\,x^7 + 64\,x^6 + 24\,x^5 - 120\,x^4 - 15\,x^3 + 102\,x^2 - 52\,x + 8$$

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

```
>$&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 simbolis lainnya.

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

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

Dengan := fungsinya bersifat numerik. Contoh yang bagus adalah integral tentu seperti

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

yang tidak dapat dievaluasi secara simbolis.

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

```
>function map f(x) := integrate("x^x",1,x)
>f(0:0.5:2)
     [-0.783431,  -0.410816,  0,  0.676863,  2.05045]
```

Fungsi dapat memiliki nilai default untuk parameter.

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

Sekarang fungsi tersebut dapat dipanggil dengan atau tanpa parameter "dasar".

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

Selain itu, dimungkinkan untuk menggunakan parameter yang ditetapkan.

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

Sering kali, 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)
```

$$b^2 - a\,b + b + a^2$$

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

Fungsi simbolik semacam itu dapat digunakan untuk variabel simbolik.

Namun, fungsi tersebut juga dapat digunakan untuk vektor numerik.

```
>v=[3,4]; f(v)
     17
```

Ada pula fungsi yang murni simbolis, 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)
```

$$y^4 - 6\,x^2\,y^2 + x^4$$

$$0$$

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)
```

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

Singkatnya

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

Menyelesaikan Ekspresi

Ekspresi dapat diselesaikan secara numerik dan simbolik.

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

```
>solve("x^2-2",1)
```
```
1.41421356237
```

Ini juga berlaku 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])
```

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

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

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

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

```
>solve(px,1,y=2), px(%)
      0.966715594851
      2
```

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

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

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

Cara termudah untuk mendapatkan nilai numerik adalah dengan mengevaluasi solusi secara numerik seperti sebuah 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])
```

$$\frac{\left(\sqrt{5}-1\right)^2}{4}$$

$$0$$

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

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

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

Fungsi f() dapat melihat variabel global. Namun, sering kali kita ingin menggunakan parameter lokal.

$$a^x-x^a=0,1$$

dengan a=3.

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

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

```
>solve({{"f",3}},2,y=0.1)
      2.54116291558
```

Ini juga berlaku untuk ekspresi. Namun, elemen daftar bernama harus digunakan. (Informasi lebih lanjut tentang daftar ada di tutorial tentang sintaks EMT).

```
>solve({{"x^a-a^x",a=3}},2,y=0.1)
      2.54116291558
```

Menyelesaikan Pertidaksamaan

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

```
>&load(fourier_elim)
            C:/Program Files/Euler x64/maxima/share/maxima/5.35.1/share/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])
```

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

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

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

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

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

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

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

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

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

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

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

```
>&fourier_elim([max(x,y) > 6, x # 8, abs(y-1) > 12],[x,y])
              [6 < x, x < 8, y < - 11] or [8 < x, y < - 11]
      or [x < 8, 13 < y] or [x = y, 13 < y] or [8 < x, x < y, 13 < y]
      or [y < x, 13 < y]
```

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

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

Bahasa Matriks

Dokumentasi inti EMT berisi pembahasan terperinci tentang bahasa matriks Euler.

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

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

Produk matriks dilambangkan dengan sebuah titik.

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

```
                4
>b' // transpose b
     [3,  4]
>inv(A) //inverse A
               -2              1
              1.5           -0.5
>A.b //perkalian matriks
               11
               25
>A.inv(A)
                1              0
                0              1
```

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

```
>A.A
                7             10
               15             22
>A^2 //perpangkatan elemen2 A
                1              4
                9             16
>A.A.A
               37             54
               81            118
>power(A,3) //perpangkatan matriks
               37             54
               81            118
>A/A //pembagian elemen-elemen matriks yang seletak
                1              1
                1              1
>A/b //pembagian elemen2 A oleh elemen2 b kolom demi kolom (karena b vektor
kolom)
          0.333333       0.666667
              0.75              1
>A\b // hasilkali invers A dan b, A^(-1)b
               -2
              2.5
>inv(A).b
               -2
              2.5
>A\A   //A^(-1)A
                1              0
                0              1
>inv(A).A
                1              0
                0              1
>A*A //perkalin elemen-elemen matriks seletak
                1              4
                9             16
```

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

```
>b^2 // perpangkatan elemen-elemen matriks/vektor
                9
               16
```

Jika salah satu operan merupakan vektor atau skalar, ia diekspansi dengan cara alami.

```
>2*A
                 2              4
                 6              8
```

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

```
>[1,2]*A
                 1              4
                 3              8
```

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

```
>A*[2,3]
                 2              6
                 6             12
```

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

```
>dup([1,2],2) // dup: menduplikasi/menggandakan vektor [1,2] sebanyak 2
kali (baris)
                 1              2
                 1              2
>A*dup([1,2],2)
                 1              4
                 3              8
```

Hal ini juga berlaku untuk dua vektor, yang satu merupakan vektor baris dan yang lainnya merupakan vektor kolom. Kita menghitung i*j untuk i,j dari 1 hingga 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
                 1              2              3              4              5
                 2              4              6              8             10
                 3              6              9             12             15
                 4              8             12             16             20
                 5             10             15             20             25
```

Sekali lagi, ingatlah bahwa ini bukan produk matriks!

```
>(1:5).(1:5)' // hasilkali vektor baris dan vektor kolom
     55
>sum((1:5)*(1:5)) // sama hasilnya
     55
```

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

```
>(1:10)<6 // menguji elemen-elemen yang kurang dari 6
     [1,  1,  1,  1,  1,  0,  0,  0,  0,  0]
```

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

```
>sum((1:10)<6) // banyak elemen yang kurang dari 6
     5
```

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

Kita memperoleh vektor 0 dan 1, di mana 1 berarti benar.

```
>t=(1:10)^2; t==25 //menguji elemen2 t yang sama dengan 25 (hanya ada 1)
     [0,  0,  0,  0,  1,  0,  0,  0,  0,  0]
```

Dari vektor tersebut, "nonzeros" memilih elemen yang bukan nol.

Dalam kasus ini, kita memperoleh indeks semua elemen yang lebih besar dari 50.

```
>nonzeros(t>50) //indeks elemen2 t yang lebih besar daripada 50
     [8,  9,  10]
```

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

```
>t[nonzeros(t>50)] //elemen2 t yang lebih besar daripada 50
     [64,  81,  100]
```

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

```
>t=1:1000; nonzeros(mod(t^2,11)==5 && mod(t^2,13)==3)
     [4,  48,  95,  139,  147,  191,  238,  282,  290,  334,  381,  425,
     433,  477,  524,  568,  576,  620,  667,  711,  719,  763,  810,  854,
     862,  906,  953,  997]
```

EMT tidak sepenuhnya efektif untuk komputasi integer. Ia menggunakan floating point presisi ganda secara internal. Namun, ia sering kali sangat berguna.

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

```
>t=1:1000; length(nonzeros(isprime(t^2+1)))
     112
```

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

```
>seed(2); A=random(3,4)
          0.765761      0.401188      0.406347      0.267829
           0.13673      0.390567      0.495975      0.952814
          0.548138      0.006085      0.444255      0.539246
```

Ia mengembalikan indeks elemen, yang bukan nol.

```
>k=mnonzeros(A<0.4) //indeks elemen2 A yang kurang dari 0,4
                 1             4
                 2             1
                 2             2
                 3             2
```

Indeks ini dapat digunakan untuk menetapkan elemen pada nilai tertentu.

```
>mset(A,k,0) //mengganti elemen2 suatu matriks pada indeks tertentu
          0.765761      0.401188      0.406347             0
                 0             0      0.495975      0.952814
```

```
         0.548138             0      0.444255      0.539246
```

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

```
>mset(A,k,-random(size(A)))
         0.765761      0.401188      0.406347     -0.126917
        -0.122404     -0.691673      0.495975      0.952814
         0.548138     -0.483902      0.444255      0.539246
```

Dan adalah mungkin untuk mendapatkan unsur-unsur dalam sebuah vektor.

```
>mget(A,k)
    [0.267829,  0.13673,  0.390567,  0.006085]
```

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

```
>ex=extrema(A)
         0.267829             4      0.765761             1
          0.13673             1      0.952814             4
         0.006085             2      0.548138             1
```

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

```
>ex[,3]'
    [0.765761,  0.952814,  0.548138]
```

Ini tentu saja sama dengan fungsi max().

```
>max(A)'
    [0.765761,  0.952814,  0.548138]
```

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

```
>j=(1:rows(A))'|ex[,4], mget(-A,j)
                 1             1
                 2             4
                 3             1
    [-0.765761,  -0.952814,  -0.548138]
```

Fungsi Matriks Lainnya (Membangun Matriks)

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

```
>v=1:3; v_v
                 1             2             3
                 1             2             3
```

Dengan cara yang sama, kita dapat menempelkan suatu matriks ke sisi lain yang berdampingan, jika keduanya memiliki jumlah baris yang sama.

```
>A=random(3,4); A|v'
         0.032444     0.0534171      0.595713      0.564454             1
```

```
          0.83916       0.175552       0.396988        0.83514              2
        0.0257573       0.658585       0.629832       0.770895              3
```

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

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

```
>A|1
          0.032444      0.0534171       0.595713       0.564454              1
           0.83916       0.175552       0.396988        0.83514              1
         0.0257573       0.658585       0.629832       0.770895              1
```

Dimungkinkan untuk membuat matriks dari vektor baris dan kolom.

```
>[v;v]
                  1              2              3
                  1              2              3
>[v',v']
                  1              1
                  2              2
                  3              3
```

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

```
>"[x,x^2]"(v')
                  1              1
                  2              4
                  3              9
```

Untuk mendapatkan ukuran A, kita dapat menggunakan fungsi berikut.

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

Untuk vektor, ada length().

```
>length(2:10)
     9
```

Ada banyak fungsi lain yang menghasilkan matriks.

```
>ones(2,2)
                  1              1
                  1              1
```

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

```
>ones(5)*6
     [6,  6,  6,  6,  6]
```

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

```
>random(2,2)
           0.66566       0.831835
             0.977       0.544258
```

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

```
>redim(1:9,3,3) // menyusun elemen2 1, 2, 3, ..., 9 ke bentuk matriks 3x3
                 1             2             3
                 4             5             6
                 7             8             9
```

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

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

Mari kita menguji.

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

Fungsi multdup() menduplikasi elemen suatu vektor.

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

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

```
>flipx(1:5) //membalik elemen2 vektor baris
     [5,  4,  3,  2,  1]
```

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

```
>rotleft(1:5) // memutar elemen2 vektor baris
     [2,  3,  4,  5,  1]
```

Fungsi khusus adalah drop(v,i), yang menghapus elemen dengan indeks di i dari vektor v.

```
>drop(10:20,3)
     [10,  11,  13,  14,  15,  16,  17,  18,  19,  20]
```

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

```
>v=primes(50), i=indexof(v,10:20), drop(v,i)
     [2,  3,  5,  7,  11,  13,  17,  19,  23,  29,  31,  37,  41,  43,  47]
     [0,  5,  0,  6,  0,  0,  0,  7,  0,  8,  0]
     [2,  3,  5,  7,  23,  29,  31,  37,  41,  43,  47]
```

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

```
>drop(1:10,shuffle([0,0,5,5,7,12,12]))
     [1,  2,  3,  4,  6,  8,  9,  10]
```

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

Kita mulai dengan matriks identitas.

```
>A=id(5) // matriks identitas 5x5
                1              0              0              0              0
                0              1              0              0              0
                0              0              1              0              0
                0              0              0              1              0
                0              0              0              0              1
```

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

```
>setdiag(A,-1,1:4) //mengganti diagonal di bawah diagonal utama
                1              0              0              0              0
                1              1              0              0              0
                0              2              1              0              0
                0              0              3              1              0
                0              0              0              4              1
```

Perhatikan bahwa kita tidak mengubah matriks A. Kita mendapatkan matriks baru sebagai hasil dari setdiag().

Berikut ini adalah fungsi yang mengembalikan matriks tri-diagonal.

```
>function tridiag (n,a,b,c) := setdiag(setdiag(b*id(n),1,c),-1,a); ...
 tridiag(5,1,2,3)
                2              3              0              0              0
                1              2              3              0              0
                0              1              2              3              0
                0              0              1              2              3
                0              0              0              1              2
```

Diagonal matriks juga dapat diekstraksi dari matriks. Untuk menunjukkan hal ini, kami merestrukturisasi vektor 1:9 menjadi matriks 3x3.

```
>A=redim(1:9,3,3)
                1              2              3
                4              5              6
                7              8              9
```

Sekarang kita dapat mengekstrak diagonalnya.

```
>d=getdiag(A,0)
     [1,  5,  9]
```

Misalnya, kita dapat membagi matriks berdasarkan diagonalnya. Bahasa matriks memastikan bahwa vektor kolom d diterapkan ke matriks baris demi baris.

```
>fraction A/d'
              1          2          3
            4/5          1        6/5
            7/9        8/9          1
```

Vektorisasi

Hampir semua fungsi di Euler juga berfungsi untuk masukan matriks dan vektor, jika ini masuk akal.

Misalnya, fungsi sqrt() menghitung akar kuadrat dari semua elemen vektor atau matriks.

```
>sqrt(1:3)
     [1,  1.41421,  1.73205]
```

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

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

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

Dalam contoh berikut, kita membuat vektor nilai t[i] dengan spasi 0,1 dari -1 hingga 1. Kemudian kita membuat vektor nilai fungsi latex: s = t^3-t

```
>t=-1:0.1:1; s=t^3-t
     [0,  0.171,  0.288,  0.357,  0.384,  0.375,  0.336,  0.273,  0.192,
     0.099,  0,  -0.099,  -0.192,  -0.273,  -0.336,  -0.375,  -0.384,
     -0.357,  -0.288,  -0.171,  0]
```

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

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

```
>shortest (1:5)*(1:5)'
          1       2       3       4       5
          2       4       6       8      10
          3       6       9      12      15
          4       8      12      16      20
          5      10      15      20      25
```

Perhatikan bahwa ini sangat berbeda dari perkalian matriks. Perkalian matriks dilambangkan dengan titik "." dalam EMT.

```
>(1:5).(1:5)'
     55
```

Secara default, vektor baris dicetak dalam format ringkas.

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

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

```
>v:=[1,2]; v.v', %^2
     5
     25
```

Untuk mentranspos suatu matriks, kita menggunakan tanda apostrof.

```
>v=1:4; v'
                  1
                  2
                  3
                  4
```

Jadi kita dapat menghitung matriks A dikali vektor b.

```
>A=[1,2,3,4;5,6,7,8]; A.v'
                 30
                 70
```

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

```
>v'.v
                  1                2                3                4
                  2                4                6                8
                  3                6                9               12
                  4                8               12               16
```

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

```
>v.v'
     30
```

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

```
>norm(v)^2
     30
```

Operator dan fungsi mematuhi bahasa matriks Euler.

Berikut ini ringkasan aturannya.

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

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

Berikut ini adalah kasus sederhana dengan operator ^.

```
>[1,2,3]^2
     [1,  4,  9]
```

Berikut ini adalah kasus yang lebih rumit. Vektor baris dikalikan vektor kolom, keduanya diekspansi dengan cara menduplikasi.

```
>v:=[1,2,3]; v*v'
                  1              2              3
                  2              4              6
                  3              6              9
```

Perhatikan bahwa produk skalar menggunakan produk matriks, bukan *!

```
>v.v'
     14
```

Ada banyak fungsi untuk matriks. Kami memberikan daftar singkatnya. Anda harus merujuk ke dokumentasi untuk informasi lebih lanjut tentang perintah-perintah ini.

sum,prod menghitung jumlah dan hasil perkalian baris-baris

cumsum,cumprod melakukan hal yang sama secara kumulatif

menghitung nilai ekstrem dari setiap baris

extrema mengembalikan vektor dengan informasi ekstrem

diag(A,i) mengembalikan diagonal ke-i

setdiag(A,i,v) menetapkan diagonal ke-i

id(n) matriks identitas

det(A) determinan

charpoly(A) polinomial karakteristik

eigenvalues(A) nilai eigen

```
>v*v, sum(v*v), cumsum(v*v)
     [1,  4,  9]
     14
     [1,  5,  14]
```

The : operator generates an equally spaces row vector, optionally with a step size.

```
>1:4, 1:2:10
     [1,  2,  3,  4]
     [1,  3,  5,  7,  9]
```

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

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

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

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

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

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

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

```
>v[1:2], A[:,2]
     [2,  4]
                 2
                 5
                 8
```

Bentuk singkat dari : adalah menghilangkan indeks sepenuhnya.

```
>A[,2:3]
                 2              3
                 5              6
                 8              9
```

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

```
>A{5}
     5
```

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

```
>redim(A,1,prod(size(A))), flatten(A)
     [1,  2,  3,  4,  5,  6,  7,  8,  9]
     [1,  2,  3,  4,  5,  6,  7,  8,  9]
```

Untuk menggunakan matriks pada tabel, mari kita atur ulang ke format default, dan hitung tabel nilai sinus dan kosinus. Perhatikan bahwa sudut dalam radian secara default.

```
>defformat; w=0°:45°:360°; w=w'; deg(w)
                 0
                45
                90
               135
               180
               225
               270
               315
               360
```

Sekarang kita tambahkan 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 membuat beberapa tabel dari beberapa fungsi sekaligus.

Dalam contoh berikut, kita menghitung t[j]^i untuk i dari 1 hingga n. Kita memperoleh matriks, yang setiap barisnya merupakan tabel t^i untuk satu i. Yaitu, 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 input vektor harus "divektorkan". Ini dapat dicapai dengan kata kunci "map" dalam definisi fungsi. Kemudian fungsi tersebut akan dievaluasi untuk setiap elemen parameter vektor.

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

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

Kata kunci "map" akan memvektorkan fungsi tersebut. Fungsi tersebut sekarang akan berfungsi
untuk vektor angka.

```
>f([1:5])
     [0,  2.05045,  13.7251,  113.336,  1241.03]
```

Sub-Matriks dan Elemen Matriks

Untuk mengakses elemen matriks, gunakan notasi tanda kurung.

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

Kita dapat mengakses baris matriks yang lengkap.

```
>A[3]
     [7,  8,  9]
```

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

```
>v=1:8; v[2]
     2
```

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

```
>A[2,3]
     6
```

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

Di sini kita menginginkan baris pertama dan kedua dari A.

```
>A[[1,3]]
                1               2               3
                7               8               9
```

Kita bahkan dapat menyusun ulang A menggunakan vektor indeks. Untuk lebih tepatnya, kita tidak mengubah A di sini, tetapi menghitung versi A yang telah disusun ulang.

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

Trik indeks juga berfungsi dengan kolom.

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

```
>A[1:2,2:3]
                2               3
                5               6
```

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

```
>A[:,2]
                2
                5
                8
```

Atau, biarkan indeks pertama kosong.

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

Kita juga bisa mendapatkan baris terakhir A.

```
>A[3]
     [7,  8,  9]
```

Sekarang mari kita ubah elemen A dengan menetapkan submatriks A ke suatu nilai. Hal ini pada kenyataannya mengubah matriks A yang tersimpan.

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

Kita juga dapat menetapkan nilai ke baris A.

```
>A[1]=[-1,-1,-1]
               -1              -1              -1
                4               5               9
                7               8               9
```

Kita bahkan dapat menetapkannya ke submatriks jika ukurannya tepat.

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

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

Selain itu, beberapa jalan pintas diperbolehkan.

```
>A[1:2,1:2]=-1
               -1             -1             -1
               -1             -1              9
                7              8              9
```

Peringatan: Indeks yang tidak sesuai batas akan mengembalikan matriks kosong, atau pesan kesalahan, tergantung pada pengaturan sistem. Pesan kesalahan adalah standar. Namun, perlu diingat bahwa indeks negatif dapat digunakan untuk mengakses elemen matriks yang dihitung dari akhir.

```
>A[5]
     Row index 5 out of bounds!
     Error in:
     A[5] ...
         ^
```

Sortir dan Acak

Fungsi sort() mengurutkan vektor baris.

```
>sort([2,9,5,7,3,1])
     [1,  2,  3,  5,  7,  9]
```

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

Mari kita acak sebuah vektor.

```
>v=shuffle(1:8)
     [4,  3,  5,  7,  1,  6,  8,  2]
```

Indeks berisi urutan v yang tepat.

```
>{vs,ind}=sort(v); v[ind]
     [1,  2,  3,  4,  5,  6,  7,  8]
```

Ini juga berlaku untuk vektor string.

```
>s=["a","d","e","a","aa","g"]
     a
     d
     e
     a
     aa
     g
>{ss,ind}=sort(s); ss
     a
     a
     aa
     d
     e
```

```
g
```

Seperti yang Anda lihat, posisi entri ganda agak acak.

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

Fungsi unik mengembalikan daftar yang diurutkan dari elemen unik suatu vektor.

```
>intrandom(1,10,10), unique(%)
     [8,  1,  4,  4,  9,  2,  6,  5,  10,  6]
     [1,  2,  4,  5,  6,  8,  9,  10]
```

Ini juga berlaku untuk vektor string.

```
>unique(s)
     a
     aa
     d
     e
     g
```

Aljabar Linier

EMT memiliki banyak fungsi untuk memecahkan sistem linier, sistem renggang, atau masalah regresi.

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

```
>A=[5,6;7,8]; b=[4;3]; A\b
                -7
               6.5
```

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

```
>A=normal(100,100); b=sum(A); longest totalmax(abs(inv(A).b-1))
       8.992806499463768e-14
```

Jika sistem tidak mempunyai solusi, penyesuaian linier meminimalkan norma kesalahan Ax-b.

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

Determinan matriks ini adalah -1.

```
>det(A)
     -1
```

Matriks Simbolik

Maxima memiliki matriks simbolik. Tentu saja, Maxima dapat digunakan untuk masalah aljabar linear sederhana tersebut. Kita dapat mendefinisikan matriks untuk Euler dan Maxima dengan &:=, lalu menggunakannya dalam ekspresi simbolik. Bentuk [...] yang biasa digunakan untuk mendefinisikan matriks dapat digunakan dalam 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(%)
```

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

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

```
>$&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)
```

$$(1-x)\,(2-x)-a\,b$$

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

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

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

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

Untuk mengekstrak vektor eigen tertentu dibutuhkan pengindeksan yang cermat.

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

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

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

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

```
>A(a=6,b=8)
                  1                  6
                  8                  2
```

Dalam ekspresi simbolik, gunakan dengan.

```
>$&A with [a=6,b=8]
```

$$\begin{pmatrix}1 & 6\\ 8 & 2\end{pmatrix}$$

Akses terhadap baris matriks simbolik bekerja seperti halnya matriks numerik.

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

$$[1,a]$$

Ekspresi simbolik dapat berisi sebuah penugasan. Dan itu mengubah matriks A.

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

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

Terdapat fungsi simbolik di Maxima untuk membuat vektor dan matriks. Untuk ini, rujuk 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)
```

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

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

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

```
>$&invert(B)()
                -2              1
               1.5           -0.5
```

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

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

```
>longest B.xinv(B)
                              1                             0
                              0                             1
```

Misalnya nilai eigen A dapat dihitung secara numerik.

```
>A=[1,2,3;4,5,6;7,8,9]; real(eigenvalues(A))
     [16.1168,  -1.11684,  0]
```

Atau secara simbolis. Lihat tutorial tentang Maxima untuk detailnya.

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

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

Nilai Numerik dalam Ekspresi Simbolik

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

```
>A &:= [1,pi;4,5]
                  1        3.14159
                  4              5
```

Masih terdapat perbedaan antara bentuk numerik dan bentuk simbolik. Saat mengubah matriks ke bentuk simbolik, pendekatan pecahan untuk bilangan riil akan digunakan.

```
>$&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 juga dapat melakukan komputasi dengan angka floating point, dan bahkan dengan angka floating point besar dengan 32 digit. Namun, evaluasinya jauh lebih lambat.

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

$$1.4142135623730950488016887242097_B \times 10^0$$

$$1.41421356237309$$

Ketepatan angka floating point besar dapat diubah.

```
>&fpprec:=100; &bfloat(pi)
             3.14159265358979323846264338327950288419716939937510582097494\
     459230781640628620899862803482534211706 8b0
```

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

Perlu dicatat bahwa ini hanya diperlukan jika variabel telah didefinisikan dengan ":=" atau "=" sebagai variabel numerik.

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

$$-5.424777960769379$$

Demo - Suku Bunga

Di bawah ini, kami menggunakan Euler Math Toolbox (EMT) untuk 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 5000 (misalnya dalam dolar).

```
>K=5000
     5000
```

Sekarang kita asumsikan suku bunga 3% per tahun. Mari kita tambahkan satu suku bunga sederhana dan hitung hasilnya.

```
>K*1.03
     5150
```

Euler juga akan memahami sintaksis berikut.

```
>K+K*3%
     5150
```

Namun lebih mudah menggunakan faktor

```
>q=1+3%, K*q
     1.03
     5150
```

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

```
>K*q^10
     6719.58189672
```

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

```
>format(12,2); K*q^10
         6719.58
```

Mari kita cetak angka tersebut dibulatkan menjadi 2 digit dalam kalimat lengkap.

```
>"Starting from " + K + "$ you get " + round(K*q^10,2) + "$."
     Starting from 5000$ you get 6719.58$.
```

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

```
>K*q^(0:10)
     Real 1 x 11 matrix

         5000.00      5150.00      5304.50      5463.64      ...
```

Bagaimana keajaiban ini bekerja? Pertama, ekspresi 0:10 menghasilkan vektor bilangan bulat.

```
>short 0:10
     [0,  1,  2,  3,  4,  5,  6,  7,  8,  9,  10]
```

Maka semua operator dan fungsi di Euler dapat diaplikasikan ke 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 hingga q^10. Ini dikalikan dengan K, dan kita memperoleh vektor nilai.

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

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

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

Mari kita bandingkan kedua hasil, dengan dan tanpa pembulatan.

```
>longest oneyear(1234.57), longest 1234.57*q
                   1271.61
                 1271.6071
```

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

Cara termudah adalah fungsi iterate, yang mengulang fungsi yang diberikan beberapa kali.

```
>VKr=iterate("oneyear",5000,10)
     Real 1 x 11 matrix

         5000.00      5150.00      5304.50      5463.64      ...
```

Kita 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, kita menggunakan indeks dalam tanda kurung siku.

```
>VKr[2], VKr[1:3]
         5150.00
         5000.00      5150.00      5304.50
```

Anehnya, kita juga dapat menggunakan vektor indeks. Ingat bahwa 1:3 menghasilkan vektor [1,2,3].

Mari kita bandingkan elemen terakhir dari nilai yang dibulatkan dengan nilai penuh.

```
>VKr[-1], VK[-1]
         6719.60
         6719.58
```

Perbedaannya sangat kecil.

Menyelesaikan Persamaan

Sekarang kita ambil fungsi yang lebih maju, yang menambahkan nilai uang tertentu setiap tahun.

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

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

```
>R=200; iterate("onepay",5000,10)
     Real 1 x 11 matrix

          5000.00      5350.00      5710.50      6081.82      ...
```

Bagaimana jika kita menghilangkan jumlah yang sama setiap tahun?

```
>R=-200; iterate("onepay",5000,10)
     Real 1 x 11 matrix

          5000.00      4950.00      4898.50      4845.45      ...
```

Kita melihat bahwa uang berkurang. Jelas, jika kita hanya memperoleh bunga sebesar 150 pada tahun pertama, tetapi mengurangi 200, kita akan kehilangan uang setiap tahun.

Bagaimana kita dapat menentukan berapa tahun uang tersebut akan bertahan? Kita harus menulis sebuah loop untuk ini. Cara termudah adalah dengan melakukan iterasi yang 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 nonzeros(VKR<0) mengembalikan vektor indeks i, di mana VKR[i]<0, dan min menghitung indeks minimal.

Karena vektor selalu dimulai dengan indeks 1, jawabannya adalah 47 tahun.

Fungsi iterate() memiliki satu trik lagi. Fungsi ini dapat mengambil kondisi akhir sebagai argumen. Kemudian, fungsi ini akan mengembalikan nilai dan jumlah iterasi.

```
>{x,n}=iterate("onepay",5000,till="x<0"); x, n,
          -19.83
           47.00
```

Mari kita coba menjawab pertanyaan yang lebih ambigu. Asumsikan kita tahu bahwa nilainya adalah 0 setelah 50 tahun. Berapa tingkat bunganya?

Ini adalah pertanyaan yang hanya dapat dijawab secara numerik. Di bawah ini, kita akan memperoleh rumus yang diperlukan. Kemudian Anda akan melihat bahwa tidak ada rumus yang mudah untuk tingkat bunga. Namun untuk saat ini, kita bertujuan untuk mencari solusi numerik.

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

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

Iterasinya sama seperti di atas

Namun, kita tidak lagi menggunakan nilai global R dalam ekspresi kita. Fungsi seperti iterate() memiliki trik khusus di Euler. Anda dapat meneruskan nilai variabel dalam ekspresi sebagai parameter titik koma. Dalam kasus ini P dan R.

Selain itu, kita hanya tertarik pada nilai terakhir. Jadi, kita ambil indeks [-1].

Mari kita coba uji coba.

```
>f(5000,-200,3,47)
          -19.83
```

Sekarang kita bisa memecahkan masalah kita.

```
>solve("f(5000,-200,x,50)",3)
            3.15
```

Rutin solve menyelesaikan ekspresi=0 untuk variabel x. Jawabannya adalah 3,15% per tahun. Kita ambil nilai awal 3% untuk algoritma tersebut. Fungsi solve() selalu membutuhkan nilai awal.

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

```
>solve("f(5000,x,3,20)",-200)
         -336.08
```

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

Solusi Simbolis untuk Masalah Suku Bunga

Kita dapat menggunakan bagian simbolis Euler untuk mempelajari masalah tersebut. Pertama, kita mendefinisikan fungsi onepay() secara simbolis.

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

$$R + q\,K$$

Sekarang kita dapat mengulanginya.

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

$$q\,\left(q\,\left(q\,\left(R+q\,K\right)+R\right)+R\right)+R$$

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

Kita melihat suatu pola. Setelah n periode kita memiliki

Rumus tersebut adalah rumus untuk jumlah geometrik, yang diketahui oleh 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 mereduksinya menjadi hasil bagi.

Mari kita buat 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 sebelumnya. Namun, fungsinya lebih efektif.

```
>longest f(5000,-200,3,47), longest fs(5000,-200,3,47)
          -19.82504734650985
          -19.82504734652684
```

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

```
>solve("fs(5000,-330,3,x)",30)
           20.51
```

Jawaban ini menyatakan bahwa akan negatif setelah 21 tahun.

Kita juga dapat menggunakan sisi simbolik Euler untuk menghitung rumus pembayaran.

Asumsikan kita mendapatkan pinjaman sebesar K, dan membayar n kali cicilan sebesar R (dimulai setelah tahun pertama) sehingga menyisakan utang residual sebesar Kn (pada saat pembayaran terakhir). Rumus untuk ini jelas

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

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

Biasanya rumus ini diberikan dalam bentuk lateks: i = \frac{P}{100}

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

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

Kita dapat mencari laju R secara simbolis.

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

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

Seperti yang dapat Anda lihat dari rumus, fungsi ini mengembalikan kesalahan floating point untuk i=0. Namun, Euler memplotnya.

Tentu saja, kita memiliki limit berikut.

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

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

Jelas, tanpa bunga, kita harus membayar kembali 10 suku bunga sebesar 500.

Persamaan ini juga dapat diselesaikan untuk n. Akan terlihat lebih bagus jika kita menerapkan beberapa penyederhanaan.

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

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

```
>a =1
                1.00
>$& a
```

$$a$$