Terjemahan

Tuzia Afiany Resta
23030630030
Matematika E 2023

EMT untuk Perhitungan Aljabar

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

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

Contoh pertama

Menyederhanakan bentuk aljabar:

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

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

Menjabarkan:

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

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

Baris Perintah

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

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

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

```
  16.7551608191
```

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

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

```
  50.2654824574
  100.530964915
```

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

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

```
  0.540302305868
```

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

```
  0.857553215846
```

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

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

```
  1.41666666667
  1.41421568627
  1.41421356237
```

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

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

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

Garis yang dimulai dengan %% akan benar-benar tidak terlihat.

```
  81
```

Euler mendukung loop dalam baris perintah, asalkan sesuai dengan 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 multi-garis. Pastikan baris diakhiri dengan "...".

```
>x := 1.5; //komentar diletakkan di sini sebelum ...
>repeat xnew:=(x+2/x)/2; until xnew~=x; ...
>   x := xnew; ...
>end; ...
>x,
```

Struktur bersyarat juga berfungsi.

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

```
  Thought so!
```

Saat Anda menjalankan perintah, kursor dapat berada di posisi mana pun 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.

Saat Anda menggerakkan kursor di sepanjang garis, pasangan buka dan tutup tanda kurung atau tanda kurung akan disorot. Juga, perhatikan baris statusnya. Setelah kurung buka dari fungsi sqrt (), baris status akan menampilkan teks bantuan untuk fungsi tersebut. Jalankan perintah dengan tombol kembali.

```
>sqrt(sin(10°)/cos(20°))
```

```
  0.429875017772
```

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

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

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

```
  2.5
```

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

Sintaks Dasar

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

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

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

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

```
  30.65625
```

EMT menggunakan sintaks pemrograman untuk ekspresi. Memasuki

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

Anda harus mengatur tanda kurung yang benar dan menggunakan / untuk pecahan. Perhatikan tanda 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 harus memasukkannya dalam bentuk baris.

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

```
  23.2671801626
```

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

Hasil dari perhitungan ini adalah bilangan floating point. Ini secara default dicetak dengan akurasi sekitar 12 digit. Di baris perintah berikut, kita juga belajar bagaimana kita bisa merujuk ke hasil sebelumnya dalam baris yang sama.

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

```
  0.47619047619
  10/21
```

Perintah Euler bisa berupa ekspresi atau perintah primitif. Ekspresi dibuat dari operator dan fungsi. Jika perlu, itu harus berisi tanda kurung untuk memaksa urutan eksekusi yang benar. Jika ragu, menetapkan braket adalah ide yang bagus. Perhatikan bahwa EMT menampilkan tanda kurung buka dan tutup saat mengedit baris perintah.

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

```
  14.4978445072
```

Operator numerik Euler termasuk

```
+ unary atau operator plus
- unary atau operator minus
*, /
. produk matriks
a ^ b pangkat untuk positif a atau bilangan bulat b (a ** b bekerja
```

juga)

```
n! operator faktorial
```

dan masih banyak lagi.

Berikut 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
konj, re, im, arg, konj, nyata, kompleks
beta, betai, gamma, complexgamma, ellrf, ellf, ellrd, elle
bitand, bitor, bitxor, bitnot
```

Beberapa perintah memiliki alias, mis. ln untuk log.

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

```
  2
  0.5
```

```
>sin(30°)
```

```
  0.5
```

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

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

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

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

```
>longest 1/3
```

```
       0.3333333333333333
```

Representasi ganda internal membutuhkan 8 byte.

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

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

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

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

String

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

```
>"Sebuah string bisa berisi apa saja."
```

```
  Sebuah string bisa berisi apa saja.
```

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

```
>"Luas lingkaran berjari-jari " + 2 + " cm adalah " + pi * 4 + " cm ^ 2."
```

```
  Luas lingkaran berjari-jari 2 cm adalah 12.5663706144 cm ^ 2.
```

Fungsi print juga mengubah angka menjadi string. Ini bisa mengambil sejumlah digit dan sejumlah tempat (0 untuk output padat), dan secara optimal satu unit.

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

```
  Golden Ratio : 1.61803
```

Tidak ada string khusus, yang tidak dicetak. Itu dikembalikan oleh beberapa fungsi, ketika hasilnya tidak penting. (Ini dikembalikan secara otomatis, jika fungsi tidak memiliki pernyataan pengembalian.)

```
>none
```

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

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

```
  1234.5
```

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

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

```
  affe
  charlie
  bravo
```

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

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

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

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

String unicode dapat digabungkan seperti string lainnya.

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

```
 = 45°
```

I

Dalam komentar, entitas yang sama seperti, dll. Dapat digunakan. Ini mungkin alternatif cepat untuk Latex. (Keterangan lebih lanjutpada komentar di bawah).

Ada beberapa fungsi untuk membuat atau menganalisis string unicode. Fungsi strtochar () akan mengenali Unicodestring, dan terjemahkan 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 bilangan Unicode. Fungsi kebalikannya adalah chartoutf ().

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

```
Ü is a German letter
```

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

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

```
We have =.
```

Dimungkinkan juga untuk menggunakan entitas numerik.

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

```
Ähnliches
```

Nilai Boolean

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

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

```
0
1
```

”dan” adalah operator ”&&” dan ”atau” adalah operator ”||”, seperti dalam bahasa C. (Kata ”dan” dan ”atau” bisahanya digunakan dalam kondisi untuk ”jika”.)

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

```
1
```

Operator Boolean mematuhi aturan bahasa matriks.

```
>(1:10)>5, nonzeros(%)
```

```
[0,  0,  0,  0,  0,  1,  1,  1,  1,  1]
[6,  7,  8,  9,  10]
```

Anda dapat menggunakan fungsi nonzeros () untuk mengekstrak elemen tertentu dari vektor. Dalam contoh, kami menggunakan fileisprime bersyarat (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 kami melihat default, kami mengatur ulang format.

```
>defformat; pi
```

```
  3.14159265359
```

Secara internal, EMT menggunakan standar IEEE untuk bilangan ganda dengan sekitar 16 digit desimal. Untuk melihat secara lengkapjumlah digit, gunakan perintah ”longestformat”, atau kita menggunakan operator ”terpanjang” untuk menampilkan hasilnyaformat terpanjang.

```
>longest pi
```

```
        3.141592653589793
```

Berikut adalah representasi heksadesimal internal dari bilangan ganda

```
>printhex(pi)
```

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

Format keluaran dapat diubah secara permanen dengan perintah format.

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

```
      0.33333
      3.14159
      0.84147
```

defaultnya adalah format (12).

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

```
  0.333333333333
```

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

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

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

Format default untuk skalar adalah format (12). Tapi ini bisa diubah.

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

```
  3.1416
```

Fungsi "format terpanjang" mengatur format skalar juga.

```
>longestformat; pi
```

```
  3.141592653589793
```

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

```
shortestformat shortformat longformat, longestformat
format(panjang,digit) goodformat(panjang)
fracformat(panjang)
defformat
```

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

Tetapi format keluaran EMT dapat diatur dengan cara yang fleksibel.

```
>longestformat; pi,
```

```
  3.141592653589793
```

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

```
    3.14159
```

defaultnya adalah defformat().

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

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

```
>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 direpresentasikan dengan tepat. Itukesalahan bertambah sedikit, seperti yang Anda lihat dalam perhitungan berikut.

```
>longest 0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1-1
```

```
   -1.110223024625157e-16
```

Tetapi dengan "longformat" default Anda tidak akan melihat ini. Untuk kenyamanan, keluaran angka 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, gunakantanda kurung setelah ekspresi. Jika Anda ingin menggunakan string sebagai ekspresi, gunakan konvensi untuk menamainya"fx" atau "fxy" dll. Ekspresi lebih diutamakan daripada fungsi.Variabel global dapat digunakan dalam evaluasi.

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

```
        12.56637061435917
```

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

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

```
  -0.919535764538
```

Perhatikan bahwa ekspresi akan selalu menggunakan variabel global, meskipun ada variabel dalam fungsi yang samanama. (Jika tidak, evaluasi ekspresi dalam fungsi dapat memberikan hasil yang sangat membingungkan bagi penggunayang disebut fungsi.)

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

```
  36
```

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

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

```
  45
```

Sebagai referensi, kami berkomentar bahwa koleksi panggilan (dibahas di tempat lain) dapat berisi ekspresi. Jadi kita bisa membuatnyacontoh 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 menjadihindari kebingungan antara ekspresi dan fungsi simbolik.

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

```
  12
```

Dengan cara konvensi, ekspresi simbolik atau numerik harus diberi nama fx, fxy dll. Skema penamaan initidak boleh digunakan untuk fungsi.

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

Bentuk ekspresi khusus memungkinkan variabel apa pun sebagai parameter tanpa nama untuk evaluasi ekspresision, 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 EMT yang membutuhkan ekspresi dalam "x".

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

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

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

```
  -0.475
```

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

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

```
  -0.425
```

Ekspresi tidak perlu simbolis. Ini diperlukan, jika ekspresi berisi fungsi, yang hanyadikenal di kernel numerik, bukan di Maxima.

Matematika Simbolik

EMT melakukan matematika simbolis dengan bantuan Maxima. Untuk detailnya, mulailah dengan tutorial berikut, atau telusurireferensi untuk Maxima. Para ahli di Maxima harus memperhatikan bahwa ada perbedaan dalam sintaks antara filesintaks asli Maxima dan sintaks default ekspresi simbolik di EMT.

Matematika simbolik terintegrasi mulus ke dalam Euler dengan &. Ekspresi apa pun yang dimulai dengan & adalah simbolikekspresi. Itu dievaluasi dan dicetak oleh Maxima.

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

```
>$&44!
```

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

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

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

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()
```

Untuk mempelajari lebih lanjut tentang fungsi tertentu, klik dua kali di atasnya. Misalnya, coba klik dua kali pada "& binomial" dibaris perintah sebelumnya. Ini membuka dokumentasi Maxima yang disediakan oleh penulis program itu.

Anda akan belajar bahwa yang berikut ini juga berfungsi

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

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

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

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

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

Ekspresi simbolik dicetak oleh Maxima dalam bentuk 2D. Alasan untuk ini adalah bendera simbolis khusus ditali.

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

Untuk mencetak ekspresi simbolik dengan LaTeX, gunakan $ infront dari & (atau Anda dapat menghapus &) sebelum perintah. jangan menjalankan perintah Maxima dengan $, jika Anda belum menginstal LaTeX

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

Ekspresi simbolik diurai oleh Euler. Jika Anda membutuhkan sintaks kompleks dalam satu ekspresi, Anda dapat menyertakanekspresi dalam "...". Menggunakan lebih dari sekedar ekspresi sederhana dimungkinkan, tetapi sangat tidak disarankan.

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

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

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

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

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

Ekspresi simbolik dapat digunakan dalam ekspresi simbolik lainnya.

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

Masukan langsung dari perintah Maxima juga tersedia. Mulai baris perintah dengan "::". Sintaks Maximadiadaptasi ke sintaks EMT (disebut "mode kompatibilitas").

```
>&factor(20!)
```

2432902008176640000

```
>::: factor(10!)
```

$$8 \quad 4 \quad 2$$
$$2 \quad 3 \quad 5 \quad 7$$

```
>:: factor(20!)
```

$$18 \quad 8 \quad 4 \quad 2$$
$$2 \quad 3 \quad 5 \quad 7 \quad 11 \quad 13 \quad 17 \quad 19$$

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

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

$$g^2$$

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

$$x^3\, E^x$$

Variabel semacam itu dapat digunakan dalam ekspresi simbolik lainnya. Perhatikan, bahwa dalam perintah berikut ini tangan kanansisi & = dievaluasi sebelum penugasan ke Fx.

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

```
                5
125 E
```

```
18551.64488782208
```

```
>fx(5)
```

```
  18551.6448878
```

Untuk evaluasi ekspresi dengan nilai variabel tertentu, Anda dapat menggunakan operator ”dengan”.

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

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

```
       10        5
1000 E   - 125 E
```

```
2.20079141499189e+7
```

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

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

```
>tex(fx)
```

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

Ekspresi simbolik dapat dievaluasi seperti ekspresi numerik.

```
>fx(0.5)
```

```
  0.206090158838
```

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

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

Penugasan juga bisa bersifat simbolis.

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

Pemecahan perintah memecahkan ekspresi simbolik untuk variabel di Maxima. Hasilnya adalah vektor solusi.

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

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

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

```
  1.56155281281
```

Nilai numerik dari solusi simbolik dapat dihitung dengan evaluasi hasil simbolik. Eulerakan membaca tugas x = dll. Jika Anda tidak memerlukan hasil numerik untuk perhitungan lebih lanjut Andadapat juga membiarkan Maxima menemukan nilai numerik.

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

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

Untuk mendapatkan solusi simbolis tertentu, seseorang dapat menggunakan "dengan" dan indeks.

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

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]
```

Ekspresi simbolis dapat memiliki bendera, yang menunjukkan perlakuan khusus dalam Maxima. Beberapa bendera dapat digunakan sebagaiperintah juga, orang lain tidak bisa. Bendera ditambahkan dengan "|" (a nicer form of "ev(...,flags)")

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

Fungsi

Dalam EMT, fungsi adalah program yang ditentukan dengan perintah "fungsi". Ini bisa menjadi fungsi satu baris atau multiline fungsi.

Fungsi satu baris dapat berupa numerik atau simbolik. Fungsi satu baris numerik ditentukan oleh ": =".

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

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

```
>f(2)
```

```
  4.472135955
```

Fungsi ini akan bekerja untuk vektor juga, mengikuti bahasa matriks Euler, karena ekspresi yang digunakan dalamfungsi adalah vektorisasi.

```
>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. Sebagai ganti ekspresi, kita hanya perlu memberikan nama fungsi.

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

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

```
  0.786151377757
```

Secara default, jika Anda perlu menimpa fungsi built-in, Anda harus menambahkan kata kunci "overwrite". Menimpafungsi built-in berbahaya dan dapat menyebabkan masalah pada fungsi lain tergantung pada fungsinya.

Anda masih bisa memanggil fungsi built-in sebagai "_...", jika itu berfungsi di inti Euler.

```
>function overwrite sin (x) := _sin(x°) // redine sinus dalam derajat
>sin(45)
```

```
  0.707106781187
```

Lebih baik kita menghapus definisi ulang sin ini.

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

```
  0.707106781187
```

Parameter Default

Fungsi numerik dapat memiliki parameter default.

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

Menghilangkan parameter ini menggunakan nilai default.

```
>f(4)
```

```
  16
```

Menyetelnya menimpa nilai default.

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

```
  80
```

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

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

```
  16
```

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

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

```
  24
```

Tetapi parameter yang ditetapkan menggantikan nilai global.

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

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

```
  20
```

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

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

Fungsi simbolik dapat digunakan dalam ekspresi simbolik.

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

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

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

```
  178.635099908
```

Mereka dapat digunakan untuk mendefinisikan fungsi atau ekspresi simbolik lainnya.

```
>function G(x) &= factor(integrate(g(x),x)); $&G(c) // integrate: mengintegralkan
>solve(&g(x),0.5)
```

```
  0.703467422498
```

Cara berikut juga berfungsi, karena Euler menggunakan ekspresi simbolik dalam fungsi g, jika tidak menemukan variabel simbolis g, dan jika terdapat fungsi simbolik g.

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

```
  0.703467422498
```

```
>function P(x,n) &= (2*x-1)^n; $&P(x,n)
>function Q(x,n) &= (x+2)^n; $&Q(x,n)
>$&P(x,4), $&expand(%)
>P(3,4)
```

625

```
>$&P(x,4)+ Q(x,3), $&expand(%)
>$&P(x,4)-Q(x,3), $&expand(%), $&factor(%)
>$&P(x,4)*Q(x,3), $&expand(%), $&factor(%)
>$&P(x,4)/Q(x,1), $&expand(%), $&factor(%)
>function f(x) &= x^3-x; $&f(x)
```

With &= the function is symbolic, and can be used in other symbolic expressions.

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

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

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

yang tidak dapat dievaluasi secara simbolis.

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

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

```
  [-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 ditetapkan.

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

```
  2
```

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

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

Fungsi simbolik seperti itu dapat digunakan untuk variabel simbolik.

Tetapi fungsinya juga dapat digunakan untuk vektor numerik.

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

```
  17
```

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

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

$$\text{diff(expr, y, 2) + diff(expr, x, 2)}$$

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

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

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

Untuk meringkas

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

Memecahkan Ekspresi

Ekspresi dapat diselesaikan secara numerik dan simbolik.

Untuk menyelesaikan ekspresi sederhana dari satu variabel, kita dapat menggunakan fungsi Solving (). Diperlukan nilai awal untuk memulai pencarian. Secara internal, Solving () menggunakan metode garis potong.

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

```
  1.41421356237
```

Ini bekerja untuk ekspresi simbolik juga. Ambil fungsi berikut.

```
>$&solve(x^2=2,x)
>$&solve(x^2-2,x)
>$&solve(a*x^2+b*x+c=0,x)
>$&solve([a*x+b*y=c,d*x+e*y=f],[x,y])
>px &= 4*x^8+x^7-x^4-x; $&px
```

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

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

```
  0.966715594851
  2
```

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

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

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

```
>longest sol()
```

```
      -0.6180339887498949        1.618033988749895
```

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

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

Sistem pemecahan persamaan secara simbolis dapat dilakukan dengan vektor persamaan dan penyelesaian pemecah simbolik (). Jawabannya adalah daftar persamaan.

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

Fungsi f () dapat melihat variabel global. Namun seringkali 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 parameternya (cara lainnya adalah parameter titik koma).

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

```
  2.54116291558
```

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

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

```
  2.54116291558
```

Menyelesaikan Pertidaksamaan

Untuk menyelesaikan pertidaksamaan, EMT tidak akan dapat melakukan, melainkan dengan bantuan Maxima, yaitu 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
```

```
>$&fourier_elim([x^2 - 1>0],[x]) // x^2-1 > 0
>$&fourier_elim([x^2 - 1<0],[x]) // x^2-1 < 0
>$&fourier_elim([x^2 - 1 # 0],[x]) // x^-1 <> 0
>$&fourier_elim([x # 6],[x])
>$&fourier_elim([x < 1, x > 1],[x]) // tidak memiliki penyelesaian
>$&fourier_elim([minf < x, x < inf],[x]) // solusinya R
>$&fourier_elim([x^3 - 1 > 0],[x])
>$&fourier_elim([cos(x) < 1/2],[x]) // ??? gagal
>$&fourier_elim([y-x < 5, x - y < 7, 10 < y],[x,y]) // sistem pertidaksamaan
>$&fourier_elim([y-x < 5, x - y < 7, 10 < y],[y,x])
>$&fourier_elim((x + y < 5) and (x - y >8),[x,y])
>$&fourier_elim(((x + y < 5) and x < 1) or  (x - y >8),[x,y])
>&fourier_elim([max(x,y) > 6, x # 8, abs(y-1) > 12],[x,y])
```

```
       [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])
```

Bahasa Matriks

Dokumentasi inti EMT berisi diskusi terperinci tentang bahasa matriks Euler.

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

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

```
                1               2
                3               4
```

Produk matriks dilambangkan dengan titik.

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

```
                3
                4
```

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

```
  [3,  4]
```

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

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

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

```
11
25
```

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

```
1  0
0  1
```

Poin utama dari bahasa matriks adalah bahwa semua fungsi dan operator mengerjakan elemen untuk 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 dengan elemen. Pekerjaan yang sama untuk vektor.

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

```
                9
               16
```

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

```
>2*A
```

```
                2               4
                6               8
```

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

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

```
                1               4
                3               8
```

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

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

```
                2               6
                6              12
```

Dapat dibayangkan perkalian ini seolah-olah vektor baris v telah diduplikasi untuk membentuk matriks dengan ukuran yang sama dengan A.

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

```
1  2
1  2
```

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

```
1  4
3  8
```

Ini juga berlaku untuk dua vektor di mana satu adalah vektor baris dan yang lainnya adalah vektor kolom. Kami menghitung i * j untuk i, j dari 1 sampai 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
```

Sekali lagi, ingatlah bahwa ini bukan hasil perkalian 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.
Kami mendapatkan vektor 0 dan 1, di mana 1 berarti benar.

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

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

Dari vektor seperti itu, "nonzeros" memilih elemen bukan nol.
Dalam hal ini, kami mendapatkan indeks dari semua elemen yang lebih besar dari 50.

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

```
  [8,  9,  10]
```

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

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

```
  [64,  81,  100]
```

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

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

```
[4,  48,  95,  139,  147,  191,  238,  282,  290,  334,  381,  425,
433,  477,  524,  568,  576,  620,  667,  711,  719,  763,  810,  854,
862,  906,  953,  997]
```

EMT tidak sepenuhnya efektif untuk perhitungan integer. Ini menggunakan titik mengambang presisi ganda secara internal. Namun, seringkali ini sangat berguna.

Kita bisa 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
```

Ini mengembalikan indeks elemen, yang bukan nol.

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

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

Indeks ini dapat digunakan untuk mengatur elemen ke 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 menyetel elemen pada indeks ke entri beberapa matriks lainnya.

```
>mset(A,k,-random(size(A)))
```

```
0.765761     0.401188     0.406347    -0.126917
-0.122404    -0.691673    0.495975     0.952814
0.548138     -0.483902    0.444255     0.539246
```

Dan dimungkinkan untuk mendapatkan elemen dalam vektor.

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

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

Fungsi berguna lainnya 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 di setiap baris.

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

```
  [0.765761,  0.952814,  0.548138]
```

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

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

```
  [0.765761,  0.952814,  0.548138]
```

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

```
>j=(1:rows(A))'|ex[,4], mget(-A,j)
```

```
             1             1
             2             4
             3             1
  [-0.765761,  -0.952814,  -0.548138]
```

Fungsi Matriks Lainnya (Building Matrix)

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

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

```
       1       2       3
       1       2       3
```

Demikian juga, kita dapat melampirkan matriks ke sisi lain secara berdampingan, jika keduanya memiliki jumlah baris yang sama.

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

```
    0.032444   0.0534171    0.595713    0.564454           1
     0.83916    0.175552    0.396988     0.83514           2
   0.0257573    0.658585    0.629832    0.770895           3
```

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

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

```
>A|1
```

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

Dimungkinkan untuk membuat matriks vektor baris dan kolom.

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

```
       1       2       3
       1       2       3
```

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

```
       1       1
       2       2
       3       3
```

Tujuan utamanya adalah untuk menafsirkan vektor ekspresi untuk vektor kolom.

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

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

Untuk mendapatkan ukuran A, kita bisa menggunakan fungsi-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]
```

Juga matriks bilangan acak 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 menyusun kembali elemen-elemen matriks menjadi matriks lain.

```
>redim(1:9,3,3) // menyusun elemen2 1, 2, 3, ..., 9 ke bentuk matriks 3x3
```

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

Dengan fungsi berikut, kita dapat menggunakan this dan fungsi dup untuk menulis fungsi rep (), yang mengulang vektor sebanyak 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 vektor.

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

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

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

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

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

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

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

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

Fungsi khusus adalah drop (v, i), yang menghilangkan 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) mengacu pada indeks elemen di v, bukan nilai elemen. Jika Anda ingin menghapus elemen, Anda harus menemukan elemennya terlebih dahulu. Indeks fungsi (v, x) dapat digunakan untuk mencari elemen x dalam vektor yang diurutkan v.

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

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

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

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

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

Ada beberapa fungsi khusus untuk mengatur diagonal atau untuk menghasilkan matriks diagonal. Kami 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 kami mengatur diagonal bawah (-1) menjadi 1: 4.

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

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

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

Berikut adalah fungsi yang mengembalikan matriks tri-diagonal.

```
>function tridiag (n,a,b,c) := setdiag(setdiag(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 mendemonstrasikan ini, kami merestrukturisasi vektor 1: 9 menjadi matriks 3x3.

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

1	2	3
4	5	6
7	8	9

Sekarang kita bisa mengekstrak diagonal.

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

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

Misalnya. Kita dapat membagi matriks dengan diagonalnya. Bahasa matriks memperhatikan bahwa vektor kolom d diterapkan ke matriks baris demi baris.

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

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

Vektorisasi

Hampir semua fungsi di Euler bekerja untuk matriks dan input vektor juga, jika memungkinkan.

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, vektor nilai fungsi dapat dibuat dengan mudah.

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

$$s = t^3 - t$$

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

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

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

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

```
>shortest (1:5)*(1:5)'
```

```
 1   2   3   4   5
 2   4   6   8  10
 3   6   9  12  15
 4   8  12  16  20
 5  10  15  20  25
```

Perhatikan, ini sangat berbeda dari hasil perkalian matriks. Produk matriks dilambangkan dengan titik "." di EMT.

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

```
  55
```

Secara default, vektor baris dicetak dalam format kompak.

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

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

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

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

```
  5
  25
```

Untuk mentranpose matriks, kami menggunakan apostrof.

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

```
1
2
3
4
```

Sehingga 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 real.

```
>v.v'
```

```
  30
```

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

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

```
  30
```

Operator dan fungsi mematuhi bahasa matriks Euler.

Berikut adalah ringkasan aturannya.

- Fungsi yang diterapkan ke vektor atau matriks diterapkan ke setiap elemen.

- Seorang operator yang beroperasi pada dua matriks dengan ukuran yang sama diterapkan berpasangan ke elemen matriks.

- Jika kedua matriks memiliki dimensi yang berbeda, keduanya diperluas dengan cara yang masuk akal, sehingga memiliki ukuran yang sama.

Mis., Nilai skalar dikalikan vektor mengalikan nilai dengan setiap elemen vektor. Atau matriks dikalikan dengan vektor (dengan *, bukan.) Memperluas vektor ke ukuran matriks dengan menduplikasinya.

Berikut ini adalah kasus sederhana dengan operator ^.

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

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

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

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

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

Perhatikan bahwa produk skalar menggunakan produk matriks, bukan *!

```
>v.v'
```

```
  14
```

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

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

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

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

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

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

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

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

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

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

Unsur-unsur matriks disebut dengan "A [i, j]".

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

```
  6
```

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

```
>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{4}
```

```
  4
```

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

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

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

Untuk menggunakan matriks untuk tabel, mari kita reset ke format default, dan menghitung tabel nilai sinus dan cosinus. 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 kami 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, kami menghitung t [j] ^ i untuk i dari 1 ke n. Kami mendapatkan matriks, di mana setiap baris adalah tabel t ^ i untuk satu i. Yaitu, matriks memiliki elemen

$$a_{i,j} = t_j^i, \ quad1 \ lej \ le101, \ quad1 \ lei \ len$$

Fungsi yang tidak bekerja untuk input vektor harus di-vectorisasi. Hal ini dapat dicapai dengan kata kunci "peta" dalam definisi fungsi. Kemudian fungsi tersebut akan dievaluasi untuk setiap elemen dari parameter vektor.

Integrasi numerik integ () hanya berfungsi untuk batas interval skalar. Jadi kita perlu melakukan vektorisasi.

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

Kata kunci ”map” membuat fungsi menjadi vektor. Fungsi tersebut sekarang akan bekerja 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 braket.

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

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

Kita dapat mengakses baris matriks yang lengkap.

```
>A[2]
```

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

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

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

```
  2
```

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

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

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

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

Di sini kita ingin baris pertama dan kedua dari A.

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

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

Kami bahkan dapat menyusun ulang A menggunakan vektor indeks. Tepatnya, kami tidak mengubah A di sini, tetapi menghitung versi A.

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

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

Trik indeks juga bekerja dengan kolom.

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

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

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

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

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

```
     3
     6
     9
```

Cara lainnya, biarkan indeks pertama kosong.

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

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

Kita juga bisa mendapatkan baris terakhir A.

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

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

Sekarang mari kita ubah elemen A dengan menetapkan submatrix dari A ke beberapa nilai. Ini sebenarnya mengubah matriks A yang disimpan.

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

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

Kami juga dapat menetapkan nilai ke baris A.

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

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

Kami bahkan dapat menetapkan 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 di luar batas menampilkan matriks kosong, atau pesan kesalahan, bergantung pada pengaturan sistem. Standarnya adalah pesan kesalahan. Ingat, bagaimanapun, bahwa indeks negatif dapat digunakan untuk mengakses elemen matriks yang dihitung dari akhir.

```
>A[4]
```

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

Sorting and Shuffling

Fungsi sort () mengurutkan vektor baris.

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

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

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

Mari kita mengacak vektor.

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

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

Indeks tersebut berisi urutan v.

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

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

Ini bekerja untuk vektor string juga.

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

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

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

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

Seperti yang Anda lihat, posisi entri ganda agak acak.

```
>ind
```

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

Fungsi unik mengembalikan daftar elemen unik vektor yang diurutkan.

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

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

Ini bekerja untuk vektor string juga.

```
>unique(s)
```

```
  a
  aa
  d
  e
```

Aljabar linier

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

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

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

```
8.790745908981989e-13
```

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

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

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

Determinan dari matriks ini adalah 0.

```
>det(A)
```

```
0
```

Matriks Simbolik

Maxima memiliki matriks simbolis. Tentu saja, Maxima dapat digunakan untuk soal-soal aljabar linier sederhana. Kita dapat mendefinisikan matriks untuk Euler dan Maxima dengan &: =, dan kemudian menggunakannya dalam ekspresi simbolik. Bentuk [...] biasa untuk mendefinisikan matriks dapat digunakan di Euler untuk mendefinisikan matriks simbolik.

```
>A &= [a,1,1;1,a,1;1,1,a]; $A
>$&det(A), $&factor(%)
>$&invert(A) with a=0
>A &= [1,a;b,2]; $A
```

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

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

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

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

Untuk mengekstrak vektor eigen tertentu, perlu pengindeksan yang cermat.

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

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

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

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

```
                1                4
                5                2
```

Dalam ekspresi simbolik, gunakan dengan.

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

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

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

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

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

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

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

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

Hasilnya dapat dievaluasi secara numerik 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 kuat, yang membuat upaya lebih besar dan mendapatkan hasil yang lebih tepat.

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

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

```
                          1                          0
                          0                          1
```

Misalnya. nilai eigen dari A dapat dihitung secara numerik.

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

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

Atau secara simbolis. Lihat tutorial tentang Maxima untuk detailnya.

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

Nilai Numerik dalam Ekspresi simbolik

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

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

```
                1       3.14159
                4             5
```

Masih terdapat perbedaan antara bentuk numerik dan simbolik. Saat mentransfer matriks ke bentuk simbolis, pendekatan pecahan untuk real akan digunakan.

```
>$&A
```

Untuk menghindarinya, ada fungsi "mxmset (variabel)".

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

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

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

Ketepatan angka floating point besar dapat diubah.

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

```
        3.14159265358979323846264338327950288419716939937510582097494\
 459230781640628620899862803482534211706 8b0
```

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

Perhatikan bahwa ini hanya diperlukan, jika variabel telah ditentukan dengan ": =" atau "=" sebagai variabel numerik.

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

Demo - Suku Bunga

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

```
>K=5000
```

```
  5000
```

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

```
>K*1.03
```

```
  5150
```

Euler akan memahami sintaks berikut juga.

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

```
  5150
```

Tapi lebih mudah menggunakan faktornya

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

```
  1.03
  5150
```

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

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

```
  6719.58189672
```

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

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

```
      6719.58
```

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

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

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

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

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

```
  Real 1 x 11 matrix
       5000.00      5150.00      5304.50      5463.64      ...
```

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

```
>short 0:10
```

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

Kemudian semua operator dan fungsi di Euler dapat diterapkan ke elemen vektor untuk elemen. Begitu

```
>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 mendapatkan nilai vektor.

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

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

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

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

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

```
                  1271.61
                1271.6071
```

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

Cara termudah adalah fungsi iterasi, yang mengulang fungsi tertentu beberapa kali.

```
>VKr=iterate("oneyear",5000,10)
```

```
  Real 1 x 11 matrix

      5000.00      5150.00      5304.50      5463.64      ...
```

Kami dapat mencetaknya dengan cara yang ramah, menggunakan format kami dengan tempat desimal tetap.

```
>VKr'
```

```
      5000.00
      5150.00
      5304.50
      5463.64
      5627.55
      5796.38
      5970.27
      6149.38
      6333.86
      6523.88
      6719.60
```

Untuk mendapatkan elemen tertentu dari vektor, kami menggunakan indeks dalam tanda kurung siku.

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

```
      5150.00
      5000.00     5150.00     5304.50
```

Anehnya, kita juga bisa menggunakan indeks vektor. Ingat bahwa 1: 3 menghasilkan vektor [1,2,3]. Mari kita bandingkan elemen terakhir dari nilai yang dibulatkan dengan nilai penuh.

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

```
      6719.60
      6719.58
```

Perbedaannya sangat kecil.

Memecahkan Persamaan

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

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

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

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

```
  Real 1 x 11 matrix
       5000.00      5350.00      5710.50      6081.82      ...
```

Bagaimana jika kita menghapus jumlah yang sama setiap tahun?

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

```
  Real 1 x 11 matrix
       5000.00      4950.00      4898.50      4845.45      ...
```

Kami melihat bahwa uang berkurang. Jelas, jika kita hanya mendapatkan 150 bunga di tahun pertama, tetapi menghapus 200, kita kehilangan uang setiap tahun.

Bagaimana kita bisa menentukan berapa tahun uang itu akan bertahan? Kami harus menulis loop untuk ini. Cara termudah adalah melakukan iterasi cukup lama.

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

```
  Real 1 x 51 matrix

      5000.00      4950.00      4898.50      4845.45      ...
```

Dengan menggunakan bahasa matriks, kita dapat menentukan nilai negatif pertama dengan cara berikut.

```
>min(nonzeros(VKR<0))
```

```
        48.00
```

Alasan untuk ini adalah bahwa 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. Ini bisa mengambil kondisi akhir sebagai argumen. Maka itu 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 0 setelah 50 tahun. Berapa tingkat bunganya?

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

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

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

Iterasinya sama seperti di atas

$$x_{n+1} = x_n \; cdot \; kiri(1 + \;\; fracP100 \; kanan) + R$$

Tapi kami lebih lama menggunakan nilai global R dalam ekspresi kami. Fungsi seperti iterate () memiliki trik khusus di Euler. Anda dapat meneruskan nilai variabel dalam ekspresi sebagai parameter titik koma. Dalam hal ini P dan R.

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

Mari kita coba tes.

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

```
       -19.83
```

Sekarang kita bisa menyelesaikan masalah kita.

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

```
         3.15
```

Rutin menyelesaikan memecahkan ekspresi = 0 untuk variabel x. Jawabannya adalah 3,15% per tahun. Kami mengambil nilai awal 3% untuk algoritme. Fungsi Solving () selalu membutuhkan nilai awal.

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

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

```
-336.08
```

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

Solusi Simbolis untuk Masalah Suku Bunga

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

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

Sekarang kita dapat mengulang ini.

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

Kami melihat sebuah pola. Setelah n periode yang kita miliki

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

Rumusnya adalah rumus jumlah geometris, yang dikenal dengan Maxima.

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

Ini agak rumit. Jumlahnya dievaluasi dengan bendera ”simpsum” untuk menguranginya 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)
```

Fungsinya sama dengan fungsi f kita sebelumnya. Tapi itu lebih efektif.

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

```
-19.82504734650985
-19.82504734652684
```

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

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

```
        20.51
```

Jawaban ini mengatakan bahwa itu akan menjadi negatif setelah 21 tahun.

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

Asumsikan kita mendapat pinjaman sebesar K, dan membayar n pembayaran R (dimulai setelah tahun pertama) meninggalkan sisa utang Kn (pada saat pembayaran terakhir). Rumusnya jelas

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

Biasanya rumus ini diberikan dalam bentuk

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

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

Kita bisa mencari nilai R secara simbolis.

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

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

Tentu saja, kami memiliki batasan berikut.

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

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

Persamaan ini juga bisa diselesaikan untuk n. Ini terlihat lebih bagus, jika kita menerapkan beberapa penyederhanaan padanya.

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

Aljabar Linear Elementer

Matriks

Membuat matriks dari persamaan linear & Menuliskannya di kode EMT

$$4v - 3w - 5x - 3y + z = 7$$

$$2v - w + 3x + 6y + 2z = -4$$

$$v + w + x + y + z = 1$$

$$-3v + 9w - 8x + 3y - z = 10$$

$$v + 5w - 3x + 6y - 3z = 9$$

```
>spla &= 4*v-3*w-5*x-3*y+1*z= 7; splb &= 2*v-1*w+3*x+6*y+2*z= -4; splc &= v+w+x+y+z= 1; ...
>spld &= -3*v+9*w-8*x+3*y-z= 10; sple &= v+5*w-3*x+6*y-3*z=9; $&spla, $&splb, $&splc, $&spld, $&sple
```

Mencari nilai-nilai variabel dari persamaan tersebut

```
>sspl &= solve([spla,splb,splc,spld,sple],[v,w,x,y,z]); $&sspl
```

Membuat Augmented Matriks dari persamaan diatas

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

4	-3	-5	-3	1
2	-1	3	6	2
1	1	1	1	1
-3	9	-8	3	-1
1	5	-3	6	-3

```
>B := [7;-4;1;10;9]
```

7
-4
1
10
9

Mencari nilai-nilai lain yang bisa di dapatkan dari persamaan tersebut

```
>det(A) // mencari nilai determinan
```

```
  3717
```

```
>&echelon (A) // mencari nilai eselon baris
```

$$\begin{bmatrix} 1 & \frac{3}{4} & \frac{5}{4} & \frac{3}{4} & \frac{1}{4} \\ 0 & 1 & 11 & 15 & 3 \\ 0 & 0 & 1 & \frac{49}{34} & \frac{9}{34} \\ 0 & 0 & 0 & 1 & \frac{77}{797} \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

```
>&rank (A) // mencari rank dari suatu matriks
```

```
>kernel (A)
```

```
0
0
0
0
0
```

```
>A.kernel (A)
```

```
0
0
0
0
0
```

```
>&invert (A)// mencari invers dari matriks tersebut
```

$$\begin{bmatrix} \frac{355}{3717} & -\frac{376}{3717} & \frac{71}{177} & -\frac{358}{3717} & \frac{484}{3717} \end{bmatrix}$$

```
[                                          ]
[   146      505     77       53      136   ]
[ - ----   - ----   ---     ----    ----   ]
[   3717     3717    177     3717    3717   ]
[                                          ]
[   326      262     41       289     100   ]
[ - ----   - ----   ---   - ----    ----   ]
[   3717     3717    177     3717    3717   ]
[                                          ]
[    4       88       41     16       11    ]
[ - ---     ---     - ---    ---     ---    ]
[   531      531      177    531     531    ]
[                                          ]
[  145      527      29      482      797   ]
[  ----     ----     ---     ----   - ----  ]
[  3717     3717     177     3717     3717  ]
```

```
>A*B // mengalikan kedua matriks yang ada
```

```
      28      -21      -35      -21        7
      -8        4      -12      -24       -8
       1        1        1        1        1
     -30       90      -80       30      -10
       9       45      -27       54      -27
```

```
>A.A // cross product terhadap matriks itu sendiri
```

```
      15      -36      -13      -38       -7
```

-7	62	-64	21	-9
5	11	-12	13	0
-12	14	13	58	7
-10	28	-32	24	11

```
>transpose (A)// mengtranspose matriks tersebut
```

4	2	1	-3	1
-3	-1	1	9	5
-5	3	1	-8	-3
-3	6	1	3	6
1	2	1	-1	-3

Membuat Matriks Identitas

```
>C=id(4)
```

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

Vektor

Diberikan sebuah vektor dibawah ini, lalu dicari berbagai nilainya

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

```
  1
  1
  1
```

```
>w:=[1;2;-1]
```

```
  1
  2
 -1
```

```
>v:=[1;1;1]; w:=[1;2;-1]; v1:=w-(w'.v)/(v'.v)*v; fracprint(v1); // mencari ortogonalitas vektor ters
```

```
  1/3
  4/3
 -5/3
```

```
>v2:=crossproduct(v,w), // mencari nilai crossproduct
```

```
-3
 2
 1
```

```
>M:=[v/norm(v),v1/norm(v1),v2/norm(v2)] // mencari normalitas dari vektor tersebut, seharusnya membe
```

```
       0.57735      0.154303     -0.801784
       0.57735      0.617213      0.534522
       0.57735     -0.771517      0.267261
```

```
>sort([2,-1,1])// mengurutkan nilai vektor
```

```
  [-1,  1,  2]
```

Mencoba Latihan Soal

Faktorkan

$$z^2 - 81$$

```
>&factor (z^2 - 81)// mengfaktorkan persamaan kuadrat
```

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

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

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

```
                                    2      2
- 25 (az - ab) (az + ab) (az  + ab )
```

$$t^2 - \frac{27}{100} + \frac{3}{5}t$$

```
>&factor (t^2-27/100+3/5*t)
```

$$\frac{(10\ t - 3)\ (10\ t + 9)}{100}$$

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

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

Selesaikan Pertidaksamaan

$$\left|\frac{2x+1}{3}\right| > 5$$

```
>$&fourier_elim([abs(2*x+1)/3> 5],[x])
```

Selesaikan Persamaan tersebut menjadi suatu matriks yang diketahui nilai-nilainya.

$$4u + 4v - 5x - 3y + z = 7$$

$$3u + 2v - w + 3x + 6y = -4$$

$$6u + w + x + y + z = 1$$

$$-3v + 9w - 8x + 3y - z = 10$$

$$2u + v + 5w - 3x + 6y - 3z = 9$$

$$u + 4v + 3w - 2x + y + 5z = 4$$

```
>A := [4,4,0,-5,-3,1;3,2-1,3,0,6;6,0,1,1,1,1;0,-3,9,-8,3,1;2,1,5,-3,6,-3;1,4,3,-2,1]// Membuat Augme
```

```
  Real 6 x 6 matrix
             4             4             0            -5     ...
             3             1             3             0     ...
             6             0             1             1     ...
             0            -3             9            -8     ...
             2             1             5            -3     ...
             1             4             3            -2     ...
```

```
>B := [7;-4;1;10;9;4]// Augmented Matriks dari hasil persamaan tersebut
```

```
             7
```

```
-4
 1
10
 9
 4
```

```
>spla &= 4*u+4*v+0-5*x-3*y+1*z= 7; splb &= 3*u+2*v-1*w+3*x+6*y= -4; splc &= 6*u+w+x+y+z= 1; ...
>spld &= -3*v+9*w-8*x+3*y-z= 10; sple &= 2*u+v+5*w-3*x+6*y-3*z=9; splf &= u+4*v+3*w-2*x+y+5*z=4; $&s
```

Cari nilai-nilainya

```
>sspl &= solve([spla,splb,splc,spld,sple,splf],[u,v,w,x,y,z]); $&sspl
```