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:

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

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

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

Menjabarkan:

```
>$&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

Satu baris perintah Euler terdiri dari satu atau beberapa perintah Euler yang diakhiri dengan titik koma ";" atau koma ",". Titik koma mencegah hasil perintah untuk dicetak. Koma setelah perintah terakhir dapat dihilangkan.

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

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

```
16.7551608191
```

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

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

```
50.2654824574
100.530964915
```

Baris perintah dieksekusi sesuai urutan saat pengguna menekan tombol "return" (Enter). Jadi, Anda akan mendapatkan nilai baru setiap kali mengeksekusi baris kedua.

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

```
0.540302305868
```

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

```
0.857553215846
```

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

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

```
1.41666666667
1.41421568627
1.41421356237
```

Ini juga merupakan cara yang baik untuk membagi perintah panjang menjadi dua atau lebih baris. Anda bisa menekan 'Ctrl+Return' untuk membagi satu baris menjadi dua pada posisi kursor saat ini, atau 'Ctrl+Back' untuk menggabungkan baris.

Untuk melipat semua multi-baris, tekan 'Ctrl+L'. Kemudian, baris-baris berikutnya hanya akan terlihat jika salah satu dari mereka memiliki fokus. Untuk melipat satu multi-baris, mulai baris pertama dengan "%+ ".

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

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

```
81
```

Euler mendukung loop dalam baris perintah, selama mereka sesuai dalam satu baris tunggal atau multi-baris. Dalam program, batasan ini tidak berlaku, tentu saja. Untuk informasi lebih lanjut, silakan lihat pengantar berikut.

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

```
1.5
1.41666666667
1.41421568627
1.41421356237
1.41421356237
```

Tidak masalah menggunakan multi-baris. Pastikan setiap baris diakhiri dengan " ...".

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

```
    1.41421356237
```

Struktur kondisional juga berfungsi.

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

```
    Thought so!
```

Ketika Anda menjalankan sebuah perintah, kursor bisa berada di posisi mana pun pada baris perintah. Anda bisa kembali ke perintah sebelumnya atau melompat ke perintah berikutnya dengan tombol panah. Atau, Anda bisa mengklik bagian komentar di atas perintah untuk menuju ke perintah tersebut.

Saat Anda memindahkan kursor di sepanjang baris, pasangan tanda kurung buka dan tutup akan disorot. Juga, perhatikan baris status. Setelah tanda kurung buka dari fungsi 'sqrt()', baris status akan menampilkan teks bantuan untuk fungsi tersebut. Jalankan perintah dengan menekan tombol 'return' (Enter).

```
>sqrt(sin(10°)/cos(20°))
```

```
    0.429875017772
```

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

Anda bisa mengklik dua kali pada perintah apa pun untuk membuka bantuan untuk perintah tersebut. Cobalah mengklik dua kali perintah `exp` di bawah pada baris perintah.

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

```
    2.5
```

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

Sintaksis Dasar

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

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

Perintah ganda dalam satu baris dipisahkan dengan "," atau ";". Titik koma menekan keluaran perintah. Pada akhir baris perintah, "," dianggap ada jika ";" tidak ada.

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

```
30.65625
```

EMT menggunakan sintaksis pemrograman untuk ekspresi. Untuk memasukkan

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

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

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

```
8.77908249441
```

untuk menghitung ekspresi rumit seperti

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

Anda perlu memasukkannya dalam bentuk baris.

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

```
23.2671801626
```

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

Hasil dari perhitungan ini adalah angka titik mengambang. Secara default, hasilnya dicetak dengan akurasi sekitar 12 digit. Dalam baris perintah berikut, kita juga akan belajar bagaimana merujuk pada hasil sebelumnya dalam baris yang sama.

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

```
0.47619047619
10/21
```

Perintah Euler bisa berupa ekspresi atau perintah primitif. Ekspresi terdiri dari operator dan fungsi. Jika perlu, ekspresi harus mengandung tanda kurung untuk memaksa urutan eksekusi yang benar. Jika ragu, menambahkan tanda kurung adalah ide yang baik. Perhatikan bahwa EMT menampilkan tanda kurung buka dan tutup saat mengedit baris perintah.

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

```
14.4978445072
```

Operator numerik di Euler meliputi:

- + operator unary atau plus
- - operator unary atau minus
- *, / operator perkalian dan pembagian
- . produk matriks
- a^b pangkat untuk `a` positif atau `b` integer (bentuk `a**b` juga berfungsi)
- n! operator faktorial

Dan banyak lagi.

Berikut beberapa fungsi yang mungkin Anda butuhkan. Ada banyak lagi:

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

Beberapa perintah memiliki alias, misalnya ln untuk log.

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

```
    2
    0.5
```

```
>sin(30°)
```

```
    0.5
```

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

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

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

Angka Real

Tipe data utama di Euler adalah angka real. Angka real diwakili dalam format IEEE dengan akurasi sekitar 16 digit desimal.

```
>longest 1/3
```

```
         0.3333333333333333
```

Representasi internal ganda memerlukan 8 byte.

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

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

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

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

String

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

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

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

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

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

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

Fungsi print juga mengonversi angka menjadi string. Fungsi ini dapat menerima jumlah digit, jumlah tempat desimal (0 untuk output padat), dan idealnya sebuah satuan.

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

```
Golden Ratio : 1.61803
```

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

```
>none
```

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

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

```
1234.5
```

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

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

```
affe
charlie
bravo
```

Vektor string kosong 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 mengandung kode UTF-8. Untuk menghasilkan string seperti itu, gunakan u"..." dan salah satu entitas HTML.

String Unicode dapat digabungkan seperti string lainnya.

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

```
α = 45°
```

I

Dalam komentar, entitas yang sama seperti 'α', 'β', dll., dapat digunakan. Ini bisa menjadi alternatif cepat untuk LaTeX. (Detail lebih lanjut tentang komentar akan dijelaskan di bawah).

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

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

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

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

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

```
Ü is a German letter
```

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

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

```
We have α=β.
```

Anda juga dapat 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-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, kita menggunakan kondisi 'isprime(n)'.

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

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

```
>"elephant">"ant"
```

```
1
```

```
> (17:25)<23, nonzeros(%)
```

```
[1,  1,  1,  1,  1,  1,  0,  0,  0]
[1,  2,  3,  4,  5,  6]
```

```
> "river">"sea"
```

```
0
```

```
>N=10|7:10:77
```

```
[10,  7,  17,  27,  37,  47,  57,  67,  77]
```

```
>N[nonzeros(isprime(N))]
```

```
[7,  17,  37,  47,  67]
```

```
>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 bahwa kita melihat format default, kita reset formatnya.

```
>defformat; pi
```

```
3.14159265359
```

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

```
>longest pi
```

```
3.141592653589793
```

Berikut adalah representasi internal dalam format heksadesimal dari angka ganda.

```
>printhex(pi)
```

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

Format output dapat diubah secara permanen dengan perintah format.

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

```
0.33333
3.14159
0.84147
```

Default-nya adalah 'format(12)'.

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

```
0.333333333333
```

Fungsi seperti "shortestformat", "shortformat", dan "longformat" bekerja untuk vektor sebagai berikut:

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

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

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

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

```
    3.1416
```

Fungsi "longestformat" juga mengatur format skalar.

```
>longestformat; pi
```

```
    3.141592653589793
```

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

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

Akurasi internal EMT adalah sekitar 16 tempat desimal, sesuai dengan 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 defaultnya adalah 'defformat()'.

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

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

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

```
          4.934802200544679
```

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

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

```
    25/12
```

Karena format internal menggunakan cara biner untuk menyimpan angka, nilai 0.1 tidak akan diwakili secara tepat. Kesalahan ini akan bertambah sedikit demi sedikit, seperti yang Anda lihat dalam perhitungan berikut.

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

```
     -1.110223024625157e-16
```

Namun, dengan format default "longformat", Anda tidak akan menyadari hal ini. 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 berniat menggunakan string sebagai ekspresi, gunakan konvensi untuk menamainya dengan "fx" atau "fxy", dll. Ekspresi memiliki prioritas lebih tinggi dibandingkan fungsi.

Variabel global dapat digunakan dalam evaluasi.

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

```
          12.56637061435917
```

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

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

```
    -0.919535764538
```

Perlu dicatat bahwa ekspresi akan selalu menggunakan variabel global, bahkan jika ada variabel dalam fungsi dengan nama yang sama. (Jika tidak, evaluasi ekspresi 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" daripada nilai global, Anda perlu menambahkan "at=value".

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

```
    45
```

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

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

```
    45
```

Ekspresi dalam x sering digunakan seperti halnya fungsi. Perhatikan 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 dengan fx, fxy, dll. Skema penamaan ini sebaiknya tidak digunakan untuk fungsi

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

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

Sebuah bentuk khusus dari ekspresi memungkinkan penggunaan variabel apa pun sebagai parameter tanpa nama untuk evaluasi ekspresi, tidak hanya "x", "y", dll. 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 EMT yang memerlukan ekspresi dalam "x".

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

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

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

```
    -0.475
```

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

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

```
-0.425
```

Sebuah ekspresi tidak harus bersifat simbolik. Ini diperlukan jika ekspresi tersebut mengandung fungsi yang hanya dikenal dalam kernel numerik, bukan dalam Maxima.

Matematika Simbolik

EMT melakukan matematika simbolik dengan bantuan Maxima. Untuk detail lebih lanjut, mulailah dengan tutorial berikut atau telusuri referensi untuk Maxima. Para ahli Maxima perlu memperhatikan bahwa terdapat perbedaan dalam sintaksis antara sintaksis asli Maxima dan sintaksis default dari ekspresi simbolik di EMT.

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

Pertama-tama, Maxima memiliki aritmetika "tak hingga" yang dapat menangani angka yang 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 (begitu juga bagian numerik dari EMT).

contoh lain:

$$C(25,5) = \frac{25!}{20! \cdot 5!}$$

```
>$& 25!/(20!*5!)
```

$$53130$$

```
>$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 yang disediakan oleh pengembang program tersebut.

Anda akan mengetahui bahwa hal berikut juga berlaku:

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

```
>$binomial(x,3) // C(x,3)$binomial(25,5)
```

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

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 sebuah persamaan dalam persamaan lainnya.

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

Seperti yang Anda lihat dalam contoh sebelumnya dan berikutnya, jika Anda memiliki LaTeX terinstal, Anda dapat mencetak ekspresi simbolik dengan 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 Anda tidak memiliki LaTeX terinstal.

```
>$(8-6)^2/(7^2+1)
```

$$\frac{2}{25}$$

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

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

Ekspresi simbolik diparsing oleh Euler. Jika Anda membutuhkan sintaks yang kompleks dalam satu ekspresi, Anda dapat menyertakan ekspresi dalam "...". Menggunakan lebih dari satu ekspresi sederhana memungkinkan, tetapi sangat tidak dianjurkan.

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

25

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

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

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

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

Sekali lagi, % merujuk pada hasil sebelumnya.

Untuk mempermudah, kita simpan 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 (25!)
```

$$2^{22}\,3^{10}\,5^{6}\,7^{3}\,11^{2}\,13\,17\,19\,23$$

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

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

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

$$2^{18}\,3^{8}\,5^{4}\,7^{2}\,11\,13\,17\,19$$

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

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

$$g^{2}$$

```
> fx &= x^5*exp(y), $fx
```

```
 5   y
x  E
```

$$x^5\,e^y$$

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

```
 3   x
x  E
```

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

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

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

```
      5
125 E
```

$$125\,e^5$$

```
18551.64488782208
```

```
>fx(5)
```

```
    18551.6448878
```

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

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

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

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

```
2.20079141499189e+7
```

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

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

Untuk mendapatkan kode LaTeX dari sebuah ekspresi, Anda dapat menggunakan perintah 'tex'.

```
>tex(fx)
```

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

Ekspresi simbolik dapat dievaluasi seperti halnya ekspresi numerik.

```
>fx(0.5)
```

```
0.206090158838
```

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

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

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

Penugasan juga dapat bersifat simbolik.

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

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

```
>$&fx with x=10+t
```

$$(t+10)^5\, e^{y}$$

Perintah 'solve' menyelesaikan ekspresi simbolik untuk variabel di Maxima. Hasilnya adalah vektor solusi.

```
> $&solve(x^5+x=7,x)
```

$$\left[0 = x^5 + x - 7\right]$$

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

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

Bandingkan dengan perintah numerik "solve" di Euler, yang memerlukan nilai awal dan, jika perlu, nilai target.

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

```
1.56155281281
```

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

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

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

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

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

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

```
>$&solve(x^3+x=1,x), x3 &= x with %[3]; $&x3
```

$$\left[x=\left(\frac{\sqrt{31}}{23^{\frac{3}{2}}}+\frac{1}{2}\right)^{\frac{1}{3}}\left(-\frac{\sqrt{3}i}{2}-\frac{1}{2}\right)-\frac{\frac{\sqrt{3}i}{2}-\frac{1}{2}}{3\left(\frac{\sqrt{31}}{23^{\frac{3}{2}}}+\frac{1}{2}\right)^{\frac{1}{3}}},x=\left(\frac{\sqrt{31}}{23^{\frac{3}{2}}}+\frac{1}{2}\right)^{\frac{1}{3}}\left(\frac{\sqrt{3}i}{2}-\frac{1}{2}\right)-\frac{-\frac{\sqrt{3}i}{2}-\frac{1}{2}}{3\left(\frac{\sqrt{31}}{23^{\frac{3}{2}}}+\frac{1}{2}\right)^{\frac{1}{3}}},x=\left(\frac{\sqrt{31}}{23^{\frac{3}{2}}}+\frac{1}{2}\right)^{\frac{1}{3}}-\frac{1}{3\left(\frac{\sqrt{31}}{23^{\frac{3}{2}}}+\frac{1}{2}\right)^{\frac{1}{3}}}\right]$$

$$\left(\frac{\sqrt{31}}{23^{\frac{3}{2}}}+\frac{1}{2}\right)^{\frac{1}{3}}-\frac{1}{3\left(\frac{\sqrt{31}}{23^{\frac{3}{2}}}+\frac{1}{2}\right)^{\frac{1}{3}}}$$

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

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

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

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

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

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

$$2$$

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

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

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

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

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

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

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

Fungsi

Dalam matematika, fungsi adalah suatu relasi yang menghubungkan setiap elemen dari satu himpunan(disebut domain)dengan tepat satu elemen di himpunan lain(disebut kodomain). sedangkan dalam EMT, fungsi adalah program yang didefinisikan dengan perintah "function". Fungsi ini bisa berupa fungsi satu baris atau fungsi multi-baris.

Fungsi satu baris bisa berupa fungsi numerik atau simbolik. Fungsi numerik satu baris didefinisikan dengan ":=".

```
>function f(y) := y^2*sqrt(7y^3-5)
>f(20)
```

```
    94653.050664
```

```
>function f(m) := (3m^2-2m-m^3=2)-(5m^2-8m-m^3+4)
>f(5)
```

```
    25.495097568
```

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

Untuk gambaran umum, berikut adalah semua kemungkinan definisi untuk fungsi satu baris. Sebuah fungsi dapat dievaluasi 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 tersebut ter-vectorisasi.

```
>f(4:5.6:6)
```

```
    16.4924225025
```

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

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

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

```
      1
```

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

```
      0.786151377757
```

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

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

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

```
      0.707106781187
```

Sebaiknya kita hilangkan definisi ulang tentang 'sin' ini.

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

```
      0.707106781187
```

Parameter Bawaan

Fungsi numerik dapat memiliki parameter bawaan. Parameter bawaan dalam pemrograman merujuk pada parameter fungsi saat dipanggil. Jika fungsi dipanggil tanpa menyertakan nilai untuk parameter tersebut, maka nilai default yang ditentukan akan digunakan.

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

Menghilangkan parameter ini akan menggunakan nilai default.

```
>f(4)
```

```
      16
```

Menyetelnya akan menimpa nilai default.

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

```
      80
```

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

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

```
      16
```

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

```
>function f(x,a=10) := a*2x^3
>f(5)
```

2500

```
>f(7,11)
```

7546

```
>f(17,a=10)
```

98260

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

24

Namun parameter yang ditetapkan mengesampingkan nilai global.

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

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

20

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

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

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

Fungsi simbolik dapat digunakan dalam ekspresi simbolik.

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

$$\frac{d}{d\,x}\,g\left(x\right)$$

$$\%at\left(\frac{d}{d\,x}\,g\left(x\right),x=\frac{4}{3}\right)$$

```
>function k(x) &= 2*x^5-x*exp(-5*x); $&k(x)
```

$$2\,x^5 - x\,e^{-5\,x}$$

```
>$&diff(k(x),x), $&% with x=102/77
```

$$5\,x\,e^{-5\,x}-e^{-5\,x}+10\,x^4$$

$$\frac{433\,e^{-\frac{510}{77}}}{77}+\frac{1082432160}{35153041}$$

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

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

```
    178.635099908
```

Mereka dapat digunakan untuk mendefinisikan fungsi simbolik atau ekspresi lainnya.

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

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

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

```
    0.703467422498
```

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

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

```
    0.703467422498
```

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

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

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

$$(x+2)^n$$

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

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

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

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

```
    625
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$$x^3-x$$

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

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

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

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

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

yang tidak dapat dievaluasi secara simbolik.

Jika kita mendefinisikan ulang fungsi tersebut dengan kata kunci "map", fungsi tersebut dapat digunakan untuk vektor x. Secara internal, fungsi ini 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 tersebut dapat dipanggil dengan atau tanpa parameter "base".

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

```
    2
    6.7
```

Selain itu, memungkinkan untuk menggunakan parameter yang telah ditetapkan.

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

```
    2
```

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

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

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

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

Fungsi simbolik seperti itu dapat digunakan untuk variabel simbolik.

Namun, fungsi tersebut juga dapat digunakan untuk vektor numerik.

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

```
    17
```

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

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

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

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

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

$$0$$

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

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

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

Untuk merangkum

- &= mendefinisikan fungsi simbolik,
- := mendefinisikan fungsi simbolik,
- &&= mendefinisikan fungsi yang sepenuhnya simbolik.

Menyelesaikan Ekspresi

Ekspresi dapat diselesaikan secara numerik dan simbolik.

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

```
>solve("3x^2+8x+4",1)
```

```
    -0.666666666667
```

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

```
    1.41421356237
```

Ini juga berlaku untuk ekspresi simbolik. Ambil fungsi berikut ini.

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

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

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

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

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

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

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

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

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

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

```
>$&solve(3*[5-3*(4-t)]-2=5*[3*(5*t-4)=8],t)
```

$$[[37-66\,t=9\,t-63]=0]$$

```
>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
```

```
>km &=7*m^10+m^5-m^7-m; $&km
```

$$7\,m^{10}-m^7+m^5-m$$

```
>km
```

```
                10     7     5
           7 m    - m   + m   - m
```

```
    Function solvekm not found.
    Try list ... to find functions!
    Error in:
    solvekm(%) (km,1,y=2) ...
               ^
```

Menyelesaikan ekspresi simbolik dalam bentuk simbolik mengembalikan daftar solusi. Kita menggunakan pemecah simbolik 'solve()' yang disediakan oleh Maxima.

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

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

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

```
>longest sol()
```

```
        -0.6180339887498949       1.618033988749895
```

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

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

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

$$0$$

Menyelesaikan sistem persamaan secara simbolik dapat dilakukan dengan vektor persamaan dan pemecah simbolik 'solve()'. Jawabannya 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 meneruskan parameter tambahan ke f() adalah dengan menggunakan daftar yang berisi nama fungsi dan parameter (cara lainnya adalah menggunakan parameter titik koma).

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

```
    2.54116291558
```

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

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

```
    2.54116291558
```

Menyelesaikan Pertidaksamaan

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

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

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

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

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

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

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

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

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

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

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

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

$$emptyset$$

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

$$universalset$$

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

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

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

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

```
>$&fourier_elim((2*x+3*y+z>7) and (4*x-2*y-z<8),[x,y])
```

$$fourier\_elim(z + 3y + 2x > 7 \wedge -z - 2y + 4x < 8, [x, y])$$

```
>$&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])
```

$$fourier\_elim(y + x < 5 \wedge x < 1 \vee x - y > 8, [x, y])$$

```
>$&fourier_elim((2*x + 3*y < 7) and (2*x - y >2),[x,y])
```

$$fourier\_elim(3y + 2x < 7 \wedge 2x - y > 2, [x, y])$$

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

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

```
>&fourier_elim([max(11*x^2,5*y^3) > 10, x # 7, abs(2*y-1) > 8],[x,y])
```

$$\text{fourier\_elim}\left(\left[\max\left(11\,x^2,\,5\,y^3\right) > 10,\; x \# 7,\; \text{mabs}(2\,y-1) > 8\right],\,[x,\,y]\right)$$

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

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

```
>$&fourier_elim([(x^5+6)/(4*x^7-9) <= 10],[x])
```

$$\text{fourier\_elim}\left(\left[\frac{x^5+6}{4\,x^7-9} \le 10\right],[x]\right)$$

Bahasa Matriks

Dokumentasi inti EMT memuat pembahasan rinci tentang bahasa matriks Euler. Matriks adalah susunan bilangan, simbol, atau ekspresi yang diatur dalam baris dan kolom dalam bentuk persegi panjang.

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

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

```
                1              2
                3              4
```

Produk matriks dilambangkan dengan titik.

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

```
                3
                4
```

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

```
    [3,  4]
```

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

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

```
>A.b //perkalian matriks

    11
    25

>A.inv(A)

     1     0
     0     1
```

Pokok utama dari bahasa matriks adalah bahwa semua fungsi dan operator bekerja pada 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 produk matriks, tetapi perkalian elemen demi elemen. Hal yang sama berlaku untuk vektor.

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

                9
               16
```

Jika salah satu operan adalah vektor atau skalar, maka operan tersebut akan diperluas dengan cara yang alami.

```
>2*A

                2             4
                6             8
```

E.g., Misalnya, jika operannya adalah vektor kolom, elemen-elemennya akan diterapkan ke semua baris dari A.

```
>[1,2]*A

                1             4
                3             8
```

Jika operannya adalah vektor baris, maka elemen-elemennya akan 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 dengan A.

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

```
                  1              2
                  1              2
```

```
>A*dup([1,2],2)
>M=[3,5;7,7]
```

$$M = \begin{bmatrix} 3 & 5 \\ 7 & 7 \end{bmatrix}$$

```
>N=[7;4]
```

$$N = \begin{bmatrix} 7 \\ 4 \end{bmatrix}$$

```
>M*[5,7]
```

```
[5 M, 7 M]
```

```
>maximamode on
```

```
    Maxima mode is on (compatibility mode)
```

Ini juga berlaku untuk dua vektor di mana satu adalah vektor baris dan yang lainnya adalah vektor kolom. Kita menghitung i*j untuk i, j dari 1 hingga 5. Triknya adalah mengalikan 1:5 dengan transposenya. Bahasa matriks Euler secara otomatis menghasilkan tabel nilai.

```
>(1:5)*(1:5)' // hasilkali elemen-elemen vektor baris dan vektor kolom
```

```
    incorrect syntax: ' is not an infix operator
    (1:5)*(1:5)';
              ^
```

Sekali lagi, ingat bahwa ini bukan produk matriks!

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

```
incorrect syntax: ' is not an infix operator
(1:5).(1:5)';
          ^
```

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

```
sum: wrong number of arguments.
 -- an error. To debug this try: debugmode(true);
```

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

```
>(3:2)*(9:11)'
```

```
incorrect syntax: ' is not an infix operator
(3:2)*(9:11)';
            ^
```

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

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

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

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

```
5
```

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

Kita mendapatkan vektor yang berisi 0 dan 1, di mana 1 menunjukkan benar (true).

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

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

Dari vektor seperti itu, "nonzeros" memilih elemen-elemen yang tidak nol.

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

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

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

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

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

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

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

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

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

EMT tidak sepenuhnya efektif untuk komputasi integer karena menggunakan floating point presisi ganda secara internal. Namun, seringkali masih sangat berguna.

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

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

```
112
```

Fungsi'nonzeros()' hanya 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 'mnonzeros()' mengembalikan indeks dari elemen-elemen yang tidak nol dalam matriks.

```
>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 mengatur elemen-elemen tersebut ke nilai tertentu.

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

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

Fungsi 'mset()' juga dapat mengatur elemen pada indeks-indeks tersebut ke entri dari matriks lainnya.

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

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

Dan memungkinkan untuk mendapatkan elemen-elemen tersebut dalam bentuk vektor.

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

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

Fungsi lain yang berguna adalah 'extrema', yang mengembalikan nilai minimum dan maksimum di setiap baris matriks serta posisi-posisinya.

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

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

Kita dapat menggunakan fungsi ini untuk mengekstrak nilai maksimum di setiap baris.

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

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

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

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

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

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

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

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

Fungsi Matriks Lain (Membangun Matriks)

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

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

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

Demikian pula, kita dapat menempelkan sebuah matriks di samping matriks lainnya 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 kedua matriks tidak memiliki jumlah baris yang sama, matriks yang lebih pendek akan diisi dengan nol.

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
```

Memungkinkan untuk membuat matriks dari vektor baris dan vektor kolom.

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

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

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

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

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

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

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

Untuk mendapatkan ukuran dari 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
```

Ada banyak fungsi lain yang menghasilkan matriks.

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

```
                1             1
                1             1
```

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

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

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

Juga, matriks dengan angka acak dapat dihasilkan menggunakan 'random' (distribusi seragam) atau 'normal' (distribusi Gauss).

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

```
          0.66566      0.831835
            0.977      0.544258
```

Berikut adalah fungsi berguna lainnya, yang mengubah struktur elemen dari sebuah matriks menjadi matriks lainnya.

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

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

Dengan fungsi berikut, kita dapat menggunakan ini dan fungsi 'dup' untuk menulis fungsi 'rep()', yang mengulang vektor 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()' menggandakan elemen-elemen dari sebuah vektor.

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

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

Fungsi 'flipx()' dan 'flipy()' membalikkan urutan baris atau kolom dari sebuah matriks. Misalnya, fungsi `flipx()` membalikkan secara horizontal.

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

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

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

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

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

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

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

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

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

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

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

As you see, it does not harm to include indices out of range (like 0), double indices, or unsorted indices.

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

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

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

Kita mulai dengan matriks identitas.

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

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

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

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

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

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

Berikut adalah fungsi yang mengembalikan matriks tri-diagonal.

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

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

Diagonal dari sebuah matriks juga dapat diambil dari matriks tersebut. Untuk menunjukkan hal 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 mengambil diagonalnya.

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

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

Misalnya, kita bisa membagi matriks dengan diagonalnya. Bahasa matriks akan memastikan bahwa vektor kolom d diterapkan pada matriks baris per baris.

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

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

Vektorisasi

vektorisasi adalah konsep dalam komputai yang mengacu pada teknik untuk menerapkan operasi matematika atau fungsi secara langsung pada vektor tanpa menggunakan perulangan eksplisit. Hampir semua fungsi di Euler juga bekerja untuk input matriks dan vektor, kapan pun hal tersebut masuk akal.

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

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

```
assignment: cannot assign to 1
 -- an error. To debug this try: debugmode(true);
```

```
>sqrt(5:9)
```

```
assignment: cannot assign to 5
 -- an error. To debug this try: debugmode(true);
```

Jadi, Anda dapat dengan mudah membuat tabel nilai. Ini adalah 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 dihasilkan dengan mudah.

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

$$s = t^3 - t$$

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

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

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

Misalnya, hasil kali antara vektor kolom dan vektor baris akan diperluas menjadi matriks jika sebuah operator diterapkan. Dalam contoh berikut, v' adalah vektor yang ditransposisi (sebuah vektor kolom).

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

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

Perhatikan bahwa ini berbeda dari produk matriks. Produk matriks dilambangkan dengan titik "." dalam EMT.

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

```
55
```

Secara default, vektor baris dicetak dalam format yang ringkas.

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

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

Untuk matriks, operator khusus . menyatakan perkalian matriks, dan A' menyatakan transposisi. Matriks 1x1 dapat digunakan seperti halnya bilangan real.

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

```
5
25
```

Untuk mentransposisikan matriks, kita menggunakan apostrof.

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

```
1
2
3
4
```

Jadi kita dapat menghitung matriks A dikali vektor b.

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

```
30
70
```

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

```
>v'.v
```

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

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

```
>v.v'
```

```
30
```

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

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

```
30
```

Operator dan fungsi mematuhi bahasa matriks Euler.

Berikut ini adalah ringkasan aturannya.

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

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

Berikut ini adalah kasus sederhana dengan operator ^.

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

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

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

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

1	2	3
2	4	6
3	6	9

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

```
>v.v'
```

```
    14
```

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

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

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

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

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

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

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

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

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

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

Elemen-elemen dari sebuah matriks disebut dengan ❓A[i,j]❓.

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

```
6
```

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

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

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

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

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

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

Bentuk singkat untuk : adalah menghilangkan indeks sepenuhnya.

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

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

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

```
>A{4}
```

```
4
```

Sebuah matriks juga dapat diratakan, dengan menggunakan fungsi 'redim()'. Hal ini diimplementasikan dalam fungsi 'flatten()'.

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

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

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

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

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

Sekarang kita menambahkan kolom ke matriks.

```
>M = deg(w)|w|cos(w)|sin(w)
```

```
                0                0                1                0
               45         0.785398         0.707107         0.707107
               90           1.5708                0                1
              135          2.35619        -0.707107         0.707107
              180          3.14159               -1                0
              225          3.92699        -0.707107        -0.707107
              270          4.71239                0               -1
              315          5.49779         0.707107        -0.707107
              360          6.28319                1                0
```

Dengan menggunakan bahasa matriks, kita dapat membuat 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. Dengan kata lain, matriks tersebut memiliki elemen-elemen lateks: a_{i, j} = t_j^i, \quad 1 \le j \le 101, \quad 1 \le i \le n

Sebuah fungsi yang tidak bekerja untuk input vektor harus "divektorkan". Hal ini dapat dicapai dengan kata kunci "map" dalam definisi fungsi. Kemudian fungsi akan dievaluasi untuk setiap elemen parameter vektor.

Integrasi numerik integrate() hanya bekerja untuk batas interval skalar. Jadi kita perlu membuat vektornya.

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

Kata kunci "map" membuat vektor fungsi. Fungsi ini 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 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 baris lengkap dari sebuah matriks.

```
>A[2]
```

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

Untuk vektor baris atau kolom, ini mengembalikan elemen vektor.

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

```
2
```

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

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

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

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

Di sini kita menginginkan baris pertama dan kedua dari A.

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

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

Kita bahkan dapat menyusun ulang A menggunakan vektor indeks. Tepatnya, kita tidak mengubah A di sini, tetapi menghitung versi susunan ulang dari A.

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

```
                7              8              9
                4              5              6
```

```
                1               2               3
```

Trik indeks juga bekerja dengan kolom.

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

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

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

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

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

```
                3
                6
                9
```

Sebagai alternatif, biarkan indeks pertama kosong.

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

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

Kita juga bisa mendapatkan baris terakhir A.

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

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

Sekarang mari kita ubah elemen-elemen dari A dengan memberikan sebuah submatriks dari A ke suatu nilai. Hal ini sebenarnya mengubah matriks A yang tersimpan.

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

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

Kita juga dapat menetapkan nilai pada deretan A.

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

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

Kami bahkan dapat menetapkan ke sub-matriks jika memiliki ukuran yang tepat.

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

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

Selain itu, beberapa jalan pintas diperbolehkan.

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

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

Peringatan: Indeks di luar batas akan mengembalikan matriks kosong, atau pesan kesalahan, tergantung pada pengaturan sistem. Standarnya adalah pesan kesalahan. Namun, ingatlah bahwa indeks negatif dapat digunakan untuk mengakses elemen-elemen matriks yang dihitung dari akhir.

```
>A[4]
```

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

Pengurutan dan Pengocokan

Fungsi mengurutkan() mengurutkan vektor baris.

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

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

Sering kali diperlukan untuk mengetahui indeks vektor yang diurutkan dalam vektor aslinya. Hal ini dapat digunakan untuk menyusun ulang vektor lain dengan cara yang sama.

Mari kita mengacak sebuah vektor.

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

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

Indeks berisi urutan v yang tepat.

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

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

Hal ini juga berlaku untuk vektor string.

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

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

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

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

Seperti yang Anda lihat, posisi entri ganda agak acak.

```
>ind
```

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

Fungsi unique mengembalikan daftar terurut dari elemen unik sebuah vektor.

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

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

Hal ini juga berlaku untuk vektor string.

```
>unique(s)
```

```
a
aa
d
e
```

Aljabar Linier

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

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

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

```
-4
4.5
```

Sebagai contoh lain, kita membuat matriks 200x200 dan jumlah barisnya. Kemudian kita selesaikan Ax = b dengan menggunakan matriks kebalikannya. Kita mengukur kesalahan sebagai deviasi maksimal dari semua elemen dari 1, yang tentu saja merupakan solusi yang benar.

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

```
      8.790745908981989e-13
```

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

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

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

Determinan dari matriks ini adalah 0.

```
>det(A)
```

```
    0
```

Matriks Simbolik

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Untuk mengekstrak vektor eigen tertentu, diperlukan pengindeksan yang cermat.

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

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

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

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

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

```
                1              4
                5              2
```

Dalam ekspresi simbolik, gunakan dengan.

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

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

Akses ke baris matriks simbolik bekerja seperti halnya matriks numerik.

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

$$[1,a]$$

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

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

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

Terdapat fungsi-fungsi simbolik dalam Maxima untuk membuat vektor dan matriks. Untuk hal 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)
```

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

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

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

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

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

Euler juga memiliki sebuah fungsi yang kuat xinv(), yang melakukan usaha yang lebih besar dan mendapatkan hasil yang lebih tepat.

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

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

```
                              1                             0
                              0                             1
```

Misalnya, nilai eigen dari A dapat dihitung secara numerik.

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

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

Atau secara simbolis. Lihat tutorial tentang Maxima untuk mengetahui detailnya.

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

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

Nilai Numerik dalam Ekspresi simbolik

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

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

```
                1        3.14159
                4              5
```

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

```
>$&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 angka floating point, dan bahkan dengan angka mengambang yang besar dengan 32 digit. Namun, evaluasinya jauh lebih lambat.

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

$$1.41421356237309504880168872420 97_B \times 10^0$$

$$1.414213562373095$$

Ketepatan angka floating point yang besar dapat diubah.

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

```
          3.14159265358979323846264338327950288419716939937510582097494\
    459230781640628620899862803482534211706 8b0
```

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

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

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

$$-5.424777960769379$$

Demo - Suku Bunga

Di bawah ini, kami menggunakan Euler Math Toolbox (EMT) untuk menghitung suku bunga. Kami melakukannya secara numerik dan simbolis untuk menunjukkan kepada Anda bagaimana Euler dapat digunakan untuk memecahkan masalah kehidupan nyata.

Asumsikan Anda memiliki modal awal sebesar 5000 (katakanlah dalam dolar).

```
>K=5000
```

```
     5000
```

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

```
>K*1.03
```

```
     5150
```

Euler juga akan memahami sintaks berikut.

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

```
     5150
```

Tetapi akan lebih mudah menggunakan faktor

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

```
     1.03
     5150
```

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

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

```
     6719.58189672
```

Untuk tujuan kita, kita bisa menetapkan formatnya menjadi 2 digit setelah titik desimal.

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

```
          6719.58
```

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

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

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

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

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

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

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

```
>short 0:10
```

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

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

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

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

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

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

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

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

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

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

```
                  1271.61
                1271.6071
```

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

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

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

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

Kita bisa mencetaknya dengan cara yang bersahabat, menggunakan format kami dengan angka desimal yang tetap.

```
>VKr'
```

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

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

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

```
5150.00
5000.00     5150.00     5304.50
```

Yang mengejutkan, kita juga dapat menggunakan vektor indeks. Ingatlah bahwa 1:3 menghasilkan vektor [1,2,3].

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

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

```
6719.60
6719.58
```

Perbedaannya sangat kecil.

Menyelesaikan Persamaan

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

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

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

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

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

Bagaimana jika kita menghapus jumlah yang sama setiap tahun?

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

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

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

Bagaimana kita dapat menentukan berapa tahun uang itu akan bertahan? Kita harus menulis perulangan untuk ini. Cara termudah adalah dengan melakukan perulangan yang cukup lama.

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

```
    Real 1 x 51 matrix
        5000.00     4950.00     4898.50     4845.45     ...
```

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

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

```
          48.00
```

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

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

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

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

```
         -19.83
          47.00
```

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

Ini adalah pertanyaan yang hanya bisa dijawab secara numerik. Di bawah ini, kami akan menurunkan rumus yang diperlukan. Kemudian Anda akan melihat bahwa tidak ada rumus yang mudah untuk suku bunga. Namun untuk saat ini, kita akan mencari solusi numerik.

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

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

Perulangannya sama seperti di atas

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

Tetapi kita tidak lagi menggunakan nilai global R dalam ekspresi kita. Fungsi-fungsi seperti iterate() memiliki trik khusus dalam Euler. Anda bisa mengoper nilai variabel dalam ekspresi sebagai parameter titik koma. Dalam hal ini P dan R.

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

Mari kita coba sebuah tes.

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

```
          -19.83
```

Sekarang kita bisa menyelesaikan masalah kita.

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

```
           3.15
```

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

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

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

```
         -336.08
```

Perhatikan bahwa Anda tidak dapat menyelesaikan jumlah tahun, karena fungsi kita mengasumsikan n sebagai nilai bilangan bulat.

Solusi Simbolik untuk Masalah Suku Bunga

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

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

$$R+q\,K$$

Sekarang kita dapat mengulanginya.

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

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

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

Kita melihat 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 untuk jumlah geometris, yang dikenal dengan Maxima.

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

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

Ini sedikit rumit. Penjumlahan dievaluasi dengan flag "simpsum" untuk menguranginya menjadi hasil bagi.

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. Tetapi fungsi ini lebih efektif.

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

```
        -19.82504734650985
        -19.82504734652684
```

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

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

```
         20.51
```

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

Kita juga bisa menggunakan sisi simbolis dari Euler untuk menghitung rumus pembayaran.

Asumsikan kita mendapatkan pinjaman sebesar K, dan membayar n kali pembayaran sebesar R (dimulai setelah tahun pertama) sehingga menyisakan sisa utang sebesar Kn (pada saat pembayaran terakhir). Rumus untuk hal ini adalah sebagai berikut

```
>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 memecahkan laju R secara simbolis.

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

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

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

Tentu saja, kita memiliki batas berikut.

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

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

Jelasnya, tanpa bunga kita harus membayar kembali 10 suku bunga 500.

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

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

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