TiEMT 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 contohcontoh 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))))
```

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

The Command Line

Sebuah baris perintah Euler terdiri dari satu atau lebih perintah Euler yang diikuti oleh titik koma ";" atau koma ",". Titik koma mencegah pencetakan hasil. Koma setelah perintah terakhir dapat dihilangkan. Baris perintah tersebut hanya akan mencetak hasil ekspresi, bukan penugasan atau perintah format.

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

16.7551608191

Perintah-perintah harus dipisahkan dengan spasi. Baris perintah tersebut mencetak kedua 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 pengguna menekan tombol enter. Jadi, Anda mendapatkan nilai baru setiap kali mengeksekusi baris kedua.

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

0.540302305868

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

0.857553215846

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

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

1.41666666667
1.41421568627
1.41421356237

Ini juga merupakan cara yang baik untuk menyebarkan perintah panjang ke dua atau lebih baris. Anda dapat menekan Ctrl+Return untuk memisahkan baris menjadi dua pada posisi kursor saat ini, atau Ctlr+Back untuk menggabungkan baris. Untuk melipat semua baris multi-baris, tekan Ctrl+L. Kemudian baris-baris berikutnya hanya akan terlihat jika salah satunya memiliki fokus. Untuk melipat satu baris multi-baris, mulai baris pertama dengan "%+ ".

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

Baris yang dimulai dengan %% akan sepenuhnya tidak terlihat.

81

Euler mendukung loop dalam baris perintah, selama loop tersebut muat dalam satu baris tunggal atau multi-baris. Dalam program, tentu saja batasan ini tidak berlaku. Untuk informasi lebih lanjut, silakan baca 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
```

Tiddak apa apa untuk menggunakan beberapa baris perintah. Pastikan untuk mengakhiri baris- baris perintah 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!
```

Ketika Anda mengeksekusi perintah, kursor dapat berada di posisi mana saja dalam 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 pergi ke perintah tersebut. Ketika Anda menggerakkan kursor di sepanjang baris, pasangan kurung buka dan tutup atau tanda kurung akan disorot. Perhatikan juga status line. Setelah kurung buka dari fungsi sqrt(), status line akan menampilkan teks bantuan untuk fungsi tersebut. Eksekusi perintah dengan tombol return.

```
>sqrt(sin(10°)/cos(20°))
```

```
0.429875017772
```

Untuk melihat bantuan untuk perintah terakhir, 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 dalam 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 mouse atau gunakan shift bersamaan dengan tombol panah apa pun. Selain itu, Anda dapat menyalin kurung yang disorot.

Basic Syntax

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

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

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

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

30.65625

enter 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 mengatur kurung yang benar dan menggunakan / untuk pecahan. Perhatikan kurung yang disorot untuk bantuan. Perhatikan bahwa konstanta Euler e dinamai E dalam EMT.

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

8.77908249441

Untuk menghitung ekspresi yang 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 kurung dengan hati-hati di sekitar sub-ekspresi yang perlu dihitung terlebih dahulu. EMT membantu Anda dengan menyorot ekspresi yang ditutup oleh kurung penutup. Anda juga harus memasukkan nama "pi" untuk huruf pi Yunani. Hasil perhitungan ini adalah bilangan floating point. Secara default, hasil ini dicetak dengan akurasi sekitar 12 digit. Dalam baris perintah berikut, kita juga belajar bagaimana kita dapat 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. Sebuah ekspresi terdiri dari operator dan fungsi. Jika diperlukan, ekspresi tersebut harus mengandung kurung untuk memaksa urutan eksekusi yang benar. Jika ragu, memasang kurung adalah ide yang baik. Perhatikan bahwa EMT menunjukkan kurung buka dan tutup saat mengedit baris perintah.

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

14.4978445072

Operator numerik Euler termasuk

+ unary or operator plus
- unary or operator minus
*, /
. the matrix product
a^b power for positive a or integer b (a**b works too) n! the factorial operator

Dan masih banyak lagi.

Berikut adalah beberapa fungsi yang mungkin Anda perlukan. Ada 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 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 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 Asli

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

```
>longest 1/3
```

```
0.3333333333333333
```

Representasi dual internalnya memakan 8 byte.

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

1.0101010101010101010101010101010101010101010101010101*2^-2

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

5.5555555555554*16^-1

Strings

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

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

A string can contain anything.

String dapat dikonkatenasi dengan | atau dengan +. Hal ini juga berlaku dengan angka, yang dikonversi menjadi string dalam hal tersebut.

```
>"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 print juga mengonversi angka menjadi string. Fungsi ini dapat menerima sejumlah digit dan sejumlah tempat (0 untuk keluaran padat), dan secara optimal sebuah 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. (String ini dikembalikan secara otomatis jika fungsi tersebut tidak memiliki pernyataan return.)

```
>none
```

Untuk mengubah string menjadi angka, cukup evaluasi saja. Hal 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

The empty string vector is denoted by [none]. String vectors can be concatenated.

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

affe
charlie
bravo affe
charlie bravo
Strings can
contain
Unicode
characters.
Internally,
these strings
contain UTF-
8 code. To
generate
such a
string, use
u"..." and
one of the
HTML
entities.

String Unicode dapat dikonkatenasikan seperti string lainnya.

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

± = 45°

I

Dalam komentar, entitas yang sama seperti &, dll. dapat digunakan. Ini mungkin merupakan alternatif cepat untuk Latex. (Detail lebih lanjut tentang komentar di bawah).

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

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

[196, 32, 105, 115, 32, 97, 32, 71, 101, 114, 109, 97, 110,
32, 108, 101, 116, 116, 101, 114]

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

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

Ü is a German letter

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

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

We have ±=.

Juga dimungkinkan untuk menggunakan entitas numerik.

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

Ähnliches

Nilai Boolean

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

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

0
1

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

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

1

Operator Boolean mengikuti aturan-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-elemen tertentu dari sebuah vektor. Dalam contoh, kami 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]

Format Output

Format keluaran default EMT mencetak 12 digit. Untuk memastikan bahwa kita melihat defaultnya, kita mereset formatnya.

```
>defformat; pi
```

```
3.14159265359
```

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

```
>longest pi
```

```
3.141592653589793
```

Berikut adalah representasi heksadesimal internal dari sebuah bilangan double.

```
>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
```

defaultnya adalah format(12).

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

```
0.333333333333
```

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

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

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

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

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

3.1416

Fungsi "longestformat" juga mengatur format skalar.

```
>longestformat; pi
```

3.141592653589793

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

shortestformat shortformat longformat, longestformat format(length,digits)
goodformat(length) fracformat(length) defformat

Akurasi internal EMT adalah sekitar 16 desimal, yang merupakan standar IEEE. Angka-angka disimpan dalam format internal ini. Namun, format keluaran EMT dapat diatur secara fleksibel.

```
>longestformat; pi,
```

3.141592653589793

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

3.14159

Format default adalah defformat().

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

Ada operator singkat yang hanya mencetak satu nilai. Operator "longest" akan mencetak semua digit yang valid dari suatu angka.

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

4.934802200544679

Ada juga operator singkat 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 diwakili secara tepat. Kesalahan tersebut bertambah sedikit demi 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 memperhatikan hal ini. 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

Ekspresi Matematika

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

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

12.56637061435917

Parameter-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 menghasilkan 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=nilai".

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

45

Sebagai referensi, kami mencatat bahwa kumpulan panggilan (dibahas di tempat lain) dapat berisi ekspresi. Jadi, kami 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. Perhatikan bahwa mendefinisikan fungsi dengan nama yang sama seperti ekspresi simbolik global menghapus variabel ini untuk menghindari kebingungan antara ekspresi simbolik dan fungsi.

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

12

Sebagai konvensi, ekspresi simbolik atau numerik harus dinamai fx, fxy, dll. Skema penamaan ini tidak boleh digunakan untuk fungsi.

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

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

Bentuk khusus dari sebuah ekspresi memungkinkan variabel apa pun sebagai parameter tanpa nama untuk evaluasi ekspresi tersebut, 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

Hal ini memungkinkan untuk memanipulasi ekspresi dalam variabel lain untuk fungsi-fungsi EMT yang membutuhkan ekspresi dalam "x". Cara paling dasar untuk mendefinisikan fungsi sederhana adalah dengan menyimpan rumusnya dalam ekspresi simbolik atau numerik. Jika variabel utamanya adalah x, ekspresi tersebut dapat dievaluasi seperti fungsi.

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

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

-0.475

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

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

-0.425

Suatu ekspresi tidak perlu bersifat simbolik. Hal ini diperlukan jika ekspresi tersebut mengandung fungsi-fungsi yang hanya diketahui dalam kernel numerik, bukan dalam Maxima.

Symbolik Matematika

Untuk detail lebih lanjut, mulailah dengan tutorial berikut atau jelajahi referensi untuk Maxima. Para ahli Maxima harus memperhatikan bahwa ada perbedaan dalam sintaks antara sintaks asli Maxima dan sintaks default ekspresi simbolik di EMT.

Matematika simbolik diintegrasikan secara mulus ke dalam Euler dengan simbol &. Setiap ekspresi yang dimulai dengan & adalah ekspresi simbolik. Ekspresi tersebut dievaluasi dan dicetak oleh Maxima.

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

```
>$&44!
```

$$2658271574788448768043625811014615890319638528000000000$$

Dengan cara ini, Anda dapat menghitung hasil 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 di atasnya. Misalnya, coba klik dua kali pada "&binomial" di baris perintah sebelumnya. Ini membuka dokumentasi Maxima sebagaimana disediakan oleh penulis program tersebut. Anda akan mengetahui 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 spesifik apa pun, gunakan "with".

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

$$120$$

Dengan cara ini Anda dapat menggunakan solusi dari suatu persamaan dalam persamaan lain. Ekspresi simbolik dicetak oleh Maxima dalam bentuk 2D. Alasan untuk ini adalah bendera simbolik khusus dalam string. Seperti yang telah Anda lihat dalam contoh-contoh sebelumnya dan selanjutnya, jika Anda memiliki LaTeX yang terinstal, 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 menjalankan perintah Maxima dengan $, jika Anda tidak memiliki LaTeX yang terinstal.

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

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

Ekspresi simbolik diuraikan oleh Euler. Jika Anda memerlukan sintaks yang kompleks dalam satu ekspresi, Anda dapat mengapit ekspresi tersebut dalam "...". Menggunakan lebih dari satu ekspresi sederhana dimungkinkan, tetapi sangat tidak dianjurkan.

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

25

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

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

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

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

Lagi, "%" mengacu pada hasil sebelumnya. Untuk mempermudah, kami menyimpan solusi ke dalam 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. Mulailah 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 seorang ahli dalam Maxima, Anda mungkin ingin menggunakan sintaks 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-variabel tersebut dapat digunakan dalam ekspresi simbolik lainnya. Perhatikan bahwa dalam perintah berikut, sisi kanan dari &= dievaluasi sebelum penugasan ke Fx.

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

5
125 E

$125e^5$

18551.64488782208

```
>fx(5)
```

18551.6448878

Untuk mengevaluasi suatu ekspresi dengan nilai-nilai spesifik dari variabel-variabelnya, 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, hal ini tidak berfungsi, karena Maxima tidak mendukungnya. Sebagai gantinya, gunakan sintaks "with" (bentuk yang lebih baik dari perintah at(...) di Maxima).

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

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

Tugasnya juga dapat bersifat simbolik.

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

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

Perintah solve menyelesaikan ekspresi simbolik untuk suatu variabel dalam Maxima.

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

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

Bandingkan dengan bilangan "solve" perintah di Euler, which needs a start value, and optionally a target value.

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

1.56155281281

Nilai numerik dari solusi simbolik dapat dihitung dengan mengevaluasi hasil simbolik. Euler akan membaca melalui tugas-tugas x=, dan sebagainya. Jika Anda tidak memerlukan hasil numerik untuk komputasi lebih lanjut, Anda juga dapat membiarkan Maxima menemukan nilai numerik.

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

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

[-3.23607, 1.23607]

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

Untuk mendapatkan solusi simbolik tertentu, 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 flag yang menunjukkan perlakuan khusus dalam Maxima. Beberapa flag dapat juga digunakan sebagai perintah, sementara yang lainnya tidak. Flag ditambahkan dengan "|" (bentuk yang lebih baik dari "ev(...,flags)")

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

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

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

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

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

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

Fungsi

Dalam EMT, fungsi adalah program yang didefinisikan dengan perintah "function". Fungsi dapat berupa fungsi satu baris atau fungsi multibaris.

Fungsi satu baris dapat bersifat numerik atau simbolik. Fungsi satu baris numerik didefinisikan dengan ":=".

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

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

```
>f(2)
```

```
4.472135955
```

Fungsi ini juga akan berfungsi untuk vektor, mematuhi bahasa matriks Euler, karena ekspresi yang digunakan dalam fungsi tersebut telah di-vectorize.

```
>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 menggunakan ekspresi, kita hanya perlu memberikan nama fungsinya.

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 dalam inti Euler.

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

```
0.707106781187
```

Lebih baik kita menghapus definisi ulang fungsi sin yang tidak perlu.

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

```
0.707106781187
```

Default Parameters

Fungsi numerik dapat memiliki nilai default untuk parameternya

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

Melewatkan parameter ini menggunakan nilai default.

```
>f(4)
```

16

Mengaturnya akan menimpa nilai default.

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

80

An assigned parameter overwrite it too. This is used by many Euler functions like plot2d, plot3d.

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

16

Jika sebuah 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

Tetapi parameter yang ditugaskan menggantikan nilai global. Jika argumen tidak ada dalam daftar parameter yang telah didefinisikan, argumen tersebut harus dideklarasikan dengan :=!

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

20

Fungsi simbolik didefinisikan dengan &=.

Fungsi-fungsi tersebut didefinisikan dalam Euler dan Maxima, dan berfungsi di kedua dunia. Ekspresi definisi dijalankan melalui Maxima sebelum definisi.

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

$$x^3 - xe^{-x}$$

Fungsi simbolik dapat digunakan dalam ekspresi simbolik.

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

$$xe^{-x} - e^{-x} + 3x^2$$

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

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

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

178.635099908

Mereka dapat digunakan untuk mendefinisikan fungsi simbolik atau ekspresi 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

Hal 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)
```

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

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

$$(x + 2)^n$$

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

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

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

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

625

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

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

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

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

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

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

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

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

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

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

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

```
>$&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 &=, fungsi tersebut bersifat simbolik, dan dapat digunakan dalam ekspresi simbolik lainnya.

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

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

Dengan :=, fungsi tersebut bersifat numerik. Contoh yang baik adalah integral tentu seperti

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

Yang tidak dapat dievaluasi secara simbolik. Jika kita mendefinisikan ulang fungsi dengan kata kunci "map", fungsi tersebut dapat digunakan untuk vektor x. Secara internal, fungsi tersebut dipanggil untuk semua nilai x sekali, 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 "base".

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

```
2
6.7
```

Selain itu, dimungkinkan untuk menggunakan parameter yang ditugaskan.

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

```
2
```

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)
```

$$b^2 - ab + b + a^2$$

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

Fungsi simbolik seperti itu dapat digunakan untuk variabel simbolik.
Tetapi fungsi tersebut juga dapat digunakan untuk vektor numerik.

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

```
17
```

Ada juga fungsi simbolik murni yang tidak dapat digunakan secara numerik.

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

```
diff(expr, y, 2) + diff(expr, x, 2)
```

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

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

$$0$$

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

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

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

Kesimpulan:

- &= mendefinisikan fungsi simbolik,

- := mendefinisikan fungsi numerik,

- &&= mendefinisikan fungsi simbolik murni.

Menyelesaikan persamaan

Ekspresi dapat diselesaikan secara numerik dan simbolik. Untuk menyelesaikan ekspresi sederhana satu variabel, kita dapat menggunakan fungsi solve(). Fungsi ini membutuhkan nilai awal untuk memulai pencarian. Secara internal, solve() menggunakan metode secant.

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

```
1.41421356237
```

Hal 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])
```

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

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

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

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

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

```
0.966715594851
2
```

Menyelesaikan ekspresi simbolik dalam bentuk simbolik menghasilkan daftar solusi. Kami menggunakan solver 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 halnya ekspresi.

```
>longest sol()
```

```
-0.6180339887498949      1.618033988749895
```

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

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

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

$$0$$

Menyelesaikan sistem persamaan secara simbolik dapat dilakukan dengan vektor persamaan dan solver simbolik solve().

```
>$&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. Tetapi seringkali kita ingin menggunakan parameter lokal.

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

with a=3.

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

Salah satu cara untuk meneruskan parameter tambahan ke f() adalah dengan menggunakan daftar yang berisi nama fungsi dan parameter-parameternya (cara lain adalah menggunakan parameter titik koma).

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

```
2.54116291558
```

Hal ini juga berlaku untuk ekspresi. Tetapi kemudian, elemen daftar bernama harus digunakan. (Lebih lanjut tentang daftar dalam tutorial tentang sintaksis 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])
```

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

```
>$&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])
```

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

```
>&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])
```

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

Bahasa Matriks

Dokumentasi inti EMT berisi diskusi terperinci tentang bahasa matriks Euler. Vektor dan matriks dimasukkan dengan kurung siku, elemen dipisahkan dengan koma,

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

1	2
3	4

Perkalian matriks dilambangkan dengan titik.

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

3
4

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

[3, 4]

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

-2	1
1.5	-0.5

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

```
11
25
```

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

```
1    0
0    1
```

Titik utama dari bahasa matriks adalah bahwa semua fungsi dan operator bekerja elemen per 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 perkalian matriks, melainkan perkalian elemen per elemen. Hal yang sama berlaku untuk vektor.

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

9
16

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

```
>2*A
```

2	4
6	8

Misalnya, jika operan tersebut adalah vektor kolom, elemen-elemennya akan diterapkan pada semua baris dari A.

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

1	4

3	8

Jika itu adalah vektor baris, maka vektor tersebut akan diterapkan pada semua kolom dari A.

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

2	6
6	12

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

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

1	2
1	2

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

1	4
3	8

Hal ini juga berlaku untuk dua vektor di mana salah satunya adalah vektor baris dan yang lainnya adalah vektor kolom. Kita menghitung i * j untuk i, j dari 1 hingga 5. Triknya adalah 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

Lagi, ingat ini bukan benuk dari 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 kesamaan. Kita mendapatkan vektor 0 dan 1,

```
>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-elemen yang tidak nol. Dalam kasus ini, kita mendapatkan indeks-indeks dari semua elemen yang lebih besar dari 50.

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

    [8, 9, 10]

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

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

    [64, 81, 100]

Sebagai contoh, mari kita mencari sumua akar 2 untuk nomor 1- n100 , yang kongruen dengan 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 bilangan bulat. EMT menggunakan floating-point presisi ganda secara internal.

Namun, EMT sering kali sangat berguna. Kita dapat memeriksa keaslian suatu bilangan prima. 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

Fungsi ini mengembalikan indeks-indeks elemen yang bukan nol.

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

1	4
2	3
2	5
3	9

Indeks-indeks ini dapat digunakan untuk menetapkan elemen-elemen ke suatu nilai.

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

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

. Fungsi mset() juga dapat menetapkan elemen-elemen pada indeks-indeks tersebut ke entri-entri matriks lain.

```
>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 ini memungkinkan untuk mendapatkan elemen di vektor.

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

[0.267829, 0.13673, 0.390567, 0.006085]

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

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

[0.765761, 0.952814, 0.548138]

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

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

[0.765761, 0.952814, 0.548138]

Namun, dengan mget(), kita dapat mengekstrak indeks-indeks tersebut dan menggunakan informasi ini untuk mengekstrak elemen-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 Yang Lain (Building Matriks)

To build a matrix, we can stack one matrix on top of another. If both do not have the same number of columns, the shorter one will be filled with 0. Untuk membangun matriks, kita dapat menumpuk satu matriks di atas yang lain. Jika keduanya tidak memiliki jumlah kolom yang sama, yang lebih pendek akan diisi dengan 0.

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

1	2	3
1	2	3

Demikian pula, kita dapat menggabungkan matriks satu di samping yang lain, 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 matriks-matriks tersebut tidak memiliki jumlah baris yang sama, matriks yang lebih pendek akan diisi dengan 0. Ada pengecualian untuk aturan ini. Angka riil yang dilampirkan pada matriks akan digunakan sebagai kolom yang diisi dengan angka 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

Ya, 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 menginterpretasikan vekto

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

1	1
2	4
3	9

Untuk mendapatkan ukuran matriks A, kita dapat menggunakan fungsi-fungsi berikut.

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

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

Untuk vetor terdapat length().

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

```
9
```

Ada banyak fungsi lain yang menghasilkan matriks.

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

1	1
1	1

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

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

[6, 6, 6, 6, 6]

Selain itu, matriks angka acak dapat dihasilkan dengan fungsi random (distribusi seragam) atau normal (distribusi Gauss).

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

0.66566	0.831835
0.977	0.544258

Berikut adalah fungsi berguna lainnya, yang mengubah struktur elemen-elemen dari 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 fungsi ini dan fungsi dup untuk menulis fungsi rep(), yang mengulang vektor n kali.

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

Mari kita uji.

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

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

Fungsi multdup() menduplikasi elemen-elemen dari sebuah vektor.

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

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

Fungsi flipx() dan flipy() membalikkan urutan baris atau kolom dari suatu matriks.

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

[5, 4, 3, 2, 1]

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

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

[2, 3, 4, 5, 1]

Fungsi khusus drop(v, i) menghapus elemen dengan indeks yang ada dalam 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) mengacu pada indeks elemen dalam v, bukan nilai elemen tersebut.

Jika Anda ingin menghapus elemen, Anda perlu menemukan elemen tersebut terlebih dahulu. Fungsi indexof(v, x) dapat digunakan untuk menemukan elemen x dalam vektor terurut v.

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

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

Seperti yang Anda lihat, tidak ada masalah jika Anda memasukkan indeks di luar jangkauan (seperti 0), indeks ganda, atau indeks yang tidak terurut.

```
>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 menghasilkan matriks diagonal.
Kita mulai dengan matriks identitas.

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

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

0	0	0	1	0
0	0	0	0	1

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

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

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

Perhatikan bahwa kita tidak mengubah matriks A. Kita mendapatkan matriks baru sebagai hasil dari setdiag(). Berikut adalah fungsi yang mengembalikan matriks tri-diagonal.

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

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

0 0 0 1 2

Diagonal suatu matriks juga dapat diekstrak dari matriks tersebut.
Untuk mendemonstrasikan hal ini, kita akan mengubah struktur vektor 1:9 menjadi matriks 3x3.

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

1	2	3
4	5	6
7	8	9

Sekarang kita bisa mengektrak diagaonalnhya

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

[1, 5, 9]

Sebagai contoh, kita dapat membagi matriks dengan diagonalnya. Bahasa matriks memastikan bahwa 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

Vektorisasi

Hampir semua fungsi dalam Euler juga berfungsi untuk input matriks dan vektor, kapan pun hal 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 membuat plot fungsi (alternatifnya menggunakan ekspresi).

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

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

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

$$s = t^3 - t$$

```
>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 memperluas operator untuk skalar, vektor, dan matriks dengan cara yang jelas. Misalnya, sebuah vektor kolom dikalikan dengan sebuah vektor baris akan diperluas menjadi matriks, jika sebuah operator diterapkan. Dalam hal berikut, v’ adalah vektor yang ditranspos (sebuah 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 cukup berbeda dari produk matriks. Produk matriks dilambangkan dengan titik "." dalam EMT.

```
>(1:5).(1:5)’
```

```
55
```

Secara default, vektor baris dicetak dalam format yang ringkas.

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

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

Untuk matriks, operator khusus . menunjukkan perkalian matriks, dan A’ menunjukkan transposisi. Sebuah matriks 1x1 dapat digunakan seperti angka real.

```
>v:=[1,2]; v.v’, %^2
```

```
5
25
```

Untuk mentranspose sebuah matriks, kita menggunakan tanda apostrof.

```
>v=1:4; v’
```

```
1
2
3
4
```

Jadi kita dapat menghitung matriks A dikalikan dengan 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 kuadrat dari vektor baris v. Hasilnya adalah vektor 1x1, yang berfungsi seperti angka real.

```
>v.v’
```

```
30
```

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

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

30

Operator dan fungsi mengikuti bahasa matriks Euler. Berikut adalah ringkasan aturannya:

- Sebuah fungsi yang diterapkan pada vektor atau matriks diterapkan pada setiap elemennya.
- Sebuah operator yang beroperasi pada dua matriks dengan ukuran yang sama diterapkan secara berpasangan pada elemen-elemen matriks tersebut.
- Jika kedua matriks memiliki dimensi yang berbeda, keduanya diperluas dengan cara yang masuk akal, sehingga memiliki ukuran yang sama. Misalnya, sebuah nilai skalar dikalikan dengan sebuah vektor mengalikan nilai tersebut dengan setiap elemen vektor. Atau sebuah matriks dikalikan dengan sebuah vektor (dengan *, bukan .) memperluas vektor menjadi ukuran matriks dengan menduplikasinya. Berikut adalah kasus sederhana dengan operator ^

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

[1, 4, 9]

Berikut adalah kasus yang lebih rumit. Sebuah vektor baris dikalikan dengan sebuah vektor kolom akan memperluas keduanya dengan menduplikasi.

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

1	2	3
2	4	6
3	6	9

Perhatikan bahwa produk skalar menggunakan produk matriks, bukan *!

```
>v.v'
```

14

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

sum,prod computes the sum and products of the rows cumsum,cumprod does the same cumulatively computes the extremal values of each row extrema returns a vector with the extremal information diag(A,i) returns the i-th diagonal setdiag(A,i,v) sets the i-th diagonal id(n) the identity matrix det(A) the determinant charpoly(A) the characteristic polynomial eigenvalues(A) the eigenvalues

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

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

Untuk menggabungkan matriks dan vektor, terdapat operator "|" dan "_".>1:4, 1:2:10

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

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

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

[1, 2, 3, 4, 5]

1	2	3
1	1	1

Elemen-elemen sebuah matriks dirujuk 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 tersebut. Untuk matriks, ini mengembalikan baris ke-i lengkap dari matriks tersebut.

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

6
[7, 8, 9]

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

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

[2, 4]

2
5
8

Bentuk singkat untuk : adalah menghilangkan indeks sepenuhnya.

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

2	3
5	6
8	9

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

```
>A{4}
```

4

Sebuah 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 untuk tabel, mari kita reset ke format default, dan menghitung sebuah tabel nilai sinus dan kosinus. Perhatikan bahwa sudut-sudut secara default dalam radian.

```
>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, di mana setiap baris adalah sebuah tabel dari t^i untuk satu i. Artinya, matriks tersebut memiliki elemen-elemen

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

Sebuah fungsi yang tidak berfungsi untuk input vektor harus "div vektorisasi". Hal ini dapat dicapai dengan kata kunci "map" dalam definisi fungsi. Kemudian fungsi tersebut akan dievaluasi untuk setiap elemen dari parameter vektor. Integrasi numerik integrate() hanya berfungsi untuk batas interval skalar. Jadi kita perlu memvektorisasinya.

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

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

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

[0, 2.05045, 13.7251, 113.336, 1241.03]

Sub-Matrices and Matrix-Elemen

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

1	2	3
4	5	6
7	8	9

5

Kita dapat mengakses seluruh baris dari sebuah matriks.

```
>A[2]
```

[4, 5, 6]

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

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

2

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

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

[4, 5, 6]

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

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

1	2	3
4	5	6

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

Sebagai alternatif, 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-elemen A dengan menetapkan sub-matriks A ke suatu nilai. Hal ini memang mengubah matriks A yang tersimpan.

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

4	2	3
4	5	6
7	8	9

Kita juga dapat menetapkan nilai ke suatu baris dari A.

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

-1	-1	-1
4	5	6

7	8	9

Kita bahkan dapat menetapkan nilai ke sub-matriks jika memiliki ukuran yang sesuai.

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

5	6	-1
7	8	6
7	8	9

.Selain itu, beberapa pintasan diperbolehkan.

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

0	0	-1
0	0	6
7	8	9

Peringatan: Indeks yang berada di luar batas akan mengembalikan matriks kosong atau pesan kesalahan, tergantung pada pengaturan sistem. Secara default, pesan kesalahan akan ditampilkan. Namun, ingatlah bahwa indeks negatif dapat digunakan untuk mengakses elemen-elemen matriks dengan menghitung dari akhir.

```
>A[4]
```

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

Pengurutan dan Pengacakan

Fungsi sort() mengurutkan vektor baris.

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

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

Seringkali perlu mengetahui indeks-indeks vektor yang diurutkan dalam vektor asli. Hal ini dapat digunakan untuk mengurutkan ulang vektor lain dengan cara yang sama. Mari kita acak sebuah vektor.

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

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

Indeks-indeks tersebut berisi urutan yang benar dari v.

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

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

Ini juga bekerja untuk vektor string juga.

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

a
d
e
a
aa
e

```
>{
ss,
in
d}
=s
or
t(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 unique mengembalikan daftar unik yang diurutkan dari elemen-elemen sebuah vektor.

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

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

Ini bekeja juga untuk vektor string.

```
>unique(s)
```

a
aa

d
e

Linear Algebra

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

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

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

```
-4
4.5
```

Sebagai contoh lain, kita menghasilkan matriks 200x200 dan jumlah barisnya. Kemudian kita menyelesaikan Ax=b menggunakan matriks invers. Kita mengukur error sebagai deviasi maksimal dari semua elemen dari 1, yang tentu saja merupakan solusi yang benar.

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

```
8.790745908981989e-13
```

Jika sistem tidak memiliki solusi, sebuah fit linier meminimalkan norma dari error Ax-b.

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

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

Determinan matriks ini adalah 0.

```
>det(A)
```

```
0
```

Simbolik Matriks

Maxima memiliki matriks simbolik. Tentu saja, Maxima dapat digunakan untuk masalah aljabar linier sederhana seperti itu. 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-matriks ini dapat digunakan dalam ekspresi simbolik lainnya. >$&det(A-x*ident(2)), $&solve(%,x)

$$(1-x)\,(2-x)-ab$$

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

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

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

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

Untuk mengekstrak eigenvektor spesifik, diperlukan 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 secara numerik di Euler seperti ekspresi simbolik lainnya.

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

1	4
5	2

Dalam ekspresi simbolik, gunakan dengar

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

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

Akses ke baris dari matriks simbolik sama seperti dengan matriks numerik.

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

$$[1,a]$$

Sebuah ekspresi simbolik dapat mengandung sebuah penugasan. Dan hal itu mengubah matriks A.

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

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

Ada fungsi simbolik dalam Maxima untuk membuat vektor dan matriks. Untuk ini, rujuklah 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 yang 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 sebagai numerik dalam ekspresi numerik. Jadi kita dapat menggunakannya di sini.

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

1	0
0	1

Misalnya, nilai eigen dari A dapat dihitung secara numerik.

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

[16.1168, -1.11684, 0]

Atau secara simbolik. Lihat tutorial tentang Maxima untuk detail tentang hal ini.

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

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

Numerik dalam Ekspresi Simbolik

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

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

```
1         3.14159
4               5
```

Masih ada perbedaan antara bentuk numerik dan simbolik. Ketika mentransfer matriks ke bentuk simbolik, pendekatan fraksional untuk bilangan real akan digunakan.

```
>$&A
```

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

Untuk menghindari hal ini, terdapat fungsi "mxmset(variable)".

```
>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))
```

$$1.4142135623730950488016887242097_B \times 10^0$$

$$1.414213562373095$$

Presisi dari bilangan floating-point besar dapat diubah.

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

```
3.14159265358979323846264338327950288419716939937510582097494\
4592307816406286208998628034825342117068b0
```

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)
```

$$-5.424777960769379$$

Demo – Suku Bangsa

Di bawah ini, kami menggunakan Euler Math Toolbox (EMT) untuk menghitung suku bunga. Kami melakukannya secara numerik dan simbolik untuk menunjukkan kepada Anda bagaimana Euler dapat digunakan untuk menyelesaikan 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 sintaks berikut.

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

```
5150
```

tetapi ini lebih mudah jika menggunakan faktor

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

```
1.03
5150
```

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

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

```
6719.58189672
```

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

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

```
6719.58
```

Baik, mari kita cetak itu dibulatkan hingga 2 digit dalam kalimat lengkap.

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

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

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

```
>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 menghasilkan vektor bilangan bulat.

```
>short 0:10
```

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

Kemudian semua operator dan fungsi dalam Euler dapat diterapkan pada 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]
```

Ini adalah vektor faktor q^0 hingga q^10. Vektor ini dikalikan dengan K, dan kita mendapat vektor nilai.

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

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

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

Baik, mari kita bandingkan kedua hasilnya, dengan dan tanpa pembulatan.

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

```
                  1271.61
                1271.6071
```

Baik, sekarang tidak ada rumus sederhana untuk tahun ke-n, dan kita harus mengulang tahun-tahun tersebut. Euler menyediakan banyak solusi untuk ini. Cara termudah adalah fungsi iterate, yang mengulangi 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 kita dengan 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 spesifik elemen dari factor, kita menggunakan

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

```
    5150.00
     5000.00     5150.00     5304.50
```

Mengejutkannya, 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 sedikit.

Solving Equations

Sekarang kita mengambil fungsi yang lebih canggih, yang menambahkan sejumlah uang tertentu setiap tahun.

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

Kita tidak perlu menentukan q atau R untuk definisi fungsi tersebut. Hanya jika kita menjalankan perintah, kita harus mendefinisikan nilai-nilai ini. Kita 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 mengeluarkan 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 mendapatkan 150 bunga di tahun pertama, tetapi mengeluarkan 200, kita kehilangan uang setiap tahun.

Bagaimana kita dapat menentukan berapa tahun uang akan bertahan? Kita harus menulis loop untuk ini. Cara termudah adalah dengan mengiterasi cukup lama.

```
>VKR=iterate("onepay",5000,50)
```

Real 1 x 51 matrix

5000.00 4950.00 4898.50 4845.45 ...

Menggunakan bahasa matriks, kita dapat menentukan nilai negatif pertama 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, kami akan menurunkan rumus yang diperlukan. Kemudian Anda akan melihat bahwa tidak ada rumus mudah untuk suku bunga. Tetapi untuk saat ini, kami bertujuan untuk solusi numerik.

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

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

Iterasinya sama seperti di atas

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

Tetapi kita tidak lagi menggunakan nilai global R dalam ekspresi kita. Fungsi seperti iterate() memiliki trik khusus dalam Euler. Anda dapat meneruskan nilai variabel dalam ekspresi sebagai parameter titik koma. Dalam hal ini P dan R. Selain itu, kami hanya tertarik pada nilai terakhir. Jadi kami mengambil indeks [-1]. Mari kita coba tes.

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

```
-19.83
```

Sekarang kita bisa menyelesaika permasalahan kita.

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

```
3.15
```

Rutin solve menyelesaikan expression=0 untuk variabel x. Jawabannya adalah 3,15% per tahun. Kami mengambil nilai awal 3% untuk algoritma. Fungsi solve() selalu membutuhkan nilai awal.

Kita dapat menggunakan fungsi yang sama untuk menyelesaikan pertanyaan berikut: Berapa banyak yang dapat kita ambil 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 menyelesaikan untuk jumlah tahun, karena fungsi kami menganggap n sebagai nilai integer.

Simbolik Untuk Masalah Suku Bunga.

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

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

$$R + q\,K$$

Sekarang kita dapat mengulang ini.

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

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

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

Kita melihat pola. Setelah n periode kita memiliki

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

Rumus tersebut adalah rumus untuk jumlah geometri, 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 flag "simpsum" untuk mereduksikannya 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 kami sebelumnya. Tetapi lebih efektif.

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

```
-19.82504734650985
-19.82504734652684
```

kita dapat menggunakannya untuk meminta waktu n. Kapan modal kita habis? Tebakan awal kita adalah 30 tahun.

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

```
20.51
```

Jawaban ini mengatakan bahwa akan negatif setelah 21 tahun.

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

Asumsikan kita mendapatkan pinjaman K, dan membayar n angsuran R (mulai setelah tahun pertama) meninggalkan utang sisa Kn (pada saat pembayaran terakhir). Rumus untuk ini jelas

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

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

Biasanya rumus ini diberikan dalam bentuk

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

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

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

Kita dapat menyelesaikan untuk laju R secara simbolik.

```
>$&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 tersebut, fungsi ini mengembalikan kesalahan floating point untuk i=0. Euler tetap 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 angsuran sebesar 500.

Persamaan tersebut juga dapat diselesaikan untuk n. Persamaan tersebut akan terlihat lebih baik jika kita menerapkan beberapa penyederhanaan.

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

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

[]

Latihan Soal

$$z^2 - 81$$

$$m^2 - 4$$

$$7(3x + 6) = 11 - (x + 2)$$

$$9(2x + 8) = 20 - (x + 5)$$

$$6 - x/x^2 - 36$$

Jawaban

$$z^2 - 8$$

```
>factor(z^2-81)
```

```
Variable z not found!
Error in ^
Error in:
factor(z^2-81) ...
          ^
```

```
>z^2-81,factor(z^2-81)
```

```
Variable z not found!
Error in ^
Error in:
z^2-81,factor(z^2-81) ...
   ^
```

```
>z=symbol("z"), factor(z^2-81)
```

```
Function symbol not found.
Try list ... to find functions!
Error in:
z=symbol("z"), factor(z^2-81) ...
             ^
```

```
>z &= z, factor(z^2-81)
```

```
Wrong argument!

Cannot combine a symbolic expression here.
Did you want to create a symbolic expression?
Then start with &.

Error in ^
Error in:
z &= z, factor(z^2-81) ...
                 ^
```

```
>&z, factor(z^2-81)
```

```
Wrong argument!

Cannot combine a symbolic expression here.
Did you want to create a symbolic expression?
Then start with &.

Error in ^
```

```
Error in:
&z, factor(z^2-81) ...
              ^
```

```
>>$&expand(z^2-81)
```

```
Syntax error in expression, or unfinished expression!
Error in:
>$&expand(z^2-81) ...
^
```

```
>&factor(z^2-81)
```

```
(z - 9) (z + 9)
```

$$25ab^4 - 25az^4$$

```
>&factor(25ab^4-25az^4)
```

```
Maxima said:
incorrect syntax: ab is not an infix operator
factor(25ab^
          ^

Error in:
&factor(25ab^4-25az^4) ...
                       ^
```

```
>&factor((25*(a*b)^24)-(25*(a*z)^4))
```

```
        4        5  6          5  6       2     10  12
- 25 a   (z - a   b  ) (z + a   b  ) (z  + a    b   )
```

$$m^2 - 4$$

```
>$factor(m^2-4)
```

$$(m-2)\ (m+2)$$

$$7(3x + 6) = 11 - (x + 2)$$

```
>solve(7*(3*x+6)=11-(x+2))
```

```
Variable x not found!
Error in:
solve(7*(3*x+6)=11-(x+2)) ...
            ^
```

```
>solve(7*(3*x+6)=11-(x+2),x)
```

```
Variable x not found!
Error in:
solve(7*(3*x+6)=11-(x+2),x) ...
            ^
```

```
>&solve(7*(3*x+6)=11-(x+2),x)
```

```
              3
       [x = - -]
              2
```

```
>solve(7*(3*x+6) = 11-(x+2), x)
```

```
Variable x not found!
Error in:
solve(7*(3*x+6) = 11-(x+2), x) ...
            ^
```

$$9(2x+8) = 20 - (x+5)$$

```
>&solve(9*(2*x+8)=20-(x+5))
```

```
[x = - 3]
```

```
>simplify((6-x)/(x^2-36))
```

```
Variable x not found!
Error in:
simplify((6-x)/(x^2-36)) ...
             ^
```

```
>&simplify((6-x)/(x^2-36))
```

```
           6 - x
simplify(-------)
           2
          x  - 36
```

```
>$&6-x/x^2-36
```

$$-\frac{1}{x}-30$$

Gabriela Saskia Damayani (23030130067)