EMT untuk Perhitungan Aljabar

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

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

Contoh pertama

Menyederhanakan bentuk aljabar:

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

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

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

Menjabarkan:

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

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

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

Baris Perintah

Sebuah baris perintah Euler terdiri dari satu atau beberapa perintah Euler yang diikuti oleh tanda titik koma ";" atau tanda koma ",". Tanda titik koma berfungsi untuk mencegah pencetakan hasil. Koma setelah perintah terakhir dapat dihilangkan.
Baris perintah berikut hanya akan menampilkan hasil dari ekspresi, bukan penugasan ataupun perintah format.

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

```
  16.7551608191
```

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

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

```
  50.2654824574
  100.530964915
```

Baris perintah dijalankan sesuai urutan saat pengguna menekan tombol enter. Jadi, Anda akan 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 "..." maka kedua baris tersebut akan selalu dijalankan secara bersamaan.

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

```
  1.41666666667
  1.41421568627
  1.41421356237
```

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

ntuk melipat semua baris ganda tekan Ctrl+L. Maka baris-baris berikutnya hanya akan terlihat jika salah satunya berada pada fokus. Untuk melipat satu baris ganda, awali baris pertama dengan "%+ ".

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

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

```
  81
```

Euler mendukung penggunaan perulangan (loop) di dalam baris perintah, selama masih muat dalam satu baris tunggal atau dalam beberapa baris (multi-line). Dalam program, tentu saja pembatasan ini tidak berlaku. Untuk informasi lebih lanjut, silakan merujuk pada 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 masalah menggunakan multi-line. Pastikan baris diakhiri dengan " ...".

```
>x := 1.5; // comments go here before the ...
>repeat xnew:=(x+2/x)/2; until xnew~=x; ...
>   x := xnew; ...
>end; ...
>x,
```

```
  1.41421356237
```

Struktur kondisi juga dapat digunakan.

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

```
  Thought so!
```

Saat Anda menjalankan sebuah perintah, kursor dapat berada di posisi mana pun dalam baris perintah. Anda bisa kembali ke perintah sebelumnya atau melompat ke perintah berikutnya menggunakan tombol panah. Atau, Anda juga dapat mengeklik bagian komentar di atas perintah untuk langsung menuju perintah tersebut.

Ketika Anda memindahkan kursor sepanjang baris, pasangan tanda kurung buka dan kurung tutup akan disorot. Perhatikan juga baris status. Setelah tanda kurung buka pada fungsi sqrt(), baris status akan menampilkan teks bantuan untuk fungsi tersebut. Jalankan perintah dengan menekan tombol enter.

```
>sqrt(sin(10°)/cos(20°))
```

```
  0.429875017772
```

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

Anda juga dapat melakukan klik ganda pada perintah apa pun untuk membuka bantuan dari perintah tersebut. Cobalah klik ganda pada perintah exp di bawah ini pada baris perintah.

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

```
  2.5
```

Anda juga dapat menyalin dan menempelkan (copy-paste) di Euler. Gunakan Ctrl-C dan Ctrl-V untuk itu. Untuk menandai teks, seret dengan mouse atau gunakan tombol Shift bersamaan dengan tombol panah. Selain itu, Anda juga dapat menyalin tanda kurung yang disorot.

Sintaks Dasar

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

Untuk menetapkan variabel, gunakan tanda "=" atau ":=". Demi kejelasan, pengantar ini menggunakan bentuk yang terakhir. Spasi tidak berpengaruh, namun spasi di antara perintah memang diperlukan.

Beberapa perintah dalam satu baris dipisahkan dengan "," atau ";". Tanda titik koma berfungsi untuk menekan tampilan keluaran perintah. Pada akhir baris perintah, tanda koma "," dianggap ada jika tanda ";" tidak dituliskan.

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

```
  30.65625
```

EMT menggunakan sintaks pemrograman untuk ekspresi. Untuk menuliskan

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

Anda harus menuliskan tanda kurung dengan benar dan menggunakan tanda "/" untuk pecahan. Perhatikan tanda kurung yang disorot sebagai bantuan. Perlu dicatat bahwa konstanta Euler e disebut dengan E di EMT.

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

```
  8.77908249441
```

Untuk menghitung sebuah ekspresi yang rumit seperti

Latex: \left(\frac{\frac17 + \frac18 + 2}{\frac13 + \frac12}\right)^2 \pi

Anda perlu menuliskannya dalam bentuk baris (line form).

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

```
  23.2671801626
```

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

Hasil dari perhitungan ini berupa bilangan pecahan desimal (floating point number). Secara bawaan, hasil ditampilkan dengan ketelitian sekitar 12 digit. Pada baris perintah berikut, kita juga akan mempelajari bagaimana cara merujuk pada hasil sebelumnya di dalam baris yang sama.

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

```
  0.47619047619
  10/21
```

Sebuah perintah Euler dapat berupa ekspresi atau perintah dasar (primitive command). Sebuah ekspresi tersusun dari operator dan fungsi. Jika diperlukan, ekspresi harus menggunakan tanda kurung untuk memastikan urutan eksekusi yang benar. Jika ragu, menambahkan tanda kurung adalah ide yang baik. Perlu dicatat bahwa EMT menampilkan tanda kurung buka dan kurung tutup saat Anda mengedit baris perintah.

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

```
  14.4978445072
```

Operator numerik dalam Euler mencakup:
+ operator unary atau operator tambah
- operator unary atau operator kurang
*, /
. perkalian matriks
a
^b perpangkatan untuk a positif atau b bilangan bulat (bisa juga ditulis a**b)
n! operator faktorial
dan masih banyak lagi.

Berikut beberapa fungsi yang mungkin Anda perlukan (masih banyak fungsi lainnya):
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 nama lain (alias), misalnya ln untuk log.

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

```
  2
  0.5
```

```
>sin(30°)
```

```
  0.5
```

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

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

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

Bilangan Real

Tipe data utama dalam Euler adalah bilangan real. Bilangan real direpresentasikan dalam format IEEE dengan ketelitian 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
```

Strings

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

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

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

String dapat digabungkan dengan tanda | atau dengan tanda +. Hal ini juga berlaku untuk angka, yang dalam kasus tersebut akan dikonversi menjadi string.

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

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

Fungsi print juga mengonversi sebuah angka menjadi string. Fungsi ini dapat menerima jumlah digit dan jumlah tempat desimal (0 untuk keluaran rapat), serta secara opsional satuan.

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

```
  Golden Ratio : 1.61803
```

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

```
>none
```

Untuk mengonversi sebuah string menjadi angka, cukup evaluasi string tersebut. Cara ini juga berlaku untuk ekspresi (lihat di bawah).

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

```
  1234.5
```

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

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

```
  affe
  charlie
  bravo
```

Vektor string kosong dinyatakan 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 ini menggunakan kode UTF-8. Untuk membuat 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 Î±, Î², dan sebagainya juga dapat digunakan. Ini bisa menjadi alternatif cepat untuk LaTeX. (Lebih detail tentang komentar ada di bagian berikut).

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

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

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

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

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

```
  Ü is a German letter
```

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

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

```
  We have =.
```

Juga dimungkinkan untuk menggunakan entitas numerik.

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

```
  Ähnliches
```

Nilai Boolean

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

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

```
  0
  1
```

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

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

```
  1
```

Operator boolean mengikuti aturan bahasa matriks.

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

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

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

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

```
  [2,  3,  5,  7,  9,  11,  13,  15,  17,  19,  21,  23,  25,  27,  29,
  31,  33,  35,  37,  39,  41,  43,  45,  47,  49,  51,  53,  55,  57,
  59,  61,  63,  65,  67,  69,  71,  73,  75,  77,  79,  81,  83,  85,
  87,  89,  91,  93,  95,  97,  99]
```

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

```
  [2,  3,  5,  7,  11,  13,  17,  19,  23,  29,  31,  37,  41,  43,  47,
  53,  59,  61,  67,  71,  73,  79,  83,  89,  97]
```

Format output

Format output default dari EMT mencetak 12 digit. Untuk memastikan kita melihat format default, kita mengatur ulang formatnya.

```
>defformat; pi
```

```
  3.14159265359
```

Secara internal, EMT menggunakan standar IEEE untuk angka bertipe double dengan sekitar 16 digit desimal. Untuk melihat seluruh jumlah digit, gunakan perintah longestformat, atau kita dapat menggunakan operator longest untuk menampilkan hasil dalam format terpanjang.

```
>longest pi
```

```
        3.141592653589793
```

Berikut adalah representasi heksadesimal internal dari angka bertipe double.

```
>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 default adalah format(12).

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

```
  0.333333333333
```

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

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

```
    0.53    0.5   0.17   0.26   0.87   0.54   0.49    0.6
    0.66   0.97   0.19   0.94  0.073   0.99   0.01   0.36
    0.52   0.43   0.17   0.18   0.29   0.75   0.47   0.32
```

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

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

```
  3.1416
```

Fungsi longestformat juga mengatur format tampilan skalar.

```
>longestformat; pi
```

```
  3.141592653589793
```

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

hortestformat, shortformat, longformat, longestformat
ormat(length, digits), goodformat(length)
racformat(length)
efformat

Akurasi internal EMT sekitar 16 angka desimal, sesuai standar IEEE. Angka disimpan dalam format internal ini.

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

```
>longestformat; pi,
```

```
  3.141592653589793
```

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

```
    3.14159
```

Format default adalah defformat().

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

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

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

```
        4.934802200544679
```

Terdapat juga operator singkat untuk mencetak hasil dalam format pecahan. Kita sudah menggunakannya sebelumnya.

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

```
  25/12
```

Karena format internal menggunakan cara biner untuk menyimpan angka, nilai 0.1 tidak dapat direpresentasikan secara tepat. Kesalahan ini akan bertambah sedikit, seperti yang terlihat pada perhitungan berikut.

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

```
   -1.110223024625157e-16
```

Namun dengan format default longformat, hal ini tidak akan terlihat. Untuk kenyamanan, output dari angka yang sangat kecil ditampilkan sebagai 0.

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

```
  0
```

Ekspresi

String atau nama dapat digunakan untuk menyimpan ekspresi matematika, yang dapat dievaluasi oleh EMT. Untuk ini, gunakan tanda kurung setelah ekspresi. Jika Anda ingin menggunakan string sebagai ekspresi, gunakan konvensi penamaan seperti fx atau fxy, dan sebagainya. Ekspresi memiliki prioritas lebih tinggi dibandingkan fungsi.

Variabel global dapat digunakan dalam proses evaluasi.

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

```
        12.56637061435917
```

Parameter diberikan ke x, y, dan z secara berurutan. Parameter tambahan dapat ditambahkan menggunakan parameter yang telah ditetapkan.

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

```
  -0.919535764538
```

Perlu diperhatikan bahwa ekspresi akan selalu menggunakan variabel global, meskipun terdapat variabel dengan nama yang sama di dalam fungsi. (Jika tidak, evaluasi ekspresi di dalam fungsi bisa menghasilkan hasil yang sangat membingungkan bagi pengguna yang memanggil fungsi tersebut.)

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

```
  36
```

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

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

```
  45
```

Sebagai referensi, kami mencatat bahwa call collections (dibahas di bagian lain) dapat berisi ekspresi. Jadi, kita dapat membuat contoh di atas sebagai berikut.

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

```
  45
```

Ekspresi dalam x sering digunakan seperti fungsi.

Perlu diperhatikan bahwa mendefinisikan fungsi dengan nama yang sama seperti ekspresi simbolik global akan menghapus variabel tersebut untuk menghindari kebingungan antara ekspresi simbolik dan fungsi.

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

```
  12
```

Sebagai konvensi, ekspresi simbolik atau numerik sebaiknya dinamai fx, fxy, dan sebagainya. Skema penamaan ini tidak boleh digunakan untuk fungsi.

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

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

Bentuk khusus dari sebuah ekspresi memungkinkan penggunaan variabel apa pun sebagai parameter tak bernama dalam evaluasi ekspresi, tidak hanya "x", "y", dan sebagainya. Untuk ini, mulailah ekspresi dengan "@(variables) ...".

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

```
  @(a,b) a^2+b^2
  41
```

Ini memungkinkan manipulasi ekspresi dalam variabel lain untuk fungsi-fungsi EMT yang memerlukan ekspresi dalam "x".

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

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

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

```
  -0.475
```

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

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

```
  -0.425
```

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

Matematika Simbolik

EMT melakukan matematika simbolik dengan bantuan Maxima. Untuk detail, mulailah dengan tutorial berikut, atau jelajahi referensi Maxima. Ahli Maxima perlu mencatat bahwa terdapat perbedaan sintaks antara sintaks asli Maxima dan sintaks default ekspresi simbolik di EMT.

Matematika simbolik terintegrasi secara mulus ke dalam Euler menggunakan &. Setiap ekspresi yang diawali dengan & adalah ekspresi simbolik. Ekspresi tersebut dievaluasi dan dicetak oleh Maxima.
Pertama-tama, Maxima memiliki aritmatika "tak terbatas" yang dapat menangani angka sangat besar.

```
>$&44!
```

$$26582715747884487680436258110146158903196385280000000000$$

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

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

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

$$2481256778$$

Tentu saja, Maxima memiliki fungsi yang lebih efisien untuk ini (demikian pula bagian numerik dari EMT).

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

$$2481256778$$

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

Anda juga akan mengetahui bahwa hal 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)
```

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

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

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

$$120$$

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

kspresi simbolik dicetak oleh Maxima dalam bentuk 2D. Hal ini disebabkan oleh adanya flag simbolik khusus dalam string.

Seperti yang akan Anda lihat pada contoh sebelumnya dan berikutnya, jika LaTeX telah terpasang, Anda dapat mencetak ekspresi simbolik menggunakan LaTeX. Jika tidak, perintah berikut akan menghasilkan pesan kesalahan.

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

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

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

Ekspresi simbolik diparsing oleh Euler. Jika Anda memerlukan sintaks kompleks dalam satu ekspresi, Anda dapat menempatkan ekspresi tersebut di dalam "...". Menggunakan lebih dari satu ekspresi sederhana dimungkinkan, tetapi sangat tidak disarankan.

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

25

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

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

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

Sekali lagi, % merujuk pada hasil sebelumnya.

Untuk mempermudah, kita menyimpan solusi ke dalam variabel simbolik. Variabel simbolik didefinisikan dengan "&=".

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

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

Ekspresi simbolik dapat digunakan dalam ekspresi simbolik lainnya.

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

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

Input langsung perintah Maxima juga tersedia. Mulailah baris perintah dengan ::. Sintaks Maxima disesuaikan dengan sintaks EMT (disebut "mode kompatibilitas").

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

2432902008176640000

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

```
  8  4  2
2  3  5  7
```

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

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

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

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

```
  2
g
```

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

```
  3   x
x   E
```

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

Variabel semacam ini dapat digunakan dalam ekspresi simbolik lainnya. Perlu diperhatikan, bahwa pada perintah berikut sisi kanan dari &= dievaluasi terlebih dahulu sebelum dilakukan penugasan ke Fx.

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

```
     5
125 E
```

$$125\, e^5$$

```
18551.64488782208
```

```
>fx(5)
```

```
  18551.6448878
```

Untuk evaluasi sebuah ekspresi dengan nilai tertentu dari variabel, Anda dapat menggunakan operator "with". Perintah berikut juga menunjukkan bahwa Maxima dapat mengevaluasi ekspresi secara numerik menggunakan float().

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

$$1000\ E_{10} - 125\ E_5$$

$$2.20079141499189e+7$$

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

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

Untuk mendapatkan kode LaTeX dari sebuah 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
```

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

Penugasan juga dapat bersifat simbolik.

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

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

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

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

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

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

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

```
  1.56155281281
```

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

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

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

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

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

Untuk mendapatkan solusi simbolik tertentu, Anda dapat menggunakan "with" dan sebuah indeks.

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

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

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

Untuk menyelesaikan sistem persamaan, gunakan vektor persamaan. Hasilnya adalah vektor solusi.

```
>sol &= solve([x+y=3,x^2+y^2=5],[x,y]); $&sol, $&x*y with sol[1]
```

$$2$$

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

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

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

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

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

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

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

Fungsi

Di EMT, fungsi adalah program yang didefinisikan dengan perintah "function". Fungsi ini dapat berupa fungsi satu baris atau fungsi multi-baris.
ungsi satu baris dapat bersifat numerik atau simbolik. Fungsi satu baris numerik didefinisikan dengan ":=".

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

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

```
>f(2)
```

```
  4.472135955
```

Fungsi ini juga akan bekerja untuk vektor, mengikuti bahasa matriks Euler, karena ekspresi yang digunakan dalam fungsi sudah 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 dipetakan. Alih-alih menggunakan ekspresi, kita hanya perlu menyediakan nama fungsi. Berbeda dengan ekspresi simbolik atau numerik, nama fungsi harus diberikan dalam bentuk string.

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

```
  0.786151377757
```

Secara default, jika Anda perlu menimpa fungsi bawaan, Anda harus menambahkan kata kunci "overwrite". Menimpa fungsi bawaan berisiko dan dapat menyebabkan masalah bagi fungsi lain yang bergantung padanya. Anda masih dapat memanggil fungsi bawaan tersebut sebagai "_...", jika fungsi tersebut ada di inti Euler.

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

```
  0.707106781187
```

Sebaiknya kita hapus pendefinisian ulang fungsi 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
```

Mengabaikan parameter ini akan menggunakan nilai default.

```
>f(4)
```

```
  16
```

Mengatur parameter ini akan menimpa nilai default.

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

```
  80
```

Parameter yang ditetapkan juga menimpa nilai default. Hal ini digunakan oleh banyak fungsi Euler seperti plot2d, plot3d.

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

```
  16
```

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

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

```
  24
```

Namun, parameter yang ditetapkan akan menimpa nilai global.

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

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

```
  20
```

Fungsi simbolik didefinisikan dengan &=. Fungsi ini didefinisikan di Euler dan Maxima, dan bekerja di kedua lingkungan. Ekspresi yang mendefinisikan fungsi dijalankan melalui Maxima sebelum definisi.

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

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

Fungsi simbolik dapat digunakan dalam ekspresi simbolik.

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

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

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

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

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

```
  178.635099908
```

Fungsi ini dapat digunakan untuk mendefinisikan fungsi atau ekspresi simbolik lainnya.

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

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

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

```
  0.703467422498
```

Hal berikut ini juga berhasil, 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(%)
```

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

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

```
  625
```

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

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

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

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

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

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

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

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

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

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

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

$$x^3-x$$

Dengan &= fungsi bersifat simbolik, dan dapat digunakan dalam ekspresi simbolik lainnya.

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

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

Dengan := fungsi bersifat numerik. Contoh yang baik adalah integral tentu seperti
Latex: f(x) = \int_1^x t^t , dt,
yang tidak dapat dievaluasi secara simbolik.
ika kita mendefinisikan ulang fungsi dengan kata kunci "map", fungsi tersebut dapat digunakan untuk vektor x. Secara internal, fungsi dipanggil untuk semua nilai x sekaligus, dan hasilnya disimpan dalam sebuah 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 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, kita ingin menggunakan fungsi untuk vektor di satu tempat, dan untuk elemen individu di tempat lain. Hal ini memungkinkan dengan parameter vektor.

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

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

Fungsi simbolik semacam ini dapat digunakan untuk variabel simbolik.
Namun, fungsi ini juga dapat digunakan untuk vektor numerik.

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

```
  17
```

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

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

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

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

$$0$$

Namun tentu saja, fungsi tersebut dapat digunakan dalam ekspresi simbolik atau dalam pendefinisian fungsi simbolik.

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

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

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

Menyelesaikan Ekspresi

Ekspresi dapat diselesaikan secara numerik maupun simbolik.

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

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

```
  1.41421356237
```

Ini juga bekerja untuk ekspresi simbolik. Perhatikan fungsi berikut.

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

```
  Syntax error in expression, or unfinished expression!
  Error in:
  \$&solve(x^2=2,x) ...
  ^
```

```
>$&solve(x^2-2,x)
```

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

```
>$&solve(a*x^2+b*x+c=0,x)
```

$$\left[x=\frac{-\sqrt{b^2-4\,a\,c}-b}{2\,a},x=\frac{\sqrt{b^2-4\,a\,c}-b}{2\,a}\right]$$

```
>$&solve([a*x+b*y=c,d*x+e*y=f],[x,y])
```

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

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

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

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

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

```
  0.966715594851
  2
```

Menyelesaikan ekspresi simbolik dalam bentuk simbolik menghasilkan daftar solusi. Kita menggunakan symbolic solver 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 mengevaluasi solusi secara numerik seperti ekspresi biasa.

```
>longest sol()
```

```
      -0.6180339887498949       1.618033988749895
```

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

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

$$0$$

Menyelesaikan sistem persamaan secara simbolik dapat dilakukan dengan vektor persamaan dan symbolic solver 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.

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

dengan a=3.

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

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

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

```
  2.54116291558
```

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

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

```
  2.54116291558
```

Menyelesaikan Pertidaksamaan

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

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

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

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

$$[1 < x] \vee [x < -1]$$

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

$$[-1 < x, x < 1]$$

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

$$[-1 < x, x < 1] \vee [1 < x] \vee [x < -1]$$

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

$$[x < 6] \vee [6 < x]$$

```
>$&fourier_elim([x < 1, x > 1],[x]) // tidak memiliki penyelesaian
```

$$\mathit{emptyset}$$

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

$$\mathit{universalset}$$

```
>$&fourier_elim([x^3 - 1 > 0],[x])
```

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

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

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

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

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

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

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

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

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

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

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

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

```
        [6 < x, x < 8, y < - 11] or [8 < x, y < - 11]
  or [x < 8, 13 < y] or [x = y, 13 < y] or [8 < x, x < y, 13 < y]
  or [y < x, 13 < y]
```

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

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

Bahasa Matriks

Dokumentasi inti EMT memuat pembahasan mendetail tentang bahasa matriks Euler.
Vektor dan matriks dimasukkan menggunakan tanda kurung siku, elemen dipisahkan dengan koma, dan baris dipisahkan dengan titik koma.

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

```
               1               2
               3               4
```

Perkalian matriks ditandai dengan titik (dot).

```
>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
```

Inti 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 perkalian 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, ia akan diperluas secara alami.

```
>2*A
```

```
                2               4
                6               8
```

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

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

```
                1               4
                3               8
```

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

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

```
                2               6
                6              12
```

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

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

```
                1               2
                1               2
```

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

```
                1               4
                3               8
```

Hal ini juga berlaku untuk dua vektor di mana salah satunya merupakan vektor baris dan yang lainnya vektor kolom. Kita menghitung i*j untuk i,j dari 1 sampai 5. Triknya adalah mengalikan 1:5 dengan transposenya. Bahasa matriks di 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
```

Ingat, ini bukan 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]
```

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

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

```
  5
```

Euler memiliki operator perbandingan, seperti "==", yang memeriksa kesetaraan.
Kita akan mendapatkan sebuah vektor berisi 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 ini, fungsi nonzeros memilih elemen yang tidak nol.
Dalam kasus ini, kita akan mendapatkan indeks semua elemen yang lebih besar dari 50.

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

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

Tentu saja, kita bisa menggunakan vektor indeks ini untuk mengambil nilai yang sesuai dalam t.

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

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

Sebagai contoh, mari kita temukan semua kuadrat dari angka 1 hingga 1000, yang memiliki sisa 5 jika dibagi 11 dan sisa 3 jika dibagi 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, EMT menggunakan floating point presisi ganda (double precision). Namun, seringkali EMT tetap sangat berguna.

Kita bisa memeriksa bilangan prima. Misalnya, mari kita cari berapa banyak kuadrat ditambah 1 yang merupakan bilangan prima.

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

```
  112
```

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

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

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

Fungsi ini mengembalikan indeks dari elemen-elemen yang tidak 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 mengubah elemen-elemen tersebut menjadi suatu 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 digunakan untuk mengatur elemen pada indeks tertentu menjadi nilai 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
```

Selain itu, dimungkinkan untuk mengambil elemen-elemen tersebut ke dalam sebuah vektor.

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

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

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

```
>ex=extrema(A)
```

```
      0.267829             4      0.765761             1
       0.13673             1      0.952814             4
      0.006085             2      0.548138             1
```

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

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

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

Fungsi Matriks Lainnya (Membangun Matriks)

Untuk membangun sebuah matriks, kita dapat menumpuk satu matriks di atas matriks lain. 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 menempelkan sebuah matriks di samping matriks lain, jika keduanya memiliki jumlah baris yang sama.

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

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

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

Ada pengecualian untuk aturan ini. Sebuah bilangan real yang ditempelkan 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 juga 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 ini adalah untuk menafsirkan sebuah vektor ekspresi sebagai vektor kolom.

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

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

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

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

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

Untuk vektor, ada fungsi length().

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

```
  9
```

Masih banyak fungsi lain yang dapat digunakan untuk menghasilkan matriks.

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

```
                1               1
                1               1
```

Fungsi-fungsi ini juga dapat digunakan dengan satu parameter. Untuk mendapatkan vektor dengan angka lain selain 1, gunakan cara berikut.

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

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

Selain itu, matriks berisi angka acak juga dapat dibuat dengan random (distribusi uniform) atau normal (distribusi Gauss).

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

```
        0.66566       0.831835
          0.977       0.544258
```

Berikut adalah fungsi lain yang berguna, yang dapat menyusun ulang elemen-elemen sebuah matriks menjadi matriks lain.

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

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

Dengan fungsi berikut, kita dapat menggunakan ini bersama fungsi dup untuk menulis fungsi rep(), yang mengulangi sebuah 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-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 rotleft() dan rotright().

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

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

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

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

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

Perlu dicatat bahwa vektor i pada drop(v,i) merujuk pada indeks elemen dalam v, bukan nilai dari elemen tersebut. Jika ingin menghapus elemen, Anda perlu menemukan elemen tersebut terlebih dahulu. Fungsi indexof(v,x) dapat digunakan untuk menemukan elemen x dalam vektor v yang sudah terurut.

```
>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 masalah jika memasukkan indeks di luar jangkauan (misalnya 0), indeks ganda, atau indeks yang tidak berurutan.

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

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

Ada beberapa fungsi khusus untuk mengatur diagonal atau menghasilkan matriks diagonal.
Kita mulai dengan matriks identitas.

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

```
1  0  0  0  0
0  1  0  0  0
0  0  1  0  0
0  0  0  1  0
0  0  0  0  1
```

Kemudian kita 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
```

Perlu dicatat bahwa kita tidak mengubah matriks A. Kita mendapatkan matriks baru sebagai hasil dari setdiag().
Berikut adalah sebuah 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 sebuah matriks juga dapat diekstrak dari matriks tersebut.
Untuk mendemonstrasikan ini, kita menyusun ulang vektor 1:9 menjadi matriks 3x3.

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

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

Sekarang kita dapat mengekstrak diagonalnya.

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

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

Misalnya, kita bisa membagi matriks dengan diagonalnya.
Bahasa matriks secara otomatis menangani bahwa vektor kolom d diterapkan ke matriks baris demi baris.

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

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

Vektorisasi

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

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

```
  [1,  1.41421,  1.73205]
```

Dengan demikian, Anda bisa dengan mudah membuat tabel nilai. Ini salah satu cara untuk memplot sebuah 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.
ada contoh berikut, kita membuat vektor nilai t[i] dengan jarak 0.1 dari -1 sampai 1. Kemudian kita menghasilkan vektor nilai dari fungsi tersebut.

$$s = t^3 - t$$

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

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

EMT memperluas operator untuk skalar, vektor, dan matriks dengan cara yang jelas dan alami.
Misalnya, sebuah vektor kolom dikalikan dengan vektor baris akan diperluas menjadi matriks, jika operator diterapkan. Pada 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
```

Perlu dicatat, ini sangat berbeda dari perkalian matriks. Perkalian matriks ditandai dengan titik "." di EMT.

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

```
  55
```

Secara default, vektor baris dicetak dalam format ringkas.

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

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

Untuk matriks, operator khusus . menunjukkan perkalian matriks, dan A' menunjukkan transpos. Matriks 1x1 dapat digunakan sama seperti bilangan nyata.

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

```
  5
  25
```

Untuk mentranspos matriks, kita menggunakan tanda petik (').

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

```
                1
                2
                3
                4
```

Dengan demikian, kita bisa menghitung matriks A dikali vektor b.

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

```
               30
               70
```

Perlu dicatat bahwa v tetap merupakan vektor baris. Jadi v'.v berbeda dari v.v'.

```
>v'.v
```

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

v.v' menghitung norma kuadrat dari v untuk vektor baris v. Hasilnya adalah vektor 1x1, yang dapat digunakan sama seperti bilangan nyata.

```
>v.v'
```

```
  30
```

Ada juga fungsi norm (beserta banyak fungsi lain dari Aljabar Linear).

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

```
  30
```

Operator dan fungsi mengikuti bahasa matriks Euler.

Berikut ringkasan aturannya:
-Sebuah fungsi yang diterapkan pada vektor atau matriks diterapkan ke setiap elemennya.
-Sebuah operator yang diterapkan pada dua matriks dengan ukuran sama diterapkan secara pasangan (pairwise) pada elemen-elemen matriks tersebut.
-Jika kedua matriks memiliki dimensi berbeda, keduanya diperluas dengan cara yang masuk akal sehingga memiliki ukuran yang sama.

Misalnya, sebuah skalar dikalikan dengan vektor akan mengalikan skalar tersebut dengan setiap elemen vektor. Atau matriks dikalikan dengan vektor (menggunakan *, bukan .) akan memperluas vektor sesuai ukuran matriks dengan menduplikasi vektor tersebut.

Berikut adalah contoh sederhana dengan operator ^.

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

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

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

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

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

Perlu dicatat bahwa perkalian skalar menggunakan perkalian matriks, bukan *!

```
>v.v'
```

```
  14
```

Ada banyak fungsi untuk matriks. Berikut adalah daftar singkat. Anda sebaiknya merujuk ke dokumentasi untuk informasi lebih lengkap tentang perintah-perintah ini.

-sum, prod menghitung jumlah dan hasil kali dari baris-baris matriks
-cumsum, cumprod melakukan hal yang sama secara kumulatif
-Menghitung nilai ekstremal dari setiap baris
-extrema mengembalikan vektor dengan informasi ekstremal
-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) nilai eigen

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

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

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

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

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

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

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

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

Elemen-elemen dari sebuah matriks diacu 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 seluruh baris ke-i dari matriks.

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

```
  6
  [7,  8,  9]
```

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

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

```
  [2,  4]
             2
             5
             8
```

Bentuk singkat dari : adalah dengan menghilangkan indeks sama sekali.

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

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

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

```
>A{4}
```

```
  4
```

Sebuah matriks juga dapat "diperlebar" (flattened) 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 sebagai tabel, mari kita kembalikan ke format default, dan menghitung tabel nilai sinus dan cosinus. Perlu dicatat bahwa sudut secara default dalam radian.

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

```
              0
             45
             90
            135
            180
            225
            270
            315
            360
```

Sekarang kita menambahkan kolom ke sebuah 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 bahasa matriks, kita dapat menghasilkan beberapa tabel dari beberapa fungsi sekaligus.

Pada contoh berikut, kita menghitung t[j]^i untuk i dari 1 hingga n. Kita mendapatkan sebuah matriks, di mana setiap baris adalah tabel t^i untuk satu i.
Artinya, matriks memiliki elemen

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

Fungsi yang tidak bekerja untuk input vektor harus "divektorisasi".
ni dapat dicapai dengan kata kunci map dalam definisi fungsi. Kemudian fungsi tersebut akan dievaluasi untuk setiap elemen dari parameter vektor.

Integrasi numerik integrate() hanya bekerja untuk batas interval skalar.
Jadi kita perlu melakukan vektorisasi.

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

Kata kunci "map" melakukan vektorisasi pada fungsi. 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 tanda kurung.

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

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

Kita dapat mengakses seluruh baris dari sebuah matriks.

```
>A[2]
```

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

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

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

```
  2
```

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

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

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

Jika indeks berupa vektor indeks, Euler akan mengembalikan baris-baris yang sesuai dari matriks.
Di 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 A yang diurutkan ulang.

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

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

Trik indeks ini juga bekerja untuk kolom.

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

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

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

Sebagai singkatan, : menunjukkan semua indeks baris atau kolom.

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

```
                3
                6
                9
```

Alternatifnya, biarkan indeks pertama kosong.

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

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

Kita juga dapat mengambil baris terakhir dari A.

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

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

Sekarang mari kita ubah elemen-elemen A dengan menetapkan sebuah sub-matriks dari A ke suatu nilai. Ini memang akan mengubah matriks A yang tersimpan.

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

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

Kita juga dapat menetapkan sebuah nilai ke satu baris dari A.

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

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

Kita bahkan dapat menetapkan nilai ke sebuah 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 pintasan diperbolehkan.

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

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

Peringatan: Indeks di luar jangkauan akan menghasilkan matriks kosong atau pesan kesalahan, tergantung pada pengaturan sistem. Secara default, ini menghasilkan pesan kesalahan. Namun, perlu diingat bahwa indeks negatif dapat digunakan untuk mengakses elemen matriks mulai dari akhir.

```
>A[4]
```

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

SPengurutan dan Acak

Fungsi sort() mengurutkan sebuah vektor baris.

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

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

Seringkali diperlukan untuk mengetahui indeks dari vektor yang sudah diurutkan dalam vektor asli. Ini dapat digunakan untuk mengurutkan 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 tepat dari v.

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

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

Ini juga bekerja untuk vektor string.

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

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

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

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

Seperti yang Anda lihat, posisi entri ganda agak acak.

```
>ind
```

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

Fungsi unique mengembalikan daftar elemen unik dari sebuah vektor yang sudah diurutkan.

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

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

Ini juga bekerja untuk vektor string.

```
>unique(s)
```

```
  a
  aa
  d
  e
```

Aljabar Linear

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

Untuk sistem linear Ax = b, Anda dapat menggunakan algoritma Gauss, matriks invers, atau linear fit. 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 membuat matriks 200x200 dan menghitung jumlah dari setiap barisnya. Kemudian kita menyelesaikan Ax = b menggunakan matriks invers. Kita mengukur kesalahan sebagai deviasi maksimum dari semua elemen terhadap 1, yang tentu saja adalah solusi yang benar.

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

```
    8.790745908981989e-13
```

Jika sistem tidak memiliki solusi, linear fit meminimalkan norma dari 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 linear yang sederhana.
ita dapat mendefinisikan matriks untuk Euler dan Maxima dengan &:=, dan kemudian menggunakannya dalam ekspresi simbolik.

Bentuk umum [...] untuk mendefinisikan matriks dapat digunakan di Euler untuk mendefinisikan matriks simbolik.

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

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

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

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

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

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

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

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

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

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

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

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

Nilai eigen juga dapat dihitung secara otomatis. Hasilnya adalah vektor dengan dua vektor, yaitu nilai eigen dan multiplikitasnya.

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

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

Untuk mengekstrak vektor eigen tertentu, diperlukan pengindeksan yang hati-hati.

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

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

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

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

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

```
                1                4
                5                2
```

Dalam ekspresi simbolik, gunakan with.

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

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

Akses ke baris matriks simbolik bekerja sama seperti pada matriks numerik

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

$$[1, a]$$

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

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

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

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

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

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

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

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

$$\left(\begin{smallmatrix}-2 & 1\\ \frac{3}{2} & -\frac{1}{2}\end{smallmatrix}\right)$$

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

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

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

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

Perlu dicatat, 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 rincian lebih lanjut.

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

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

Nilai Numerik dalam Ekspresi Simbolik

Sebuah ekspresi simbolik hanyalah sebuah string yang berisi ekspresi.
ika kita ingin mendefinisikan sebuah nilai untuk ekspresi simbolik sekaligus untuk ekspresi numerik, kita harus menggunakan &:=.

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

```
                1        3.14159
                4              5
```

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

```
>$&A
```

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

Untuk menghindari hal ini, ada fungsi mxmset(variable).

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

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

Maxima juga dapat menghitung dengan bilangan floating point, bahkan dengan bilangan floating point besar hingga 32 digit. Namun, evaluasinya jauh lebih lambat.

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

$$1.414213562373095$$

Presisi dari bilangan floating point besar dapat diubah.

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

```
          3.14159265358979323846264338327950288419716939937510582097494\
 459230781640628620899862803482534211706 8b0
```

Sebuah variabel numerik dapat digunakan dalam ekspresi simbolik dengan menggunakan @var.
Perlu dicatat bahwa ini hanya diperlukan jika variabel tersebut telah didefinisikan dengan := atau = sebagai variabel numerik.

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

$$-5.424777960769379$$

Demo - Suku Bunga

Di bawah ini, kita menggunakan Euler Math Toolbox (EMT) untuk perhitungan suku bunga.
ita melakukannya secara numerik dan simbolik untuk menunjukkan bagaimana Euler dapat digunakan untuk menyelesaikan masalah kehidupan nyata.

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

```
>K=5000
```

```
  5000
```

Sekarang kita anggap suku bunga sebesar 3% per tahun.
ari kita tambahkan satu tingkat bunga sederhana dan hitung hasilnya.

```
>K*1.03
```

```
  5150
```

Euler juga akan memahami sintaks berikut ini.

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

```
  5150
```

Namun, lebih mudah menggunakan faktor tersebut.

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

```
  1.03
  5150
```

Untuk 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 hasil tersebut dibulatkan menjadi 2 digit dalam satu kalimat lengkap.

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

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

Bagaimana jika kita ingin mengetahui hasil antara dari tahun 1 hingga tahun 9?
Untuk ini, bahasa matriks Euler sangat membantu. Anda tidak perlu menulis loop, 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 sebuah 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 vektor elemen demi elemen. Jadi

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

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

Ini menghasilkan vektor faktor q^0 hingga q^10. Vektor ini dikalikan dengan K, dan kita mendapatkan vektor nilai-nilainya.

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

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

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

```
  Variable Cfunction not found!
  Error in:
  Cfunction oneyear (K) := round(K*q,2) ...
            ^
```

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

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

```
  Function oneyear not found.
  Try list ... to find functions!
  Error in:
  longest oneyear(1234.57), longest 1234.57*q ...
                           ^
```

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

Cara termudah adalah menggunakan fungsi iterate, yang mengulangi sebuah fungsi yang diberikan sejumlah kali.

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

```
  Variable oneyear not found!
  Use global variables or parameters for string evaluation.
  Error in expression: oneyear
  niterate:
      x=f$(x,args());
  Try "trace errors" to inspect local variables after errors.
  iterate:
      return niterate(f$,x,n,till;args());
```

Kita dapat mencetaknya dengan cara yang ramah, menggunakan format kita dengan jumlah desimal tetap.

```
>VKr'
```

```
  Variable VKr not found!
  Error in:
  VKr' ...
      ^
```

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

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

```
  VKr is not a variable!
  Error in:
  VKr[2], VKr[1:3] ...
        ^
```

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

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

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

```
  VKr is not a variable!
  Error in:
  VKr[-1], VK[-1] ...
         ^
```

Perbedaannya sangat kecil.

Menyelesaikan Persamaan

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

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

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

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

```
  Real 1 x 11 matrix

     5000.00     5350.00     5710.50     6081.82      ...
```

Bagaimana jika kita mengurangi jumlah yang sama setiap tahun?

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

```
  Real 1 x 11 matrix

     5000.00     4950.00     4898.50     4845.45      ...
```

Kita melihat bahwa uang berkurang. Jelas, jika kita hanya mendapat 150 dari bunga pada tahun pertama, tetapi menarik 200, kita kehilangan uang setiap tahun.

Bagaimana kita dapat menentukan berapa lama uang akan bertahan?
ita harus menulis sebuah loop untuk ini. Cara termudah adalah mengulangi fungsi tersebut cukup lama.

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

```
  Real 1 x 51 matrix

     5000.00     4950.00     4898.50     4845.45      ...
```

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

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

```
        48.00
```

Alasannya adalah bahwa nonzeros(VKR<0) mengembalikan vektor indeks i, di mana VKR[i]<0, dan min menghitung indeks minimum.

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

Fungsi iterate() memiliki satu trik lagi. Fungsi ini dapat mengambil kondisi akhir sebagai argumen.
emudian 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.
Misalkan kita mengetahui bahwa nilai uang menjadi 0 setelah 50 tahun. Berapakah tingkat bunganya?

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

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

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

Iterasi sama seperti sebelumnya

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

Tetapi sekarang kita tidak lagi menggunakan nilai global dari R dalam ekspresi kita.
Fungsi seperti iterate() memiliki trik khusus di Euler. Anda dapat melewatkan nilai-nilai variabel dalam ekspresi sebagai parameter dengan tanda titik koma. Dalam kasus ini, P dan R.

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

Mari kita coba sebuah pengujian.

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

```
       -19.83
```

Sekarang kita dapat menyelesaikan masalah kita.

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

```
         3.15
```

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

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

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

```
      -336.08
```

Perlu dicatat bahwa Anda tidak dapat menyelesaikan untuk jumlah tahun, karena fungsi kita mengasumsikan n sebagai nilai integer.

Solusi Simbolik untuk Masalah Tingkat Bunga

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

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

$$R + q\,K$$

Sekarang kita dapat melakukan iterasi terhadap fungsi ini.

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

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

Kita melihat sebuah pola. Setelah n periode kita memiliki

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

Rumus tersebut adalah rumus jumlah deret geometri, yang sudah dikenal di Maxima.

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

$$\sum_{k=0}^{n-1} q^k = \frac{q^n - 1}{q - 1}$$

Ini agak rumit. Jumlah tersebut dievaluasi dengan flag "simpsum" untuk menyederhanakannya menjadi bentuk pecahan.
Mari kita buat sebuah fungsi untuk ini.

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

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

Fungsi ini melakukan hal yang sama seperti fungsi f kita sebelumnya. Namun, fungsi ini lebih efektif.

```
>1longest f(5000,-200,3,47), longest fs(5000,-200,3,47)
```

```
      -19.82504734650985
  Wrong argument.
  Cannot use a string here.

  Error in:
  1longest f(5000,-200,3,47), longest fs(5000,-200,3,47) ...
                              ^
```

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

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

```
        20.51
```

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

Kita juga dapat menggunakan bagian simbolik dari Euler untuk menghitung rumus pembayaran.

Misalkan kita mendapatkan pinjaman sebesar K, dan membayar n kali angsuran sebesar R (dimulai setelah tahun pertama) dengan menyisakan sisa hutang sebesar Kn (pada saat pembayaran terakhir). Rumus untuk hal ini jelas adalah

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

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

Biasanya rumus ini diberikan dalam bentuk

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

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

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

Kita dapat menyelesaikan untuk besarnya pembayaran
R secara simbolik.

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

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

Seperti yang terlihat dari rumus, fungsi tersebut menghasilkan error bilangan pecahan (floating point error) saat i=0. Namun, Euler tetap dapat membuat plot-nya.

Tentu saja, kita memiliki limit berikut

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

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

Jelas, tanpa bunga kita harus membayar 10 kali angsuran sebesar 500.
ersamaan juga dapat diselesaikan untuk n.
Hasilnya akan terlihat lebih rapi jika kita melakukan sedikit penyederhanaan padanya.

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

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

Identitas

Nama : Salma Nadia Zulfa
NIM : 24030130005
Kelas : Pendidikan Matematika A 2024

CATATAN BELAJAR

Penjumlahan, Pengurangan, Perkalian Polinomial

URAIAN MATERI
Polinomial adalah bentuk aljabar yang terdiri dari penjumlahan beberapa suku berupa konstanta dikalikan pangkat variabel.
Bantuk umumnya: P(x)=anx^n+an-1x^n-1+a1x+a0
Derajat polinomial: pangkat tertinggi dengan koefisien tak nol.

Operasi dasar:
-Penjumlahan/Pengurangan:gabungkan suku sejenis.
-Perkalian:gunakan distributif atau identitas aljabar.
Faktorisasi: memecah polinomial menjadi perkalian faktor yang lebih sederhana.
Penyederhanaan rasional: membagi faktor sejenis pada pembilang dan penyebut.

Soal 1: Kembangkan (2x-3)(x+5)

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

$$2\,x^2 + 7\,x - 15$$

Polinomial kuadrat hasil kali dua suku.

Soal 2: Sederhanakan (13x^-2x)+(-15x^-8x)

```
>&simplify((13*x^2-2*x)+(-15*x^2-8*x))
```

```
                                2
simplify(- 2 x  - 10 x)
```

Suku sejenis digabung, menghasilkan bentuk sederhana.

Soal 3 : Kembangkan (x+3)(x^2-x+2)

```
>&expand((x + 3)*(x^2 - x + 2))
```

```
  3      2
 x  + 2 x  - x + 6
```

Hasil kali binomial dengan trinomial adalah polinomial kubik.

Soal 4: Faktorkan x^3-8

```
>&factor(x^3 - 8)
```

```
                  2
(x - 2) (x  + 2 x + 4)
```

Bentuk selisih kubus : satu faktor linear (x-2) dan satu faktor kuadrat irreduksibel.

Soal 5: Faktorkan x^2+5x+6

```
>&factor(x^2 + 5*x + 6)
```

```
(x + 2) (x + 3)
```

Bentuk kuadrat sempurna difaktorkan ke dalam dua faktor linear.

Soal 6: Sederhanakan pecahan x^2-4/x-2

```
>&ratsimp((x^2 - 4)/(x - 2))
```

```
x + 2
```

Faktor (x-2) habis, menyisakan (x+2) dengan catatan

```
 tidak sama dengan 2.
```

Soal 7: Kembangkan (x+2)^3

```
>&expand((x + 2)^3)
```

$$x^3 + 6\,x^2 + 12\,x + 8$$

Bentuk binomial pangkat tiga menghasilkan polinomial kubik.

Soal 8: Selesaikan persamaan linear x-5=7

```
>&solve(x - 12)
```

$$[x = 12]$$

Persamaan disusun ulang jadi x - 12 = 0. Solusinya x=2

EMT membantu expand, factor, simplify, dan solve polinomial secara otomatis.
Hasil yang diperoleh sesuai teori dasar aljabar.
Operasi polinomial di EMT sangat cepat dibanding manipulasi manual.

Faktorisasi Polinomial

URAIAN MATERI
Faktorisasi polinomial adalah memecah polinomial menjadi perkalian faktor yang lebih sederhana.
Tujuan: mempermudah penyelesaian persamaan, penyederhanaan pecahan rasional, dan analisis akar.

Bentuk dasar faktorisasi:
-kuadrat sempurna : x^2+2ax+a^2=(x+a)^2
-selisih kuadrat : x^2-a^2=(x-a)(x+a)
-selisih kubus : x^3-a^3=(x-a)(x^2+ax+a^2)
-jumlah kubus : x^3+a^3=(x+a)(x^2-ax+a^2)

Di EMT, gunakan perintah:
&factor(...) untuk memfaktorkan polinomial.
&expand(...) untuk mengembalikan ke bentuk semula (cek hasil faktorisasi).

Soal 1: Faktorkan x^2+5x+6

```
>&factor(x^2 + 5*x + 6)
```

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

polinomial kuadrat difaktorkan menjadi dua faktor linear dengan akar -2 dan -3.

Soal 2: Faktorkan x^2-9

```
>&factor(x^2 - 9)
```

```
(x - 3) (x + 3)
```

bentuk selisih kuadrat.

Soal 3: Faktorkan x^3-8

```
>&factor(x^3 - 8)
```

```
                2
(x - 2) (x  + 2 x + 4)
```

selisih kubus : faktor linear (x-2) dan faktor kuadrat irreduksibel.

Soal 4: Faktorkan x^3+27

```
>&factor(x^3 + 27)
```

```
                2
(x + 3) (x  - 3 x + 9)
```

jumlah kubus : faktor linear (x+3) dan faktor kuadrat.

Soal 5: Faktorkan x^4+2x^2+1

```
>&factor(x^4 + 2*x^2 + 1)
```

```
   2     2
(x  + 1)
```

bentuk kuadrat sempurna dalam variabel x^2.

Soal 6: Faktorkan x^3-x^2-4x+4

```
>&factor(x^3 - x^2 - 4*x + 4)
```

```
(x - 2) (x - 1) (x + 2)
```

URAIAN MATERI
Pecahan aljabar adalah pecahan yang pembilang/penyebutnya berupa polinomial.

Operasi penjumlahan/pengurangan pecahan aljabar dilakukan dengan:
-Menentukan penyebut sekutu (KPK dari penyebut).
-Mengubah pecahan menjadi berpenyebut sekutu.
-Menjumlahkan/ mengurangkan pembilang.
-Menyederhanakan hasil.

Di EMT:
&ratsimp(expr) untuk menyederhanakan pecahan rasional.
&expand(expr) untuk memperluas hasil.
&factor(expr) untuk memfaktorkan kembali.

Soal 1: Sederhanakan x^2-1/x-1

```
>&ratsimp((x^2 - 1)/(x - 1))
```

$$x + 1$$

faktor (x-1) habis, tersisa x+1, dengan syarat

```
 tidak sama dengan 1.
```

Soal 2: Hitung (1/x)+(1/(x+1))

```
>&ratsimp(1/x + 1/(x+1))
```

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

penyebut sekutu x(x+1), pembilang 1*(x+1)+1*x = 2x+1.

Soal 3: Hitung (x/(x^2-1))-(1/(x+1))

```
>&ratsimp(x/(x^2 - 1) - 1/(x+1))
```

$$\frac{1}{x^2 - 1}$$

hasilnya sederhana sekali, penyebut awal (x^2-1) difaktorkan (x-1)(x+1).

Soal 4: Sederhanakan (x^2+2x+1)/(x+1)

```
>&ratsimp((x^2 + 2*x + 1)/(x+1))
```

$$x + 1$$

karena pembilang adalah (x+1)^2, maka hasil pembagian = x+1.

Soal 5: Hitung (2/x)-(3/(x+2))

```
>&ratsimp(2/x - 3/(x+2))
```

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

penyebut sekutu x(x+2), pembilang jadi 2(x+2) - 3x = -x-6.

Soal 6: Hitung ((x+2)/(x^2+3x+2))+(1/(x+1))

```
>&ratsimp((x+2)/(x^2+3*x+2) + 1/(x+1))
```

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

penyebut (x^2+3x+2) difaktorkan (x+1)(x+2). Setelah disederhanakan, hasil akhir = 2/(x+1).

Operasi pecahan aljabar = sama dengan pecahan biasa, hanya pakai polinomial.
EMT dengan &ratsimp otomatis melakukan KPK penyebut, menjumlahkan/mengurangkan pembilang, dan menyederhanakan hasil.
Dengan ini, pengerjaan jadi sangat cepat dibanding manual.

Perkalian dan Pembagian Pecahan Aljabar

URAIAN MATERI
Perkalian pecahan aljabar: kalikan pembilang dengan pembilang, penyebut dengan penyebut, lalu sederhanakan.

Pembagian pecahan aljabar: balik pecahan kedua, lalu ubah menjadi perkalian.

EMT menyediakan:
&ratsimp(expr) untuk menyederhanakan pecahan.
&expand(expr) untuk memperluas hasil kali.
&factor(expr) untuk memfaktorkan kembali.

Soal 1 : Hitung ((x/(x+1)) * ((x+1)/(x+2)))

```
>&ratsimp((x/(x+1)) * ((x+1)/(x+2)))
```

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

faktor (x+1) saling menghilangkan.

Soal 2: Hitung (((x^2-1)/(x^2+2*x+1)) * ((x+1)/(x-1)))

```
>&ratsimp(((x^2-1)/(x^2+2*x+1)) * ((x+1)/(x-1)))
```

$$1$$

(x^2-1)=(x-1)(x+1) dan (x^2+2x+1)=(x+1)^2, semua faktor habis : hasil = 1.

Soal 3: Hitung ((2/x) / (4/(x+1)))

```
>&ratsimp((2/x) / (4/(x+1)))
```

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

pembagian pecahan = kali kebalikan, hasil (2/x)*(x+1)/4.

Soal 4: Hitung (((x+3)/(x^2-9)) * ((x-3)/(x+1)))

```
>&ratsimp(((x+3)/(x^2-9)) * ((x-3)/(x+1)))
```

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

(x^2-9)=(x-3)(x+3), semua faktor habis : sisa 1/(x+1).

Soal 5: Hitung ((x^2/(x^2-4)) / (x/(x-2)))

```
>&ratsimp((x^2/(x^2-4)) / (x/(x-2)))
```

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

(x^2-4)=(x-2)(x+2), setelah disederhanakan : (x+2)/x.

Soal 6: Hitung ((3*x/(x^2-9)) * ((x+3)/(2*x)))

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

$$\frac{3}{2x-6}$$

(x^2-9)=(x-3)(x+3), setelah dibatalkan faktor hasilnya 3/(2(x-3)).

Perkalian pecahan aljabar = kalikan pembilang dan penyebut, lalu sederhanakan.
Pembagian pecahan aljabar = ubah ke perkalian dengan invers.
&ratsimp di EMT menyelesaikan semua langkah otomatis, termasuk faktorisasi dan penyederhanaan.

Persamaan Linear Satu Variabel

URAIAN MATERI
Persamaan linear satu variabel berbentuk umum:
ax+b=0, dengan a tidak sama dengan 0
solusinya: x=-(b/a)

Persamaan linear banyak digunakan sebagai dasar penyelesaian masalah aljabar, sistem persamaan, dan aplikasi sehari-hari.
Di EMT:
&solve(expr) untuk menyelesaikan persamaan linear maupun non-linear.
Bentuk input: tuliskan persamaan dalam bentuk ekspresi yang harus sama dengan nol.

Soal 1: Selesaikan 2x+5=0

```
>&solve(2*x + 5)
```

$$[x = -\frac{5}{2}]$$

akar tunggal linear, =-2.5.

Soal 2: Selesaikan 5x-20=0

```
>&solve(5*x - 20)
```

```
[x = 4]
```

solusi tunggal x=4.

Soal 3: Selesaikan 7x+14=0

```
>&solve(7*x + 14)
```

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

solusi tunggal x=-2

Soal 4: Selesaikan 3x-9=0

```
>&solve(3*x - 9)
```

```
[x = 3]
```

solusi tunggal x=3

Soal 5: Selesaikan 9x+27=0

```
>&solve(9*x + 27)
```

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

solusi tunggal x=-3

Soal 6: Selesaikan 4x-1=0

```
>&solve(4*x - 1)
```

$$\left[x = \frac{1}{4}\right]$$

solusi tunggal x=0,25

Semua persamaan linear satu variabel punya satu solusi tunggal (selama

```
tidak sama dengan 0).
```

EMT dengan &solve langsung memberikan nilai akar tanpa manipulasi manual.
Konsep ini menjadi dasar untuk menyelesaikan persamaan kuadrat maupun sistem persamaan.

Persamaan Kuadrat Satu Variabel

URAIAN MATERI
Persamaan kuadrat berbentuk umum: ax^2+bx+c=0, dengan a tidak sama dengan 0
Rumus Kuadrat: x=-(b+-akar(b^2-4ac)/2a)

Diskriminan D=b^2-4ac
D>0: 2 akar real berbeda.
D=0: 1 akar real kembar.
D<0: tidak punya akar real, solusinya bilangan kompleks.

Di EMT:
&solve(expr) untuk menyelesaikan persamaan kuadrat.
&factor(expr) untuk memfaktorkan polinomial kuadrat.

Selesikan (x^2 - 5*x + 6)=0

```
>&solve(x^2 - 5*x + 6)
```

```
[x = 3, x = 2]
```

dua akar real berbeda

Selesaikan x^2+2x+1=0

```
>&solve(x^2 + 2*x + 1)
```

```
[x = - 1]
```

akar kembar (diskriminan = 0)

Selesaikan x^2-9=0

```
>&solve(x^2 - 9)
```

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

selisih kuadrat dengan dua akar real berbeda.

Selesaikan 2x^2-4x-6=0

```
>&solve(2*x^2 - 4*x - 6)
```

```
[x = 3, x = - 1]
```

dua akar real berbeda.

Selesaikan x^2+4x+8=0

```
>&solve(x^2 + 4*x + 8)
```

```
[x = - 2 I - 2, x = 2 I - 2]
```

diskriminan negatif, menghasilkan dua akar kompleks konjugat.

Faktorkan x^2-7x+12

```
>&factor(x^2 - 7*x + 12)
```

```
(x - 4) (x - 3)
```

faktor linear : akar real x=3 dan x=4.

Persamaan kuadrat bisa punya 2 akar real, 1 akar kembar, atau 2 akar kompleks.
EMT menyelesaikan dengan &solve secara otomatis, tanpa perlu pakai rumus kuadrat manual.
Faktorisasi dengan &factor membantu memverifikasi hasil.

Sistem Persamaan Linear Dua Variabel

URAIAN MATERI
Sistem persamaan linear dua variabel (SPLDV) berbentuk:
a1x+b1y=c1
a2x+b2y=c2

Solusi = pasangan (x,y) yang memenuhi kedua persamaan.
Metode penyelesaian:
Substitusi
Eliminasi
Matriks / determinan (Cramer's Rule)

Di EMT:
&solve([pers1, pers2], [x, y]) ? menyelesaikan SPL dua variabel.

Selesaikan sistem :
x+y=5
x-y=1

```
>&solve([x + y = 5, x - y = 1], [x, y])
```

```
[[x = 3, y = 2]]
```

solusi unik, pasangan (3,2).

Selesaikan sistem :
2x+y=7
x-y=1

```
>&solve([2*x + y = 7, x - y = 1], [x, y])
```

```
      8      5
[[x = -, y = -]]
      3      3
```

Solusi unik (x,y)=(2.67,1.67).

Selesaikan sistem :
3x+2y=12
x-y=0

```
>&solve([3*x + 2*y = 12, x - y = 0], [x, y])
```

```
      12      12
[[x = --, y = --]]
      5       5
```

solusi unik (x,y)=(2.4,2.4)

Selesaikan sistem :
x+2y=10
2x+4y=20

```
>&solve([x + 2*y = 10, 2*x + 4*y = 20], [x, y])
```

```
[[x = 10 - 2 %r2, y = %r2]]
```

kedua persamaan setara : solusi tak hingga banyak.

Selesaikan sistem :
x-y=4
2x-2y=10

```
>&solve([x - y = 4, 2*x - 2*y = 10], [x, y])
```

```
[]
```

sistem tidak konsisten (garis sejajar, tidak berpotongan).

Selesaikan sistem :
4x+3y=24
2x-y=0

```
>&solve([4*x + 3*y = 24, 2*x - y = 0], [x, y])
```

```
      12      24
[[x = --, y = --]]
      5       5
```

SPLDV bisa punya satu solusi (unik), tak hingga solusi, atau tidak ada solusi.
EMT dengan &solve([...],[x,y]) langsung memberikan solusi maupun informasi konsistensi sistem.
Hasil sesuai teori SPLDV dari eliminasi, substitusi, atau aturan Cramer.

Sistem Persamaan Linear Tiga Variabel

URAIAN MATERI
Sistem persamaan linear tiga variabel (SPLTV) berbentuk:
a1x+b1y+c1z=d1
a2x+b2y+c2z=d2
a3x+b3y+c3z=d3

Solusi = tripel (x,y,z) yang memenuhi ketiga persamaan.
Metode penyelesaian: substitusi, eliminasi, determinan (Cramer's Rule), matriks.

Di EMT:
&solve([pers1, pers2, pers3], [x, y, z]) untuk menyelesaikan SPLTV.

Selesaikan sistem :
x+y+z=6
x-y+z=2
2x+y-z=3

```
>&solve([x + y + z = 6, x - y + z = 2, 2*x + y - z = 3], [x, y, z])
```

```
      5             7
[[x = -, y = 2, z = -]]
      3             3
```

Selesaikan sistem :
x+y+z=0
x-y+z=2
x+y-z=4

```
>&solve([x + y + z = 0, x - y + z = 2, x + y - z = 4], [x, y, z])
```

```
[[x = 3, y = - 1, z = - 2]]
```

Selesaikan sistem:
2x+y-z=1
x-y+2z=3
3x+y+z=5

```
>&solve([2*x + y - z = 1, x - y + 2*z = 3, 3*x + y + z = 5], [x, y, z])
```

$$[[x = \frac{4}{5}, y = 1, z = \frac{8}{5}]]$$

Selesaikan sistem :
x+2y+3z=14
2x-y+z=2
3x+y+2z=11

```
>&solve([x + 2*y + 3*z = 14, 2*x - y + z = 2, 3*x + y + 2*z = 11], [x, y, z])
```

$$[[x = \frac{11}{10}, y = \frac{27}{10}, z = \frac{5}{2}]]$$

Selesaikan sistem:
x+y+z=3
2x+3y+z=7
x-y+z=1

```
>&solve([x + y + z = 3, 2*x + 3*y + z = 7, x - y + z = 1], [x, y, z])
```

```
[[x = 2, y = 1, z = 0]]
```

SPLTV biasanya punya solusi unik, tapi bisa juga tidak konsisten atau punya tak hingga solusi.
EMT mempermudah penyelesaian dengan &solve([...],[x,y,z]), tanpa perlu eliminasi manual.
Semua hasil sesuai metode matriks/determinan.

URAIAN MATERI
Sistem persamaan linear n variabel berbentuk:
a11x1+a12x2+...+a1nxn=b1
a21x1+a22x2+...+a2nxn=b2
...
an1x1+an2x2+...+annxn=bn

Diselesaikan dengan berbagai cara:
Eliminasi Gauss / Gauss-Jordan
Aturan Cramer (jika determinan tak nol)
Matriks invers: X=A^-1B.

Di EMT:
&solve([...], [...]) untuk simbolik.
linsolve(A, b) untuk numerik (dengan matriks A dan vektor b).

Sistem 3 variabel:
2x+y+z=5
x-y+z=1
3x+y+2z=8

```
>&solve([2*x + y + z = 5, x - y + z = 1, 3*x + y + 2*z = 8], [x, y, z])
```

```
[[x = 0, y = 2, z = 3]]
```

Sistem 3 variabel dalam bentuk matriks
x+y+z=6
2x-y+z=3
x+2y-z=4

```
>&solve([x + y + z = 6, 2*x - y + z = 3, x + 2*y - z = 4], [x, y, z])
```

```
      11      16      15
[[x = --, y = --, z = --]]
      7       7       7
```

Sistem 4 variabel
x+y+z+w=10
2x-y+z+w=7
x+2y-z+w=8
3x+y+z-w=9

```
>&solve([x + y + z + w = 10, 2*x - y + z + w = 7, x + 2*y - z + w = 8, 3*x + y + z - w = 9
```

```
      25      29      51      61
[[x = --, y = --, z = --, w = --]]
      11      11      22      22
```

Sistem 4 variabel
x-y+z+w=4
2x+y-z+w=5
x+y+z-w=2
3x-y+2z+w=7

```
>&solve([x - y + z + w = 4, 2*x + y - z + w = 5, x + y + z - w = 2, 3*x - y + 2*z + w = 7]
```

$$[[x = 0, y = \frac{7}{2}, z = 3, w = \frac{9}{2}]]$$

Sistem 4 variabel
2x+y+z+w=6
x-y+2z+w=7
x+y-z+w=3
3x+y+z+2w=12

```
>&solve([2*x + y + z + w = 6, x - y + 2*z + w = 7, x + y - z + w = 3, 3*x + y + z + 2*w =
```

$$[[x = 7, y = -5, z = -2, w = -1]]$$

Sistem persamaan linear n variabel bisa diselesaikan dengan eliminasi, Cramer, atau matriks invers.
EMT dengan &solve mampu menangani 2, 3, bahkan 4 variabel atau lebih.
Hasil diperoleh lebih cepat tanpa prosedur manual panjang.

Pertidaksamaan Linear Satu Variabel *

URAIAN MATERI
Pertidaksamaan linear satu variabel berbentuk:
ax+b{<,=,>,=}c,dengan a tidak sama dengan 0

Prinsip dasar:
Jika kedua ruas ditambah/dikurangi bilangan sama MAKA tanda tetap.
Jika kedua ruas dikali/dibagi bilangan positif MAKA tanda tetap.
Jika dikali/dibagi bilangan negatif MAKA tanda dibalik.
Himpunan penyelesaian biasanya dinyatakan dalam interval.
Di EMT:
&solve(expr < 0, x) atau &solve(expr > 0, x) untuk mencari interval solusi.

Selesaikan 3*x + 6 <= 0

```
>&solve(3*x + 6)
```

$$[x = -2]$$

```
>3*(-3) + 6   // untuk x < -2
```

```
  -3
```

```
>3*(0) + 6    // untuk x > -2
```

```
  6
```

Jadi, untuk <=0, solusi adalah x<=-2.

Selesaikan -x+5>=0

```
>&solve(-x + 5)
```

```
                                [x = 5]
```

```
>-4 + 5   // x < 5
```

```
  1
```

```
>-6 + 5   // x > 5
```

```
  -1
```

Jadi solusinya adalah x<=5

Selesaikan 4x-1<7

```
>&solve(4*x - 1 - 7)
```

```
                                [x = 2]
```

```
>4*1 - 1 - 7    // x < 2
```

```
  -4
```

```
>4*3 - 1 - 7    // x > 2
```

```
  4
```

Jadi solusinya adalah x<2

Selesaikan 5x+10>=0

```
>&solve(5*x + 10)
```

[x = - 2]

```
>5*(-3) + 10   // x < -2
```

-5

```
>5*0 + 10      // x > -2
```

10

Jadi solusinya adalah x>=-2

Selesaikan -2x+8>0

```
>&solve(-2*x + 8)
```

[x = 4]

```
>-2*3 + 8    // x < 4
```

2

```
>-2*5 + 8    // x > 4
```

-2

Jadi solusinya adalah x<4

EMT tidak bisa langsung menyelesaikan pertidaksamaan, tapi bisa digunakan dengan dua langkah: cari akar + cek tanda interval.
Solusi berupa interval di garis bilangan real.
Dengan cara ini, semua soal pertidaksamaan linear dapat diselesaikan konsisten di EMT.