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 diabaikan/dihilangkan.

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

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

```
          16.76
```

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

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

```
          50.27
         100.53
```

Baris perintah dieksekusi sesuai urutan pengguna saat menekan tombol Enter. Jadi, anda akan mendapatkan nilai baru setiap kali anda mengeksekusi baris kedua.

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

```
            0.54
```

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

```
            0.86
```

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

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

```
            1.42
            1.41
            1.41
```

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

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

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

Sebuah baris yang dimulai dengan %% akan sepenuhnya tidak terlihat.

```
           81.00
```

Euler mendukung penggunaan loop dalam baris perintah, asalkan loop tersebut dapat dimasukkan ke dalam satu baris tunggal atau multi-baris. Dalam program, batasan ini tidak berlaku, tentu saja. Untuk informasi lebih lanjut, silakan merujuk ke pengenalan berikut.

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

```
            1.50
            1.42
            1.41
            1.41
            1.41
```

Tidak masalah untuk menggunakan multi-baris. Pastikan baris tersebut diakhiri dengan "...".

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

```
            1.41
```

Struktur kondisional juga berfungsi.

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

```
    Thought so!
```

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

Saat Anda memindahkan kursor sepanjang baris, pasangan kurung buka dan tutup atau kurung siku akan ditandai. Perhatikan juga baris status. Setelah kurung buka fungsi sqrt(), baris status akan menampilkan teks bantuan untuk fungsi tersebut. Jalankan perintah dengan tombol Enter.

```
>sqrt(sin(10°)/cos(20°))
```

```
           0.43
```

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

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

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

```
           2.50
```

Anda juga dapat menyalin dan menempelkan teks di Euler. Gunakan Ctrl-C dan Ctrl-V untuk melakukan hal ini. Untuk menandai teks, seret mouse atau gunakan tombol Shift bersama dengan tombol panah mana pun. Selain itu, Anda dapat menyalin kurung yang telah ditandai.

Sintaks Dasar

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

Untuk menetapkan variabel, gunakan “=” atau “:=”. Untuk kejelasan, pengenalan ini menggunakan bentuk terakhir. Spasi tidak berpengaruh. Namun, spasi di antara perintah diharapkan.

Perintah ganda dalam satu baris dipisahkan dengan “,” atau “;”. Tanda titik koma (;) menonaktifkan output perintah. Di akhir baris perintah, tanda koma (,) dianggap ada jika tanda titik koma (;) tidak ada.

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

```
          30.66
```

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 menggunakan / untuk pecahan. Perhatikan kurung yang ditandai untuk bantuan. Perhatikan bahwa konstanta Euler e disebut E dalam EMT.

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

```
            8.78
```

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
```

```
           23.27
```

Letakkan kurung secara hati-hati di sekitar sub-ekspresi yang perlu dihitung terlebih dahulu. EMT membantu Anda dengan menyoroti ekspresi yang ditutup oleh kurung penutup. Anda juga harus memasukkan nama “pi” untuk huruf Yunani pi.

Hasil perhitungan ini adalah bilangan floating point. Secara default, hasil ini dicetak dengan akurasi sekitar 12 digit. Dalam perintah baris berikut, kita juga belajar cara merujuk ke hasil sebelumnya dalam baris yang sama.

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

```
            0.48
    10/21
```

Sebuah perintah Euler dapat berupa ekspresi atau perintah primitif. Ekspresi terdiri dari operator dan fungsi. Jika diperlukan, ekspresi harus mengandung kurung untuk memastikan urutan eksekusi yang benar. Jika ragu, menggunakan kurung adalah ide yang baik. Perhatikan bahwa EMT menampilkan kurung pembuka dan penutup saat mengedit baris perintah.

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

```
           14.50
```

Operator numerik Euler meliputi

+ operator unary atau tambah
- operator unary atau kurang
*, /
. perkalian matriks
a^b pangkat untuk a positif atau b bilangan bulat (a**b juga berfungsi)
n! operator faktorial

dan banyak lagi.

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

```
sin,cos,tan,atan,asin,acos,rad,deg
log,exp,log10,sqrt,logbase
bin,logbin,logfac,mod,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))
```

```
            2.00
            0.50
```

```
>sin(30°)
```

```
            0.50
```

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)
```

```
     2417851639229258349412352.00
        4096.00
     2417851639229258349412352.00
```

Bilangan Riil

Perkembangan matematika mengungkapkan keterbatasan sistem bilangan yang ada. Bilangan rasional, yang dapat dinyatakan sebagai pecahan, ternyata tidak cukup untuk mengukur semua besaran dalam geometri. Sebagai contoh, panjang diagonal sebuah persegi dengan sisi 1 adalah akar kuadrat dari 2. Bilangan ini terbukti tidak dapat dinyatakan sebagai pecahan, sehingga memperkenalkan konsep bilangan irasional. Gabungan antara himpunan bilangan rasional dan himpunan bilangan irasional inilah yang membentuk himpunan bilangan riil. Sistem ini memberikan landasan yang utuh bagi analisis matematika dan pengukuran kontinu.

Bilangan riil dapat direpresentasikan sebagai titik-titik pada sebuah garis lurus yang memanjang tak terhingga ke kedua arah. Setiap titik pada garis ini berkorespondensi dengan satu dan hanya satu bilangan riil, dan sebaliknya. Representasi ini memvisualisasikan sifat urutan (titik di sebelah kanan merepresentasikan bilangan yang lebih besar) dan sifat kelengkapan (tidak ada celah atau lompatan antara titik-titik tersebut).

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

```
>longest 1/3
```

```
          0.3333333333333333
```

Representasi ganda internal membutuhkan 8 byte.

Representasi ganda adalah format penyimpanan untuk floating-point yang menggunakan 64 bit(8 byte)

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

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

Perintah "printdual" akan mencetak representasi internal dari sebuah bilangan floating-point dalam format presisi ganda (pendekatan yang sangat dekat dengan nilai aslinya tetapi tidak persis sama) meskipun ia tergantung pada konteks bahasa pemrograman tertentu.

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

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

Perintah "printhex" akan mencetak representasi dari nilai floating-point dalam bentuk heksadesimal (basis 16). Heksadesimal merupakan cara yang lebih ringkas untuk menampilkan nilai biner karena setiap digit heksadesimal mempresentasikan empat digit

biner.

String

String merupakan tipe data abstrak yang merepresentasikan sekuens terurut dari elemen-elemen karakter. Setiap karakter dalam string menempati posisi tertentu yang dapat diidentifikasi melalui indeks numerik. String didefinisikan sebagai himpunan terbatas karakter dengan panjang tertentu, dimana panjang string merupakan jumlah total karakter dalam sekuens tersebut.

Sebuah string dalam Euler didefinisikan dengan "...".

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

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

String dapat digabungkan menggunakan | atau +. Hal ini juga berlaku untuk angka, yang akan dikonversi menjadi string dalam hal ini.

```
>"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 dapat mengubah angka menjadi string. Fungsi ini dapat menerima jumlah digit dan jumlah posisi (0 untuk output padat), serta secara optimal unit.

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

```
    Golden Ratio : 1.61803
```

Ada string khusus bernama "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 string tersebut. Hal ini juga berlaku untuk ekspresi (lihat di bawah).

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

```
        1234.50
```

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

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

```
    affe
    charlie
    bravo
```

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

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

```
    affe
    charlie
```

```
bravo
affe
charlie
bravo
```

String dapat berisi karakter Unicode. Secara internal, string-string ini berisi kode UTF-8. Untuk menghasilkan string semacam 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
```

```
α = 45°
```

I

Dalam komentar, entitas yang sama seperti α, β, dan sebagainya dapat digunakan. Ini mungkin menjadi alternatif cepat untuk Latex. (Rincian lebih lanjut tentang komentar di bawah ini).

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")
```

```
Real 1 x 20 matrix
     196.00       32.00      105.00      115.00      ...
```

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 mengubah string yang berisi entitas dalam suatu variabel menjadi string Unicode.

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

```
We have α=β.
```

Ini juga memungkinkan untuk menggunakan entitas numerik.

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

```
Ähnliches
```

Nilai Boolean

Nilai boolean merupakan implementasi dari aljabar Boolean yang dikembangkan oleh George Boole. Sistem ini mendefinisikan struktur matematika dengan dua nilai kebenaran: benar (true) dan salah (false). Aljabar Boolean mematuhi aksioma-aksioma tertentu termasuk sifat komutatif, asosiatif, distributif, dan hukum komplemen.

Terdapat tiga operasi primitif dalam logika boolean:
1. Konjungsi (AND): Menghasilkan true hanya jika kedua operand true
2. Disjungsi (OR): Menghasilkan true jika minimal satu operand true
3. Negasi (NOT): Membalik nilai boolean dari true ke false atau sebaliknya

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

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

```
            0.00
            1.00
```

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

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

```
            1.00
```

Operator boolean mengikuti aturan bahasa matriks.

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

```
    Real 1 x 10 matrix
            0.00        0.00        0.00        0.00      ...
            6.00        7.00        8.00        9.00        10.00
```

Anda dapat menggunakan fungsi nonzeros() untuk mengekstrak elemen-elemen tertentu dari sebuah vektor. Dalam contoh ini, kita menggunakan kondisi isprime(n).

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

```
    Real 1 x 50 matrix
            2.00        3.00        5.00        7.00      ...
```

```
>N[nonzeros(isprime(N))] //pilih anggota2 N yang prima
```

```
    Real 1 x 25 matrix
            2.00        3.00        5.00        7.00      ...
```

Format Output

Format output default EMT mencetak 12 digit. Untuk memastikan bahwa yang kita lihat adalah bentuk default, kita perlu mengatur ulang formatnya.

```
>defformat; pi
```

```
    3.14159265359
```

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

```
>longest pi
```

```
            3.141592653589793
```

Berikut adalah representasi heksadesimal internal dari bilangan ganda.

```
>printhex(pi)
```

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

Format output dapat diubah secara permanen dengan perintah format.

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

```
          0.33333
          3.14159
          0.84147
```

Format defaultnya adalah format(12).

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

```
    0.333333333333
```

Fungsi seperti “shortestformat”, ‘shortformat’, dan “longformat” bekerja pada vektor dengan cara berikut.

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

```
       0.52    0.51    0.84    0.61    0.37    0.36    0.93    0.85
       0.76    0.17    0.87    0.98   0.091    0.96    0.39    0.34
       0.77    0.86    0.84    0.66    0.74    0.95    0.35   0.048
```

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

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

```
    3.1416
```

Fungsi "longestformat" juga mengatur format skalar.

```
>longestformat; pi
```

```
    3.141592653589793
```

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

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

Ketepatan internal EMT sekitar 16 digit desimal, yang sesuai dengan standar IEEE. Angka-angka disimpan dalam format internal ini.

Namun, format output EMT dapat diatur dengan fleksibel.

```
>longestformat; pi,

    3.141592653589793

>format(10,5); pi

      3.14159
```

Defaultnya adalah defformat().

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

Terdapat operator pendek yang hanya mencetak satu nilai. Operator "longest" akan mencetak semua digit yang valid dari sebuah angka.

```
>longest pi^2/2

          4.934802200544679
```

Ada juga operator singkat untuk mencetak hasil dalam format pecahan. Kita sudah menggunakannya di atas.

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

    25/12
```

Karena format internal menggunakan cara biner untuk menyimpan angka, nilai 0.1 tidak akan direpresentasikan secara tepat. Kesalaham ini sedikit bertambah, seperti yang dapat 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 pengaturan default "longformat", anda tidak akan menyadarinya. Untuk memudahkan, output dari angka yang sangat kecil adalh 0.

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

    0
```

Ekspresi

String atau nama dapat digunakan untuk menyimpan ekspresi matematika, yang dapat dievaluasi oleh EMT. Untuk ini, gunakan tanda kurung setelah ekspresi. Jika Anda bermaksud menggunakan string sebagai ekspresi, gunakan konvensi untuk menamainya “fx” atau “fxy” dan seterusnya. 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 tersebut ditugaskan ke x, y, dan z dalam urutan tersebut. Parameter tambahan dapat ditambahkan menggunakan parameter yang telah ditugaskan.

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

```
-0.919535764538
```

Perhatikan bahwa ekspresi akan selalu menggunakan variabel global, bahkan jika ada variabel dengan nama yang sama di dalam fungsi. (Jika tidak, evaluasi ekspresi di 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" daripada 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 refernsi, kami mencatat bahwa kumpulan panggilan (yang dibahas di tempat lain) dapat berisi ekspresi. Oleh karena itu, kmai 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.
erhatikan bahwa mendefinisikan fungsi dengan nama yang sama seperti ekspresi simbolik global akan menghapus variabel ini untuk menghindari kebingungan antara ekspresi simbolik dan fungsi.

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

```
12
```

Secara konvensional, ekspresi simbolik atau numerik sebaiknya diberi nama fx, fxy, dan seterusnya. Skema penamaan ini tidak boleh digunakan untuk fungsi.

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

Sebuah bentuk khusus dari ekspresi memungkinkan penggunaan variabel apa pun sebagai parameter tanpa nama dalam evaluasi ekspresi, bukan hanya “x”, "y", dan sebagainya. 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 memerlukan ekspresi dalam “x”.

Cara paling sederhana untuk mendefinisikan fungsi sederhana adalah dengan menyimpan rumusnya dalam ekspresi simbolik atau numerik. Jika variabel utama 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 telah ditetapkan.

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

```
-0.425
```

Sebuah ekspresi tidak perlu bersifat simbolik. Hal ini diperlukan jika ekspresi tersebut mengandung fungsi yang hanya dikenal di kernel numerik, bukan di Maxima.

Matematika Simbolik

EMT melakukan perhitungan simbolik dengan bantuan Maxima. Untuk detailnya, mulailah dengan tutorial berikut, atau telusuri referensi untuk Maxima. Para ahli Maxima perlu memperhatikan bahwa terdapat perbedaan sintaksis antara sintaksis asli Maxima dan sintaksis default ekspresi simbolik di EMT.

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

Pertama-tama, Maxima memiliki aritmetika “infinite” yang dapat menangani angka-angka sangat besar.

```
>$&44!
```

Dengan cara ini, anda dapt 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 hal ini (demikian pula bagian numerik dari EMT).

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

Untuk mempelajari lebih lanjut tentang suatu fungsi, klik ganda pada fungsi tersebut. Misalnya, coba klik ganda pada “&binomial” di baris perintah sebelumnya. Ini akan membuka dokumentasi Maxima yang disediakan oleh pengembang program tersebut.

Anda akan mengetahui bahwa cara berikut 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 “with”.

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

Dengan cara ini, Anda dapat menggunakan solusi suatu persamaan dalam persamaan lain.

Ekspresi simbolik dicetak oleh Maxima dalam bentuk 2D. Alasan di balik ini adalah bendera simbolik khusus dalam string.

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

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

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

Ekspresi simbolik diparsing oleh Euler. Jika Anda memerlukan sintaks yang kompleks dalam satu ekspresi, Anda dapat mengapit ekspresi tersebut dengan tanda kutip ganda (“...”). Penggunaan ekspresi yang lebih kompleks dari ekspresi sederhana dimungkinkan, tetapi sangat tidak disarankan.

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

25

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

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

Sekali lagi, % merujuk pada hasil sebelumnya.

Untuk memudahkan, kita menyimpan solusi ke dalam 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))
```

Input langsung perintah Maxima juga tersedia. Mulai baris perintah dengan “::”. Sintaks Maxima disesuaikan dengan sintaks EMT (disebut “mode kompatibilitas”).

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

2432902008176640000

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

$$2^8 \, 3^4 \, 5^2 \, 7$$

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

```
  18  8  4  2
2   3  5  7  11 13 17 19
```

Jika anda adalah 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
```

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

18551.64488782208
```

```
>fx(5)

    18551.6448878
```

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

Perintah berikut juga menunjukkan bahwa Maxima dapat mengevaluasi suatu ekspresi secara numerik menggunakan float().

```
>&(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 suatu ekspresi, anda dapat menggunakan perintah tex.

```
>tex(fx)
```

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

Ekspresi simbolik dapat dievaluasi sama seperti 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
```

Tugas tersebut juga dapat bersifat simbolik.

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

Perintah "solve" digunakan untuk menyelesaikan ekspresi simbolik untuk suatu variabel dalam Maxima. Hasilnya adalah vektor solusi.

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

Dibandingkan dengan perintah numerik “solve” di Euler, yang memerlukan nilai awal, dan nilai target secara opsional.

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

```
1.56155281281
```

Nilai numerik dari solusi simbolik dapat dihitung dengan mengevaluasi hasil simbolik. Euler akan memeriksa penugasan x= dan seterusnya. Jika Anda tidak memerlukan hasil numerik untuk perhitungan lebih lanjut, Anda juga dapat membiarkan Maxima menemukan nilai numeriknya.

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

```
[-3.23607,]
```

Untuk mendapatkan solusi simbolik tertentu, dapat menggunakan "with" dan sebuah 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 simbolik dapat memiliki bendera, yang menandakan perlakuan khusus dalam Maxima. Beberapa bendera dapat digunakan sebagai perintah, sementara yang lain tidak. Bendera ditambahkan dengan “|” (bentuk yang lebih baik dari “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 didefinisikan dengan perintah “function”. Fungsi ini dapat berupa fungsi satu baris atau fungsi multi-baris.
Fungsi satu baris dapat bersifat numerik atau simbolik. Fungsi numerik satu baris didefinisikan dengan “:=”.

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

Untuk gambaran umum, kami menampilkan semua definisi yang mungkin untuk fungsi satu baris. Sebuah fungsi dapat dievaluasi sama seperti fungsi Euler bawaan.

```
>f(2)
```

```
    4.472135955
```

Fungsi ini juga akan berfungsi untuk vektor, mengikuti bahasa matriks Euler, karena ekspresi yang digunakan dalam fungsi tersebut telah divektorisasi.

```
>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 digambar. Alih-alih menggunakan ekspresi, kita hanya perlu memberikan nama fungsi.

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

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

```
    0.786151377757
```

Secara default, jika anda perlu mengganti fungsi bawaan, anda harus menambahkan kata kunci “overwrite”. Mengganti fungsi bawaan dapat berbahaya dan dapat menyebabkan masalah bagi fungsi lain yang bergantung padanya.

Anda masih dapat memanggil fungsi bawaan sebagai “_...”, jika fungsi tersebut berada di inti Euler.

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

```
    0.707106781187
```

Kita sebaiknya menghapus redefinisi 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
```

Jika parameter ini diabaikan, nilai default akan digunakan.

```
>f(4)
```

```
    16
```

Menyetelnya akan mengganti nilai default.

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

```
    80
```

Parameter yang ditugaskan juga akan ditimpa. Hal ini digunakan oleh banyak fungsi Euler seperti plot2d, plot3d.

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

```
    16
```

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

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

```
    24
```

Tetapi parameter yang ditentukan secara khusus akan menggantikan nilai global.

Jika argumen tidak terdapat dalam daftar parameter yang telah didefinisikan sebelumnya, argumen tersebut harus dideklarasikan dengan “:=”!

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

```
    20
```

Fungsi simbolik didefinisikan dengan “&=”. Fungsi ini didefinisikan dalam Euler dan Maxima, dan berfungsi di kedua lingkungan tersebut. Ekspresi definisi dijalankan melalui Maxima sebelum definisi dijalankan.

```
>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, hal ini hanya akan berfungsi jika EMT dapat menginterpretasikan semua yang ada di dalam fungsi tersebut.

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

```
    178.635099908
```

Mereka dapat digunakan untuk mendefinisikan fungsi atau ekspresi simbolik lainnya.

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

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

```
    0.703467422498
```

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

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

```
    0.703467422498
```

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

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

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

$$(x+2)^n$$

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

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

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

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

```
    625
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$$x^3-x$$

Dengan &=, 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 definit seperti

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

yang tidak dapat dievaluasi secara simbolik.

Jika kita mendefinisikan ulang fungsi menggunakan 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 ini dapat dipanggil dengan atau tanpa parameter “base”.

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

```
    2
    6.7
```

Selain itu, juga memungkinkan untuk menggunakan parameter yang telah ditentukan.

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

```
2
```

Seringkali, kita ingin menggunakan fungsi untuk vektor di satu tempat, dan untuk elemen individu di tempat lain. Hal ini dapat dilakukan dengan parameter vektor.

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

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

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

Fungsi simbolik semacam itu dapat digunakan untuk variabel simbolik.

Namun, fungsi tersebut juga dapat digunakan untuk vektor numerik.

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

```
17
```

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

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

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

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

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

$$0$$

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

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

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

Untuk merangkum

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

Penyelesaian Ekspresi

Ekspresi dapat diselesaikan secara numerik dan simbolik.

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

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

```
1.41421356237
```

Ini juga berlaku untuk ekspresi simbolik. Pertimbangkan 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])
```

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

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

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

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

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

```
    0.966715594851
    2
```

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

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

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

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

```
>longest sol()
```

```
        -0.6180339887498949       1.618033988749895
```

Untuk menggunakan simbol solusi dalam ekspresi lain, cara termudah adalah “with”.

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

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

$$0$$

Penyelesaian sistem persamaan secara simbolis dapat dilakukan dengan menggunakan vektor persamaan dan pemecah simbolis solve(). Hasilnya adalah daftar dari daftar persamaan.

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

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

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

dengan 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
```

Ini juga berlaku untuk ekspresi. Namun, dalam hal ini, elemen daftar yang diberi nama harus digunakan. (Lebih lanjut tentang daftar dalam tutorial tentang sintaks EMT).

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

```
    2.54116291558
```

Menyelesaikan Pertidaksamaan

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

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

```
    Maxima said:
    file_search1: fourier_elim not found in
                  file_search_maxima,file_search_lisp.
     -- an error. To debug this try: debugmode(true);

    Error in:
    &load(fourier_elim) ...
                        ^
```

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

$$fourier\_elim\left(\left[x^2-1>0\right],[x]\right)$$

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

$$fourier\_elim\left(\left[x^2-1<0\right],[x]\right)$$

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

$$fourier\_elim\left(\left[x^2-1\neq 0\right],[x]\right)$$

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

$$fourier\_elim\left(\left[x\neq 6\right],[x]\right)$$

```
>$&fourier_elim([x < 1, x > 1],[x]) // tidak memiliki penyelesaian
>$&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 pembahasan rinci mengenai bahasa matriks pada Euler.

Vektor dan matriks dimasukkan dengan kurung siku, elemen dipisahkan dengan koma, dan 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 bekerja elemen demi elemen.

```
>A.A
```

```
               7             10
              15             22
```

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

```
               1              4
               9             16
```

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

```
              37             54
              81            118
```

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

```
              37             54
              81            118
```

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

```
               1              1
               1              1
```

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

```
        0.333333       0.666667
            0.75              1
```

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

```
   -2
  2.5
```

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

```
   -2
  2.5
```

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

```
    1     0
    0     1
```

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

```
    1     0
    0     1
```

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

```
    1     4
    9    16
```

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

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

```
    9
   16
```

Jika salah satu operand adalah vektor atau skalar, maka akan diekspansi secara alami.

```
>2*A
```

```
    2     4
    6     8
```

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

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

```
    1     4
    3     8
```

Jika itu adalah vektor baris, maka ini 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 dengan A.

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

                1              2
                1              2

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

                1              4
                3              8
```

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

Sekali lagi, ingatlah bahwa ini bukan hasil kali matriks!

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

    55

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

    55
```

Bahkan operator seperti < atau == berfungsi 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 menggunakan fungsi sum().

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

    5
```

Euler memiliki operator perbandingan, seperti “==”, yang memeriksa kesamaan.

Kita mendapatkan vektor yang berisi angka 0 dan 1, di mana 1 mewakili nilai benar.

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

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

Dari vektor tersebut, “nonzeros” memilih elemen-elemen yang tidak nol.

Dalam hal ini, kita mendapatkan indeks dari semua elemen yang nilainya lebih besar dari 50.

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

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

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

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

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

Sebagai contoh, mari kita cari semua kuadrat dari angka 1 hingga 1000 yang merupakan kelipatan 5 modulo 11 dan kelipatan 3 modulo 13.

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

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

EMT tidak sepenuhnya efektif untuk perhitungan bilangan bulat. Secara internal, ia menggunakan bilangan floating point presisi ganda. Namun, ia seringkali sangat berguna.

Kita dapat menguji nilai 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, terdapat fungsi mnonzeros().

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

```
 0.765761   0.401188   0.406347   0.267829
  0.13673   0.390567   0.495975   0.952814
 0.548138   0.006085   0.444255   0.539246
```

Fungsi ini mengembalikan indeks dari elemen-elemen yang nilainya bukan nol.

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

```
 1  4
 2  1
```

```
                2             2
                3             2
```

Indeks-indeks ini dapat digunakan untuk menetapkan elemen-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 mengatur elemen pada indeks tertentu menjadi entri dari 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-elemen dalam sebuah vektor.

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

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

Fungsi lain yang berguna adalah extrema, yang mengembalikan nilai minimum dan maksimum dalam setiap baris matriks beserta 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 maksimum di setiap baris.

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

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

Tentu saja, ini sama dengan fungsi max().

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

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

Namun, dengan fungsi 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 Lain (Membuat Matriks)

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

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

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

Demikian pula, kita dapat menggabungkan dua matriks 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 jumlah barisnya tidak sama, matriks yang lebih pendek akan diisi dengan 0.

Ada pengecualian untuk aturan ini. Bilangan real yang terkait dengan 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
```

Hal ini memungkinkan untuk membuat matriks dari vektor baris dan vektor kolom.

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

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

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

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

Tujuan utama dari hal ini adalah untuk menginterpretasikan vektor ekspresi untuk vektor kolom.

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

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

Untuk mendapatkan ukuran 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 vektor, terdapat fungsi 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 cara berikut.

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

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

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

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

```
          0.66566       0.831835
            0.977       0.544258
```

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

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

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

Dengan fungsi berikut, kita dapat menggunakan 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() membalik urutan baris atau kolom dari sebuah matriks. Artinya, fungsi flipx() membalik secara horizontal.

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

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

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

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

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

Sebuah fungsi khusus adalah drop(v,i), yang menghapus elemen-elemen dengan indeks i dari vektor v.

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

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

Perhatikan bahwa vektor i dalam drop(v,i) merujuk pada indeks elemen dalam v, bukan nilai elemen 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 v yang telah diurutkan.

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

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

Seperti yang Anda lihat, tidak ada masalah jika menyertakan indeks di luar rentang (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 menetapkan 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 tetapkan 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 dari suatu matriks juga dapat diekstraksi dari matriks tersebut. Untuk memperlihatkan hal ini, kita merestrukturisasi vektor 1:9 menjadi matriks 3x3.

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

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

Sekarang kita dapat mengekstraksi bagian diagonal.

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

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

Misalnya, kita dapat membagi matriks dengan diagonalnya. Bahasa matriks memastikan bahwa vektor kolom d diterapkan pada baris matriks secara berurutan.

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

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

Vektorisasi

Hampir semua fungsi di Euler juga berfungsi untuk masukan matriks dan vektor, asalkan 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 menggambar grafik fungsi (alternatifnya menggunakan ekspresi).

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

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

alam contoh berikut, kita menghasilkan vektor nilai t[i] dengan selisih 0.1 dari -1 hingga 1. Kemudian kita menghasilkan vektor nilai fungsi

```
>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, perkalian vektor kolom dengan vektor baris akan diperluas menjadi matriks jika operator diterapkan. Dalam contoh berikut, v' adalah vektor transpos (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 hal ini cukup berbeda dengan perkalian matriks. Perkalian 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 . melambangkan perkalian matriks, dan A' melambangkan transposisi. Matriks 1x1 dapat digunakan seperti bilangan real.

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

```
5
25
```

Untuk mentransposisi sebuah matriks, kita menggunakan tanda apostrof.

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

```
                1
                2
                3
                4
```

Jadi, kita dapat menghitung perkalian matriks A dengan vektor b.

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

```
               30
               70
```

Perhatikan bahwa v masih merupakan vektor baris. Oleh karena itu, v'.v berbeda dengan v.v'.

```
>v'.v
```

```
                1             2             3             4
                2             4             6             8
                3             6             9            12
                4             8            12            16
```

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

```
>v.v'
```

```
     30
```

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

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

```
     30
```

Operator dan fungsi mematuhi bahasa matriks yang digunakan oleh Euler.

Berikut adalah ringkasan aturan-aturan tersebut.

- Sebuah fungsi yang diterapkan pada vektor atau matriks diterapkan pada setiap elemennya.
- Operator yang bekerja 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 wajar sehingga memiliki ukuran yang sama.

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

Berikut adalah contoh sederhana dengan operator ^.

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

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

Ini adalah kasus yang lebih rumit. Perkalian vektor baris dengan vektor kolom akan diekspansi keduanya dengan cara menduplikasi.

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

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

Perhatikan bahwa hasil kali skalar menggunakan hasil kali matriks, bukan tanda bintang *!

```
>v.v'
```

```
    14
```

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

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

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

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

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

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

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

Untuk menggabungkan matriks dan vektor, 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 dari sebuah 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 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 juga dapat berupa vektor baris dari indeks. : menandakan 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 dari sebuah matriks dapat diakses seolah-olah mereka adalah vektor.

```
>A{4}
```

```
      4
```

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 dalam tabel, mari kita kembalikan ke format default, dan hitung tabel nilai sinus dan kosinus. Perhatikan bahwa sudut secara default menggunakan satuan radian.

```
>defformat; w=0°:45°:360°; w=w'; deg(w)
```

```
                  0
                 45
                 90
                135
```

```
180
225
270
315
360
```

Sekarang kita tambahkan kolom ke dalam 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 merupakan tabel dari t^i untuk satu nilai i. Artinya, matriks tersebut memiliki elemen-elemen

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

Fungsi yang tidak berfungsi untuk masukan vektor harus “divektorisasi”. Hal ini dapat dicapai dengan menggunakan kata kunci “map” dalam definisi fungsi. Kemudian fungsi tersebut akan dievaluasi untuk setiap elemen dari parameter vektor.

Fungsi integrasi numerik integrate() hanya berfungsi untuk batas interval skalar. Oleh karena itu, kita perlu memvektorisasinya.

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

Kata kunci “map” memvektorisasi fungsi. Fungsi ini sekarang akan bekerja
ntuk 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 kurung siku.

```
>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 lengkap dari sebuah matriks.

```
>A[2]
```

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

Dalam kasus vektor baris atau kolom, ini mengembalikan 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 yang kosong.

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

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

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

i sini kita ingin 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 yang diurutkan ulang dari A.

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

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

Trik indeks juga berlaku untuk 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 “:”, menandakan 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 dari A.

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

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

Sekarang mari kita ubah elemen-elemen dari A dengan mengisi submatriks dari A dengan nilai tertentu. Hal ini memang mengubah matriks A yang disimpan.

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

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

Kita juga dapat memberikan nilai pada baris A.

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

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

Kita bahkan dapat mengisi sub-matriks jika ukurannya sesuai.

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

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

Selain itu, beberapa jalan pintas diperbolehkan.

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

```
                0              0             -1
                0              0              6
                7              8              9
```

Peringatan: Indeks yang melebihi batas akan mengembalikan matriks kosong atau pesan kesalahan, tergantung pada pengaturan sistem. Pengaturan default adalah pesan kesalahan. Perlu diingat, bagaimanapun, bahwa indeks negatif dapat digunakan untuk mengakses 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]
```

Hal ini seringkali diperlukan untuk mengetahui indeks vektor yang telah 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]
```

Hal ini juga berlaku untuk vektor string.

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

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

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

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

Seperti yang anda lihat, posisi entri ganda bersifat acak.

```
>ind
```

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

Fungsi “unique” mengembalikan daftar berurutan dari elemen-elemen unik dalam 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 juga berlaku untuk vektor string.

```
>unique(s)
```

```
a
aa
d
e
```

Aljabar Linier

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

Untuk sistem linier Ax=b, Anda dapat menggunakan Algoritma Gauss, matriks invers, atau regresi linier. Operator A\b menggunakan versi dari Algoritma Gauss.

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

```
 -4
4.5
```

Sebagai contoh lain, kita menghasilkan matriks 200x200 dan menjumlahkan baris-barisnya. Kemudian kita menyelesaikan persamaan Ax=b menggunakan matriks invers. Kita mengukur kesalahan sebagai penyimpangan maksimum dari semua elemen terhadap 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 tersebut tidak memiliki solusi, regresi 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 simbolik. Tentu saja, Maxima dapat digunakan untuk masalah aljabar linier sederhana seperti ini. Kita dapat mendefinisikan matriks untuk Euler dan Maxima dengan &:=, lalu menggunakannya dalam ekspresi simbolik. Bentuk [...] yang biasa digunakan 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-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 yang berisi dua vektor nilai eigen dan multiplisitasnya.

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

Untuk mengekstrak vektor eigen tertentu, diperlukan pengindeksan yang cermat.

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

$$[1, -1]$$

Matriks simbolik dapat dievaluasi secara numerik menggunakan Euler sama seperti ekspresi simbolik lainnya.

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

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

Dalam ekspresi simbolik, gunakan "with".

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

Akses ke baris matriks simbolik berfungsi sama seperti pada matriks numerik.

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

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

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

Ada fungsi simbolik di Maxima untuk membuat vektor dan matriks. Untuk hal ini, silakan merujuk ke dokumentasi Maxima atau ke 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 pengenalan tentang Maxima.

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

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

Euler juga memiliki fungsi yang sangat kuat, xinv(), yang melakukan perhitungan yang lebih intensif dan menghasilkan hasil yang lebih akurat.

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 lebih lanjut mengenai hal ini.

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

Nilai Numerik dalam Ekspresi Simbolik

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

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

```
                  1            3.14159
                  4                  5
```

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

```
>$&A
```

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

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

Maxima juga dapat melakukan perhitungan dengan bilangan floating point, bahkan dengan bilangan floating point besar yang memiliki 32 digit. Namun, proses perhitungannya jauh lebih lambat.

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

Ketepatan angka floating point (desimal) besar dapat diubah.

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

```
            3.14159265358979323846264338327950288419716939937510582097494\
459230781640628620899862803482534211706 8b0
```

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

Perhatikan bahwa hal ini hanya diperlukan jika variabel telah didefinisikan dengan “:=” atau “=” sebagai variabel numerik.

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

Demo - Suku Bungan

Di bawah ini, kami menggunakan Euler Math Toolbox (EMT) untuk perhitungan suku bunga. Kami melakukannya secara numerik dan simbolik untuk menunjukkan kepada anda bagaimana Euler dapat digunakan untuk menyelesaikan masalah nyata dalam kehidupan sehari-hari.

Misalkan Anda memiliki modal awal sebesar 5000 (misalnya dalam dolar).

```
>K=5000
```

```
    5000
```

Sekarang kita asumsikan suku bunga sebesar 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 lebih mudah menggunakan faktor tersebut.

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

```
    1.03
    5150
```

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

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

```
    6719.58189672
```

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

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

```
         6719.58
```

Mari kita cetak angka tersebut 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 jika kita ingin mengetahui hasil sementara dari tahun 1 hingga tahun 9? Untuk hal ini, bahasa matriks Euler sangat membantu. Anda tidak perlu menulis loop, tetapi cukup masukkan

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

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

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

```
>short 0:10
```

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

Kemudian, semua operator dan fungsi dalam Euler dapat diterapkan pada vektor elemen demi elemen. Jadi

```
>short q^(0:10)
```

```
    [1,  1.03,  1.0609,  1.0927,  1.1255,  1.1593,  1.1941,  1.2299,
    1.2668,  1.3048,  1.3439]
```

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

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

Tentu saja, cara yang 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)
```

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 melalui tahun-tahun tersebut. Euler menyediakan banyak solusi untuk hal ini.

Cara termudah adalah fungsi "iterate", yang mengulang suatu fungsi sebanyak kali yang ditentukan.

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

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

Kami dapat mencetak angka tersebut dengan cara yang mudah dibaca, menggunakan format kita dengan jumlah desimal yang 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, kita menggunakan indeks dalam kurung siku.

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

```
        5150.00
        5000.00     5150.00     5304.50
```

Yang mengejutkan, 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 asli.

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

```
        6719.60
        6719.58
```

Perbedaannya sangat kecil.

Menyelesaikan Persamaan

Sekarang kita akan membahas fungsi yang lebih canggih, yang menambahkan jumlah uang tertentu setiap tahun.

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

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

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

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

Bagaimana jika kita menghilangkan jumlah yang sama setiap tahun?

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

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

Kita melihat bahwa uangnya berkurang. Jelas, jika kita hanya mendapatkan 150 bunga pada tahun pertama, tetapi mengurangkan 200, kita akan rugi setiap tahun.

Bagaimana cara menentukan berapa lama uang tersebut akan bertahan? Kita harus membuat loop untuk ini. Cara termudah adalah dengan mengulanginya 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. Fungsi ini dapat menerima 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. Anggaplah kita tahu bahwa nilainya 0 setelah 50 tahun. Berapa suku bunga yang diperlukan?

Ini adalah pertanyaan yang hanya dapat dijawab secara numerik. Di bawah ini, kita akan menyusun rumus-rumus yang diperlukan. Kemudian anda akan melihat bahwa tidak ada rumus sederhana untuk suku bunga. Namun, untuk saat ini, kita bertujuan untuk mendapatkan solusi numerik.

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

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

Iterasi ini sama seperti di atas.

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

Selain itu, kita hanya tertarik pada nilai terakhir. Oleh karena itu, kita mengambil indeks [-1].

Mari kita coba sebuah tes.

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

          -19.83
```

Sekarang kita dapat menyelesaikan permasalahan kita

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

            3.15
```

Fungsi solve() menyelesaikan persamaan expression=0 untuk variabel x. Hasilnya adalah 3,15% per tahun. Kami menggunakan nilai awal 3% untuk algoritma. Fungsi solve() selalu memerlukan nilai awal.

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

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

         -336.08
```

Catat bahwa anda tidak dapat menghitung jumlah tahun, karena fungsi kami mengasumsikan n sebagai nilai bilangan bulat.

Solusi Simbolik untuk Masalah Suku Bunga

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

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

Sekarang kita dapat mengulang proses ini.

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

Kita melihat pola. Setelah n periode, kita memiliki

Rumus tersebut adalah rumus untuk jumlah geometris, yang dikenal oleh Maxima.

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

Ini agak rumit. Jumlah tersebut dievaluasi dengan bendera “simpsum” untuk menyederhanakannya 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)
```

Fungsi ini melakukan hal yang sama seperti fungsi f kita sebelumnya. Namun, fungsi ini 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? Perkiraan awal kita adalah 30 tahun.

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

```
            20.51
```

Jawaban ini menyatakan bahwa hal tersebut akan menjadi negatif setelah 21 tahun.

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

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

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

Biasanya rumus ini dinyatakan dalam bentuk

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

Kita dapat menghitung nilai laju R secara simbolik.

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

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

Tentu saja, kita memiliki batasan berikut.

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

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

ersamaan tersebut juga dapat diselesaikan untuk n. Persamaan tersebut akan terlihat lebih rapi jika kita melakukan penyederhanaan pada persamaan tersebut.

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

Latihan Soal

1. Hitunglah operasi berikut ini

$$(3x^2 - 2x - x^3 + 2) - (5x^2 - 8x - x^3 + 4)$$

```
>$ratsimp((3*x^2-2*x-x^3+2)-(5*x^2-8*x-x^3+4))
```

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

Fungsi "ratsimp" atau Rational Simplification adalah fungsi dalam sistem aljabar Maxima yang digunakan untuk menyederhanakan ekspresi rasional.

Dengan fungsi ini, EMT akan memproses dengan mengikuti aturan dasar aljabar untuk pengurangan polinomial yang meliputi distribusi tanda negatif, pengelompokkan suku sejenis, dan penyelesaian operasi yang akan menghasilkan bentuk paling sederhana.

2. Hitunglah operasi berikut ini

$$(5x^2 + 4xy - 3y^2 + 2) - (9x^2 - 4xy + 2y^2 - 1)$$

```
>$ratsimp((5*x^2+4*x*y-3*y^2+2)-(9*x^2-4*x*y+2*y^2-1))
```

$$-5\,y^2+8\,x\,y-4\,x^2+3$$

3. Hitunglah operasi berikut ini

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

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

$$x^4-3\,x^3-4\,x^2+9\,x-3$$

4. Hitunglah operasi berikut ini

$$(2x^4-3x^2+7x)-(5x^3+2x^2-3x+5)$$

```
>$ratsimp((2*x^4-3*x^2+7*x)-(5*x^3+2*x^2-3*x+5))
```

$$2\,x^4-5\,x^3-5\,x^2+10\,x-5$$

5. Hitunglah operasi berikut ini

$$(2x^2+12xy-11)+(6x^2-2x+4)+(-x^2-y-2)$$

```
>$ratsimp((2*x^2+12*x*y-11)+(6*x^2-2*x+4)+(-x^2-y-2))
```

$$(12\,x-1)\,y+7\,x^2-2\,x-9$$

Fungsi "ratsimp" atau Rational Simplification adalah fungsi dalam sistem aljabar Maxima yang digunakan untuk menyederhanakan ekspresi rasional.

Dengan fungsi ini, EMT akan menyederhanakan ekspresi aljabar dengan menggabungkan suku-suku sejenis. Output yang dihasilkan seperti itu karena fungsi ini mencoba menyajikan ekspresi dalam bentuk kanonik (paling sederhana) yang terorganisir.

6. Hitunglah operasi berikut ini

```
>$expand((3*y+4)*(3*y-4))
```

$$9\,y^2-16$$

Fungsi "expand" pada EMT digunakan untuk melakukan perkalian polinomial, menghilangkan tanda kurung, dan menggabungkan suku-suku sejenis. Dalam operasi ini, fungsi "expand" mengenali bahwa ekspresi tersebut merupakan perkalian dua binomial.

7. Hitunglah operasi berikut ini

$$(2x+3y+4)(2x+3y-4)$$

```
>$expand((2*x+3*y+4)*(2*x+3*y-4))
```

$$9\,y^2+12\,x\,y+4\,x^2-16$$

8. Hitunglah operasi berikut ini

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

```
>$expand((x+1)*(x-1)*(x^2+1))
```

$$x^4-1$$

9. Hitunglah operasi berikut ini

$$(3x-2y)(3x+2y)$$

```
>$expand((3*x-2*y)*(3*x+2*y))
```

$$9\,x^2-4\,y^2$$

10. Hitunglah operasi berikut ini

$$(3x-2)^2$$

```
>$expand((3*x-2)^2)
```

$$9\,x^2-12\,x+4$$

Fungsi "expand" di sini memecahkan ekspresi ini denan mengalikan binomial tersebut dengan dirinya sendiri (pangkat 2). Terdapat 2 metode yang menunjukkan bagaimana EMT dapat menghasilkan jawaban tersebut.
Metode yang pertama dengan mengalikan suku per suku, dan yang kedua menggunakan rumus aljabar di mana kuadrat sempurna binomial memiliki rumus umum:

11. Faktorkan

```
>$solve(y^2-18*y+81)
```

$$[y=9]$$

Fungsi "solve" digunakan untuk mencari akar (solusi) dari sebuah persamaan. EMT melalui mesin aljabar Maxima, mengenali bahwa ekspresi y

-18y+81 adalah contoh dari polinomial kuadrat sempurna (perfect square trinomial). Kemudian Maxima akan memfaktorkan, menyelesaikan, dan menemukan solusi dari persamaan sederhana untuk y.

Hasil [9] menunjukkan bahwa persamaan tersebut memiliki satu-satunya solusi, yaitu y=9. Solusi ini terkadang disebut akar ganda karena secara teknis merupakan dua akar yang sama, yang merupakan karakteristik dari polinomial kuadrat sempurna

12. Temukan faktor selisih kuadrat dari

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

$$(z-9)\,(z+9)$$

Fungsi "factor" di EMT digunakan untuk menguraikan polinomial menjadi faktor-faktor perkaliannya yang paling sederhana. Dalam kasus ini, factor mengenali ekspresi z^2-81 sebagai selisih kuadrat, yang merupakan pola aljabar khusus.

Rumus umum untuk selisih kuadrat adalah:

$$a^2-b^2=(a-b)(a+b)$$

Mengaplikasikan rumus ini, EMT menguraikan ekspresi menjadi dua faktor binomial: satu dengan tanda minus dan satu lagi dengan tanda plus. Jadi, output yang dihasilkan oleh EMT adalah representasi dari faktorisasi polinomial tersebut.

13. Temukan faktor selisih kuadrat dari

$$7pq^4-7py^4$$

```
>$factor(7*p*q^4-7*p*y^4)
```

$$-7\,p\,(y-q)\,(y+q)\,\left(y^2+q^2\right)$$

14. Temukan faktor selisih kuadrat dari

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

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

$$-25\,a\,(z-b)\,(z+b)\,\left(z^2+b^2\right)$$

15. Temukan faktor selisih kuadrat dari

$$8a^2-8b^2$$

```
>$factor(8*a^2-8*b^2)
```

$$-8\,(b-a)\,(b+a)$$

16. Temukan faktor selisih kuadrat dari

$$16x^2-9$$

```
>$factor(16*x^2-9)
```

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

17. Faktorkan kuadrat dari binomial

$$1-8x+16x^2$$

```
>$factor(1-8*x+16*x^2)
```

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

Fungsi "factor" di EMT digunakan untuk memecah polinomial menjadi faktor-faktornya yang lebih sederhana. EMT, melalui Maxima, mengenali ekspresi tersebut sebagai kuadrat sempurna binomial. Maxima biasanya akan mengurutkan suku-suku dalam faktor, sehingga output (4x-1)^2 adalah cara standar untuk merepresentasikan jawaban ini.

18. Faktorkan kuadrat dari binomial

$$5a^2-10ab+5b^2$$

```
>$factor(5*a^2-10*a*b+5*b^2)
```

$$5\,(b-a)^2$$

19. Faktorkan kuadrat dari binomial

$$a^3+24a^2+144a$$

```
>$factor(a^3+24*a^2+144*a)
```

$$a\,(a+12)^2$$

20. Faktorkan kuadrat dari binomial

$$4z^2+12z+9$$

```
>$factor(4*z^2+12*z+9)
```

$$(2\,z+3)^2$$

21. Faktorkan

$$y^2-\frac{8}{49}+\frac{2}{7}y$$

```
>$solve(y^2-(8/49)+(2/7)*y)
```

22. Faktorkan

$$x^2 - 3x + \frac{9}{4}$$

```
>$solve(x^2-3*x+(9/4))
```

23. Faktorkan

$$(x + 0.01)^2 - x^2$$

```
>$solve((x+0.01)^2-(x^2))
```

24. Faktorkan

$$(y - 4)^2 + 5(y - 4) - 24$$

```
>$solve((y-4)^2+5*(y-4)-24)
```

25. Faktorkan

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

```
>$solve(t^2-(27/100)+(3/5)*t)
```

26. Carilah nilai C

$$F = \frac{9}{5}C + 32$$

```
>$solve(F=(9/5)*C+32, C)
```

$$\left[C = \frac{5\,F - 160}{9}\right]$$

Fungsi "solve" di EMT digunakan untuk mencari nilai variabel yang tidak diketahui dalam suatu persamaan. Dalam hal ini, Anda meminta EMT untuk mengisolasi variabel C dari persamaan yang diberikan. Proses yang dilakukan EMT (melalui Maxima) untuk menyelesaikan persamaan ini adalah dengan cara mengeliminasi konstanta dan mengisolasi variabel. Tetapi perlu diingat bahwa output yang dihasilkan belum dalam bentuk sederhana.

27. Carilah nilai r^3

$$V = \frac{4}{3}\pi r^3$$

```
>$solve([V=((4/3)*pi*r^3)], [r^3])
```

$$\left[r^3 = \frac{3\,V}{4\,\pi}\right]$$

28. Selesaikan persamaan berikut

```
>$solve([9*(2*x+8)=20-(x+5)], [x])
```

$$[x = -3]$$

Fungsi "solve" di EMT pada kasus ini menyelesaikan persamaan langkah demi langkah untuk menemukan nilai variabel x yang memenuhi persamaan.

Proses yang dilakukan oleh EMT secara internal untuk menyelesaikan persamaan ini adalah dengan memperluas kedua sisi persamaan dengan mendistribusikan angka di luar tanda kurung (ekspansi). Langkah selanjutnya adalah dengan mengelompokkan suku sejenis yang mengandung variabel x di satu sisi dan semua suku konstanta di sisi lain. Langkah terakhir yaitu menyelesaikan persamaan tersebut dengan membagi kedua sisi persamaan tersebut dengan 19 untuk mengisolasi x.

29. Selesaikan persamaan berikut

$$y^2 + 25 = 10y$$

```
>$solve(y^2+25=10*y, y)
```

$$[y = 5]$$

30. Carilah nilai w

```
>$solve(2*w+2*h+1=p, w)
```

$$\left[w = \frac{p - 2\,h - 1}{2}\right]$$

31. Selesaikan persamaan berikut

$$24 = x(x - 2)$$

```
>$solve(24=x*(x-2), x)
```

$$[x = 6, x = -4]$$

32. Selesaikan persamaan berikut

```
>$solve((5*x^2+6*x)*(12*x^2-5*x-2)=0, x)
```

$$\left[x = -\frac{1}{4}, x = \frac{2}{3}, x = -\frac{6}{5}, x = 0\right]$$

33. Selesaikan persamaan berikut

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

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

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

34. Sederhanakan ekspresi aljabar berikut

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

```
>$ratsimp((x^2-4)/(x^2-4*x+4))
```

$$\frac{x + 2}{x - 2}$$

35. Sederhanakan ekspresi aljabar berikut

$$\frac{x^3 - 6x^2 + 9x}{x^3 - 3x^2}$$

```
>$ratsimp((x^3-6*x^2+9*x)/(x^3-3*x^2))
```

$$\frac{x-3}{x}$$

Hasil output tersebut adalah hasil yang didapatkan melalui proses faktorisasi dan penyederhanaan oleh Maxima.

Langkah pertama, Maxima mengambil pembilang dan mencari faktor-faktor umumnya, sehingga menemukan bentuk kuadrat sempurna dan pembilang menjadi:

```
>$factor(x^3-6*x^2+9*x)
```

$$(x-3)^2\ x$$

Langkah kedua, Maxima melakukan hal yang sama terhadap penyebut hingga menemukan faktor umumnya, yaitu:

```
>$factor(x^3-3*x^2)
```

$$(x-3)\ x^2$$

Langkah terakhir, setelah Maxima selesaii memfaktorkan kedua bagian, kemudian Maxima menyederhanakan kedua bagian dengan mengeliminasi faktor x. Sehingga didapatkan hasil:

```
>$ratsimp(((x-3)^2*x)/(x^2*(x-3)))
```

$$\frac{x-3}{x}$$

36. Kurangi dan sederhanakan jika mungkin

$$\frac{7}{12y}-\frac{1}{12y}$$

```
>$ratsimp((7/(12*y))-(1/(12*y)))
```

$$\frac{1}{2\,y}$$

Fungsi "ratsimp" dirancang untuk menyederhanakan ekspresi rasional (pecahan). Untuk soal ini, EMT akan melakukan dua langkah utama, yaitu, mengurangi pecahan, dan melakukan penyederhanaan hasil dari pengurangan.

37. Tambahkan dan sederhanakan jika mungkin

$$\frac{3}{x+2}+\frac{2}{x^2-4}$$

```
>$ratsimp((3/(x+2))+(2/(x^2+4)))
```

$$\frac{3\,x^2+2\,x+16}{x^3+2\,x^2+4\,x+8}$$