TUGAS KELOMPOK 1

Kelas : MATEMATIKA B 2023

Nama Anggota Kelompok :

Siti Faltipah Hayani (23030630004)

Lasigi Yatindra Jago (23030630008)

Indah Sari Wijayanti (23030630014)

Aifa Maladina Ulya (23030630017)

Alya Putri Medilasari (23030630018)

Muhammad Lutfi Ramadhan (23030630021)

Nazwa Yuan Adelia Putri (23030630095)

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:

>$&5*x^(-4)*y3*-8*x^5*y(-6)

$$-\frac{40\,x}{y^3}$$

Menyederhanakan fungsi :

>$&5*y^2+3*x6-7*y^2+2*x2

$$-2\,y^2+3\,x^6+2\,x^2$$

Menjabarkan:

>$&showev('expand((5*x^{-4}+y3)*(-8*x^5-y{-6})))

$$\text{expand}\left(\left(y^3+5\,x^{\{-4\}}\right)\,\left(-y^{\{-6\}}-8\,x^5\right)\right)=-y^{\{-6\}+3}-5\,x^{\{-4\}}\,y^{\{-6\}}-8\,x^5\,y^3-40\,x^{\{-4\}+5}$$

Baris Perintah

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

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

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

```
37.6991118431
```

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

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

```
75.3982236862
150.796447372
```

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

>x := 4;

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

```
-0.653643620864
```

>x := cos(x)

```
0.793873449226
```

Jika dua garis terhubung dengan "…" kedua garis akan selalu dieksekusi secara bersamaan.

>x := 3.5; …
> x := (x+2/x)/2, x := (x+2/x)/2, x := (x+2/x)/2,

```
2.03571428571
1.50908521303
1.41719571011
```

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

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

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

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

```
81
```

Euler Math Toolbox mendukung loop di baris perintah, selama mereka masuk ke dalam satu baris atau multi-baris. Dalam program, pembatasan ini tidak berlaku, tentu saja. Untuk informasi lebih lanjut lihat pengantar berikut.

>x=6; for i=1 to 10; x := (x+2/x)/2, end; // menghitung akar 2

```
3.16666666667
1.89912280702
1.47612029496
1.4155117098
1.41421415763
1.41421356237
1.41421356237
1.41421356237
1.41421356237
1.41421356237
```

Tidak apa-apa untuk menggunakan multi-line. Pastikan baris diakhiri dengan "…".

>x := 2.5; // comments go here before the …
> repeat xnew:=(x+2/x)/2; until xnew~=x; …
> x := xnew; …
> end; …
> x,

```
1.41421356237
```

Struktur bersyarat juga berfungsi.

>if E^pi>pi E; then "Halo Rakyatku!", endif;

```
Halo Rakyatku!
```

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

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

>sqrt(sin(45°)/cos(60°))

```
1.189207115
```

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

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

>exp(log(5.7))

```
5.7
```

Sintaks Dasar

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

Untuk menyetel variabel, gunakan "=" atau ":=". Demi kejelasan, pengantar ini menggunakan bentuk yang terakhir/terbaru. Spasi tidak menjadi masalah. Tetapi ruang antara perintah diharapkan untuk ada.

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

>g:=10.73; t:=4.2; 1/2*g*t^2

```
94.6386
```

EMT menggunakan sintaks pemrograman untuk ekspresi. Untuk mengetik

Anda harus mengatur tanda kurung dengan benar dan menggunakan "/" untuk pecahan. Perhatikan tanda kurung yang disorot untuk bantuan. Perhatikan bahwa konstanta Euler e diberi nama E dalam EMT.

>E^2*(1/(9+5*log(0.7))+4/9)

```
4.30791848586
```

Untuk menghitung ekspresi rumit seperti

Anda harus memasukkannya dalam bentuk baris.

>((5/8 + 7/6 + 3) / (3/7 + 5/9))^2 * pi

```
74.4767625442
```

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

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

>2/3+5/7, fraction %

```
1.38095238095
29/21
```

Perintah Euler dapat berupa ekspresi atau perintah primitif. Ekspresi terbuat dari operator dan fungsi. Jika diperlukan, hal tersebut harus berisi tanda kurung untuk memaksa urutan eksekusi yang benar. Jika ragu, memasang braket atau tanda kurung adalah ide yang bagus. Perhatikan bahwa EMT menunjukkan tanda kurung buka dan tutup saat mengedit baris perintah.

>(cos(pi/4)+2)^3*(sin(pi/4)+5)2

```
646.172032434
```

Operator numerik Euler meliputi

- unary atau operator plus
- unary atau operator minus
- operator perkalian
  / operator pecahan
  . produk matriks
  a^b daya untuk positif a atau bilangan bulat b (a**b juga berfungsi)
  n! operator faktorial dan masih banyak lagi.

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

sin, cos, tan, atan, asin, acos, rad, deg
log, exp, log10, sqrt, logbase
bin, logbin, logfac, mod, lantai, ceil, bulat, abs, tanda
conj, re, im, arg, conj, nyata, kompleks
beta, betai, gamma, complexgamma, ellrf, ellf, ellrd, elle
bitand, bitor, bitxor, bitnot

Beberapa perintah memiliki alias, mis. ln untuk log.

>ln(E^4), arctan(tan(0.75)), logbase(30,10)

```
4
0.75
1.47712125472
```

>sin(90°)

```
1
```

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

>$2^{3}4$, $(2^{3)}4$, $2^{(3}4)$

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

Bilangan Asli

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

>longest(23/3)

```
7.666666666666667
```

Representasi ganda internal membutuhkan 8 byte.

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

>printdual(23/3)

```
1.1110101010101010101010101010101010101010101010101010101011*2^2
```

>printhex(1/7)

```
2.4924924924924*16^-1
```

Perbedaan ‘printdual’ dan ‘printhex’ adalah ‘printdual’ yakni mencetak representasi internal dari sebuah bilangan floating-point dalam format presisi ganda (pendekatan yang sangat dekat dengan nilai aslinya tetapi tidak persis sama.) meskipun ia tergantung pada konteks bahasa pemograman tertentu. sedangkan ‘printhex’ yakni representasi dari nilai floating-point dalam bentuk heksadesimal(basis 16), heksadesimal ini adalah cara yang lebih ringkas untuk menampilkan nilai biner karena setiap digit heksadesimal mempresentasikan empat digit biner.

String

Sebuah string dalam Euler didefinisikan dengan “…”

>“A string can contain anything.”

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

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

>“Terjadi Gempa Mag pada hari Senin 26 Agustus 2024 dengan pusat gempa berada di laut” +95+ ” km barat daya Gunungkidul.”

```
Terjadi Gempa Mag pada hari Senin 26 Agustus 2024 dengan pusat gempa berada
di laut 95 km barat daya Gunungkidul.
```

Pada String fungsi print mengonversi angka menjadi string. Ini dapat mengambil sejumlah digit dan sejumlah tempat (0 untuk keluaran padat), dan secara optimal satu unit

>“Golden Ratio :” + print((1+sqrt(5))/2,5,0)

```
Golden Ratio : 1.61803
```

Terdapat spesial string ‘none’, yang tidak dicetak.

>none

Untuk mengonversi string menjadi angka, cukup mengevaluasinya. Ini bekerja untuk ekspresi juga (lihat dibawah).

>“1234.5567”()

```
1234.5567
```

Untuk mendefinisikan vektor string, gunakan notasi vektor […]

>v:= [“Indonesia”,“Malaysia”,“Brunei Darussalam”]

```
Indonesia
Malaysia
Brunei Darussalam
```

Vektor pada string kosong dilambangkan dengan [none]. Dan vektor string dapat digabungkan dengan ‘|’.

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

```
Indonesia
Malaysia
Brunei Darussalam
Indonesia
Malaysia
Brunei Darussalam
```

String dapat berisi karakter Unicode. Secara internal, string ini berisi kode UTF-8. untuk menghasilkan string seperti itu, gunakan u”…” dan salah satu entitas HTML. String Unicode dapat digabungkan seperti string lainnya.

>u”β = ” + 90 + u”° ” // pdfLaTeX mungkin gagal menampilkan secara benar

```
β = 90°
```

Dalam komentar, entitas yang sama seperti alpha; beta; dll dapat

digunakan untuk lateks.

Ada beberapa fungsi untuk membuat atau menganalisis string unicode.

Fungsi strtochsr() akan mengenali string Unicode, dan menerjemahkannya

dengan benar.

>v=strtochar(u”Ä is a German letter”)

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

Perintah ini menghasilkan array atau daftar angka berupa vektor angka yang mewakili karakter dalam string dalam bentuk kode Unicode.

Fungsi kebalikannya adalah chartoutf().

>v[1]=strtochar(u”Ä“)[1]; chartoutf(v)

```
Ä is a German letter
```

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

>a=“We have α=β.”; utf(a)// PdfLaTeX mengkin gagal menampilkannya

```
We have α=β.
```

Memungkinkan juga untuk menggunakan entitas numerik.

>u”Älphabet”

```
Älphabet
```

Nilai Boolean

Nilai boolean direpresentasikan dengan 1=true atau 0=false dalam euler. String dapat dibandingkan, seperti halnya angka.

>“saya”>“aku”, 6==3

```
1
0
```

>5>1, “mobil”==“motor”

```
1
0
```

“dan” adalah operator “&&” dan “atau” adalah operator “||”, seperti dalam bahasa C. (Kata-kata “dan” dan “atau” hanya dapat digunakan dalam kondisi “jika”.

```
>2<E || E<3
```

```
1
```

```
>6>E && E<2
```

```
0
```

Operator Boolean mematuhi aturan bahasa matriks

```
>(2:9)>3, nonzeros (%)
```

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

Kita dapat menggunakan fungsi bukan nol() untuk mengekstrak elemen tertentu dari vektor. Dalam contoh,menggunakan isprima bersyarat(n).

```
>N=5 | 7:2:50 // N berisi elemen 5 dan bilangan bilangan ganjil dari 7:50
```

```
[5,  7,  9,  11,  13,  15,  17,  19,  21,  23,  25,  27,  29,  31,  33,
35,  37,  39,  41,  43,  45,  47,  49]
```

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

```
[5,  7,  11,  13,  17,  19,  23,  29,  31,  37,  41,  43,  47]
```

Output Formats

Default output formats EMT adalah 12 digit. Untuk memastikan yang kita lihat adalah bentuk default, maka perlu direset format.

```
>defformat; pi
```

```
3.14159265359
```

Secara internal, EMT menggunakan standar IEEE (Institute of Electrical and Electronics Engineers) untuk bilangan ganda dengan sekitar 16 digit desimal. Untuk melihat bentuk digit penuh, gunakan perintah “longestformat” atau gunakan operator “longest” untuk memunculkannya.

```
>longest pi
```

```
      3.141592653589793
```

```
>longestformat; pi
```

```
3.141592653589793
```

Berikut ini adalah repesentasi heksadesimal internal dari bilangan ganda.

```
>printhex(pi)
```

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

Heksadesimal adalah sistem bilangan yang menggunakan basis 16. Di mana angka 0 hingga 9 (untuk mewakili nilai 0 hingga 9) dan huruf A hingga F (untuk mewakili nilai 10 hingga 15).

Format standarnya adalah 12.

```
>format(12); 1/7
```

```
0.142857142857
```

Format output dapat diubah secara permanen dengan perintah format.

```
>format(12,5); 1/9, pi, cos(1)
```

```
    0.11111
    3.14159
    0.54030
```

Format standar untuk skalar adalah 12, tetapi ini dapat diubah.

```
>setscalarformat(7); pi
```

```
    3.14159
```

Begitu juga dengan fungsi “longestformat” mengatur format skalar.

```
>longestformat; pi
```

```
3.141592653589793
```

Notes: beberapa format output yang penting.

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

Akurasi internal EMT adalah sekitar 16 digit desimal mengikuti standar dari IEEE. Angka disimpan dalam format internal. Namun, format output EMT dapat diatur secara fleksibel.

>fraction 2+3/10+7/14+21/7

```
29/5
```

>fraction 5*7/2/3*2/5

```
7/3
```

Digunakan untuk menampilkan ke bentuk pecahan sederhana.

>longest 0.2+0.25+0.3+0.3+0.25+0.2+0.5-2.6

```
-0.6000000000000001
```

Perintah ini menunjukkan presisi penuh dari operasi aritmetika yang melibatkan angka-angka kecil, dan bagaimana kesalahan akumulatif bisa muncul dalam perhitungan biner.

Expressions

String atau nama dapat digunakan untuk menyimpan ekspresi matematika, yang dapat dievaluasi oleh EMT. Gunakan tanda kurung setelah ekspresi.

>k:=5; fx:="pi*k"; fx()

```
15.70796326794897
```

Ekspresi akan selalu menggunakan global variable, bahkan jika ada variabel dalam fungsi dengan nama yang sama.

>fx:="a*cos(x)"; fx(10,a=0.8)

```
-0.671257223261162
```

Menggunakan parameter yang ditetapkan ke x, y, z, dll. Jika tidak, evaluasi ekspresi dalam fungsi dapat memberikan hasil yang membingungkan bagi pengguna yang memanggil fungsi tersebut.

>at:=6; function f(expr,x,at) := expr(x); …
> f("at*x^4",2,3) // computes 6*2^4 not 3*2^4

```
96
```

Menggunakan global variable pada fungsi, dimana "at" merupakan global variables. Jika ingin menggunakan nilai lain untuk "at" perlu menambahkan "at=vaue".

>ut:=5; function f(expr,x,p) := expr(x,ut=p); …
> f("ut*x^3",4,2) // computes 2*4^3 not 5*4^3

```
128
```

Walaupun "ut" sebagai global variable sudah didefinisikan, tetapi didefinisikan kembali pada ekspresi fungsinya dimana "ut=p" sehingga nilainya berganti dari yang awalnya ut=5 menjadi ut=p=2.

>f &= x^2

$$x^2$$

>function f(x) := x^4

>f(2)

```
16
```

Ekspresi dalam x sering digunakan seperti fungsi.

Mendefinisikan fungsi dengan nama yang sama seperti ekspresi simbolik global (f &=)menghapus nilai variabel sebelumnya untuk menghindari kebingungan antara ekspresi simbolik dan fungsi.

> "@(a,b) $a^{3+b}2$", %(2,4)

```
@(a,b) a^3+b^2
24
```

Bentuk khusus dari ekspresi memungkinkan variabel apa pun sebagai parameter tanpa nama untuk evaluasi ekspresi, bukan hanya "x", "y" dll. Untuk ini, mulai ekspresi dengan "@(variabel) …".

>fx &= 2*x-3*t; …
> t=2.5; fx(0.8)

```
-5.9
```

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

>fx(1,t=1.5)

```
-2.5
```

Sebuah ekspresi tidak perlu simbolis. Ini diperlukan, jika ekspresi berisi fungsi, yang hanya diketahui di kernel numerik, bukan di Maxima.

Symbolic Mathematics

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

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

>$&35!

$$10333147966386144929666651337523200000000$$

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

Mari kita hitung!

lateks: C(35,15)=

>$& 35!/(20!*15!) // nilai C(35,15)

$$3247943160$$

Maxima memiliki fungsi yang lebih efisien untuk ini (seperti halnya bagian numerik dari EMT).

>$binomial(35,15) //menghitung C(35,15) menggunakan fungsi binomial()

$$3247943160$$

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

Anda akan belajar bahwa yang berikut ini juga berfungsi.

>$&binomial(x,3) with x=5 // substitusi x=5 ke C(x,3)

$$10$$

Dengan begitu kita dapat menggunakan solusi persamaan dalam persamaan lain.

>sol &= solve(x^2+3*x=9,x); $&sol, sol(), $&float(sol)

$$[x = \frac{-3\sqrt{5}-3}{2}, x = \frac{3\sqrt{5}-3}{2}]$$

[-4.854101966249685, 1.854101966249685]

$$[x = -4.854101966249685, x = 1.854101966249685]$$

Untuk mencetak ekspresi simbolis dengan LaTeX, gunakan $ di depan & (atau dapat menghilangkan &) sebelum perintah. Jangan menjalankan perintah Maxima dengan $, jika tidak menginstal LaTeX.

Untuk mendapatkan solusi simbolis tertentu, seseorang dapat menggunakan “with” dan index.

>$&solve(x^2+3*x=1,x), x2 &= x with %[2]; $&x2

$$[x = \frac{-\sqrt{13}-3}{2}, x = \frac{\sqrt{13}-3}{2}]$$

$$\frac{\sqrt{13}-3}{2}$$

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

>sol &= solve([x+y=5,x^2+y2=15],[x,y]); $&sol, $&x*y with sol[1]

$$[\,[\,x=\frac{5-\sqrt{5}}{2},y=\frac{\sqrt{5}+5}{2}\,],[\,x=\frac{\sqrt{5}+5}{2},y=\frac{5-\sqrt{5}}{2}\,]\,]$$

$$\frac{(5-\sqrt{5})\ (\sqrt{5}+5)}{4}$$

Ekspresi simbolis dapat memiliki bendera, yang menunjukkan perlakuan khusus di Maxima. Beberapa flag dapat digunakan sebagai perintah juga, yang lain tidak. Bendera ditambahkan dengan "|" (bentuk yang lebih bagus dari "ev(…,flags)").

Functions

Dalam matematika, fungsi aljabar adalah fungsi yang bisa didefinisikan sebagai akar dari sebuah persamaan aljabar. Fungsi aljabar merupakan ekspresi aljabar menggunakan sejumlah suku terbatas, yang melibatkan operasi aljabar seperti penambahan, pengurangan, perkalian, pembagian, dan peningkatan menjadi pangkat pecahan.

contohnya:

>$& f(x)=1/x

$$(x^2)(x)=\frac{1}{x}$$

>$& f(x)=sqrt(x)

$$(x^2)(x)=\sqrt{x}$$

>$& f(x)=(sqrt(1+x^3))/(x(3/7)-(sqrt(7)*x^(1/3)))

$$(x^2)(x)=\frac{\sqrt{x^3+1}}{x^{\frac{3}{7}}-\sqrt{7}x^{\frac{1}{3}}}$$

di EMT fungsi adalah program yang didefenisikan dengan perintah "function". Fungsi dapat menjadi fungsi satu baris atau fungsi multi baris. Fungsi satu baris dapat berupa numerik atau simbolis. Fungsi satu baris didefinisikan oleh ":=".

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

Lalu, semua fungsi pasti dapat didefinisikan oleh fungsi satu baris. Suatu fungsi dapat dievaluasi, contohnya kita akan mencari nilai f(5) dari fungsi f diatas.

>f(2)

```
4.47213595499958
```

Fungsi ini dapat digunakan juga dalam vektor, dengan mengikuti aturan bahasa matrik Euler, karena ekspresi yang digunakan dalam fungsi divektorkan.Kita akan mencobanya menggunakan fungsi f di atas.

>f(0:0.1:1)

```
[0,  0.1004987562112089,  0.2039607805437114,  0.3132091952673166,
0.4308131845707604,  0.5590169943749475,  0.699714227381436,
0.8544588931013591,  1.024499877989256,  1.210826164236634,
1.414213562373095]
```

Fungsi juga dapat menjadi plot, hanya dengan memberikan nama fungsi. Berbeda dengan ekpresi simbolik atau numerik, nama fungsi harus diberikan dalam string.

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

```
0.7861513777574233
```

Secara default, jika ingin menimpa fungsi bawaan bisa menambahkan kata kunci "overwrite". Menimpa fungsi bawaan berbahaya dan menyebabkan masalah bagi fungsi lain yang bergantung pada fungsi tersebut.

Jika ingin memanggil fungsi bawaan bisa memakai "_", jika fungsi tersebut merupakan fungsi inti Euler.

>function overwrite sin(x) := _sin(x°) // tentukan kembai sinus dalam derajat

>sin(45)

```
0.7071067811865476
```

Jika ingin menghapus definisi dari sin dan mendefinisikannya ulang, menggunakan perintah "forget"

>forget sin; sin(pi/4)

```
0.7071067811865476
```

Default Parameters

Parameter default adalah fungsi parameter yang memiliki nilai awal.

Fungsi numerik dapat memiliki parameter default.

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

Menghilangkan parameter ini menggunakan nilai default.

>f(4)

```
16
```

Menimpa default value.

>f(4,5)

```
80
```

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

>f(4,a=1)

```
16
```

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

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

>a=6; f(2)

```
24
```

Tetapi parameter yang ditetapkan menimpa global value.

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

>f(2,a:=5)

```
20
```

Fungsi simbolis didefinisikan dengan "$&=". Fungsi simbolis didefinisikan dalam Euler dan maxima,dan bekerja keduanya. Ekspresi yang mendefinisikan dijalankan melalui Maxima sebelum di 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 //1. turunan pertama dari g(x), 2. memasukkan nilai x=4/3

$$x\,e^{-x} - e^{-x} + 3\,x^2$$

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

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

>g(5+g(1))

```
178.6350999083138
```

Itu juga dapat digunakan untuk mendefinisikan fungsi atau ekspresi simbolik lainnya.

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

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

>solve(&g(x),0.5)

```
0.7034674224983917
```

Berikut ini juga berfungsi, karena Euler menggunakan ekspresi simbolis dalam fungsi g, jika tidak menemukan variabel simbolik g, dan jika ada fungsi simbolis g.

>solve(&g,0.5)

```
0.7034674224983917
```

Dengan &= fungsinya simbolis, dan dapat digunakan dalam ekspresi simbolik lainnya. Contohnya dalam integral tak tentu sebagai berikut.

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

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

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

$$(x+2)^n$$

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

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

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

>P(3,4)

625

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

>function f(x) &= x^3-2; $&f(x)

$$x^3-2$$

Dengan &= maka fungsi adalah simbolik, dan dapat digunakan di ekpresi simbolik lainnya.

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

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

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

//latex: f(x) = _1^x t^t , dt,

yang tidak dapat dinilai secara simbolis.

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

>function map f(x) := integrate (“x^x”,1,x)

>f(0:0.5:2)

[-0.7834305107120823, -0.4108156482543905, 0, 0.6768632787990813, 2.050446234534731]

Fungsi dapat memiliki nilai default untuk parameter.

>function mylog (x,base=10) := ln(x)/ln(base);

Sekarang fungsi dapat dipanggil dengan menggunakan suatau parameter “base” maupun tidak.

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

```
2
6.7
```

Selain itu, dimungkinkan untuk menggunakan parameter yang ditetapkan.

>mylog(E^2,base=E)

```
2
```

Sering kali, kita ingin menggunakan fungsi untuk vektor di satu tempat, dan untuk elemen individual di tempat lain. Ini tapat terjadi dengan vektor parameter.

>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.

tetapi fungsi ini juga dapat digunakan untuk vektor numerik.

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

```
17
```

Ada juga fungsi yang murni simbolis, 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)

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

$$0$$

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

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

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

Ringkasam:

- &= mendefinisikan fungsi simbolis,
- := mendefinisikan fungsi numerik,
- &&= mendefinisikan fungsi simbolis murni

Memecahkan Ekspresi

Ekspresi dapat diselesaikan secara numerik dan simbolis.

Untuk menyelesaikan ekspresi sederhana dari satu variabel, kita dapat menggunakan fungsi solve(). Perlu

nilai awal untuk memulai pencarian. Secara internal, solve() menggunakan metode secant.

>solve(“x^2-2”,1)

```
1.414213562373095
```

Ini juga berfungsi untuk fungsi simbolik, perhatikan fungsi berikut ini.

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

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

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

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

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

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

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

$$[\,[\,x=\frac{-\sqrt{a^2e^2+(4bc-2abd)e+b^2d^2}-ae+bd}{2b},y=\frac{a\sqrt{a^2e^2-2abde+4bce+b^2d^2}+a^2e-abd+2bc}{2b^2}\,],[\,$$

Sekarang kita mencari titik, di mana polinomialnya adalah 2. Dalam solve(), nilai target default y=0 dapat diubah dengan variabel yang ditetapkan.

Kami menggunakan y=2 dan memeriksa dengan mengevaluasi polinomial pada hasil sebelumnya.

```
>px &= 4*x^8+x7-x^4-x; $&px
```

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

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

```
0.966715594850973
2.000000000000001
```

Memecahkan ekspresi simbolis dalam bentuk simbolis mengembalikan daftar solusi. Kami menggunakan

pemecah simbolik solve() yang disediakan oleh Maxima.

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

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

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

```
>longest sol()
```

```
    -0.6180339887498949         1.618033988749895
```

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

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

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

$$0$$

Memecahkan sistem persamaan secara simbolis dapat dilakukan dengan vektor persamaan dan solver simbo□lis solve(). Hadilnya dalam bentuk 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 melihat variabel global. Namun seringkali kita ingin menggunakan parameter lokal.

dengan a=3.

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

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

```
>solve({{“f”,3}},2,y=0.1)
```

```
-0.7102421508578388
```

Ini juga bekerja dengan ekspresi. Tapi daftar elemen yang ada harus digunakan.

```
>solve({{“x^2+a*x”,a=3}},2,y=0.1)
```

```
0.03297097167558916
```

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.

Eliminasi Fourier adalah analog dari eliminasi Gauss untuk linear (persamaan atau pertidaksamaan). Panggilan fungsi `fourier_elim([eq1, eq2, ...], [var1, var2, ...])' melakukan eliminasi Fourier eliminasi pada pertidaksamaan linear[eq1, eq2, …]’ dengan berkenaan dengan variabel `[var1, var2, …]’; sebagai contoh

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

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

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

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

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

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

>$&fourier_elim([x^2 - 4<0],[x])

$$[-2 < x, x < 2]$$

>$&fourier_elim([x^2 - 9# 0],[x])

$$[-3 < x, x < 3] \vee [3 < x] \vee [x < -3]$$

>$&fourier_elim([x # 10],[x])

$$[x < 10] \vee [10 < x]$$

Ketika himpunan penyelesaiannya adalah kosong maka `emptyset', dan ketika himpunan penyelesaiannya adalah semua bilangan real, maka 'universalset'; sebagai contoh

>$&fourier_elim([minf < x, x < inf],[x])

universalset

>$&fourier_elim([x < 1, x > 1],[x])

emptyset

Untuk persamaan nonlinier, `fourier_elim' mengembalikan sebuah daftar persamaan yang disederhanakan:

>$&fourier_elim([x^3 - 8 > 0],[x])

$$[2 < x, x^2 + 2x + 4 > 0] \vee [x < 2, -x^2 - 2x - 4 > 0]$$

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

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

Alih-alih sebuah daftar pertidaksamaan,`fourier_elim' juga dapat berupa disjungsi atau konjungsi logika:

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

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

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

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

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

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

Fungsi `fourier_elim' mendukung operator pertidaksamaan<,<=, >, >=,#', dan `='.

Kode eliminasi Fourier memiliki sebuah preprocessor yang mengubah beberapa persamaan nonlinier yang melibatkan nilai absolut, minimum,dan fungsi maksimum menjadi linear dalam persamaan. Selain itu,preprocessor menangani beberapa ekspresi yang merupakan hasil kali atau hasil bagi dari suku-suku linier:

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

$$[6 < x, x < 8, y < -11] \vee [8 < x, y < -11] \vee [x < 8, 13 < y] \vee [x = y, 13 < y] \vee [8 < x, x < y, 13 < y] \vee [y < x, 13 < y]$$

>$&fourier_elim([(x+2)/(x-4) <= 2],[x])

$$[x = 10] \vee [10 < x] \vee [x < 4]$$

Bahasa Matriks

Dalam matematika, matriks adalah susunan[1] bilangan, simbol, atau ekspresi yang disusun dalam baris dan kolom sehingga membentuk suatu bangun persegi

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

Matriks 1x2

>a=[1;2]

$$\begin{matrix} 1 \\ 2 \end{matrix}$$

>b=[3,4;5,6]

```
                    3                    4
                    5                    6
```

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

```
Real 3 x 3 matrix

                    1    ...
                    4    ...
                    7    ...
```

Transpose matriks adalah matriks baru yang diperoleh dengan cara menukar elemen-elemen baris menjadi elemen kolom atau sebaliknya.

>a'

```
[1,  2]
```

>b'

```
                    3                    5
                    4                    6
```

>c'

```
Real 3 x 3 matrix

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

Invers matriks adalah matriks baru yang merupakan kebalikan dari matriks asal

>inv(b)

```
    -2.999999999999997       1.999999999999999
     2.499999999999998      -1.499999999999999
```

Perkalian matriks sendiri adalah proses mengalikan setiap elemen baris pada matriks pertama dengan elemen kolom pada matriks kedua.

>b.a

```
                   11
                   17
```

Perkalian dari matriks dengan invers matriks itu sendiri akan menghasilkan matriks identitas

>b.inv(b)

```
    0.9999999999999982    1.77635683940025e-15
                     0                       1
```

Perkalian matriks dan perpangkatan matriks

>b.b

```
                   29                   36
                   45                   56
```

>b^2

```
                    9                   16
                   25                   36
```

>b.b.b

```
                  267                  332
                  415                  516
```

>power(b,3)

```
                  267                  332
                  415                  516
```

Pembagian matriks

>a/a

```
                    1
                    1
```

>a/b

```
    0.3333333333333333                 0.25
                   0.4    0.3333333333333333
```

Perkalian invers matriks dengan matriks lainny

>b\a

```
     0.9999999999999993
    -0.4999999999999994
```

>inv(b).a

```
                     1
    -0.4999999999999996
```

Perkalian skalar

>b*2

```
                     6                     8
                    10                    12
```

>2*b

```
                     6                     8
                    10                    12
```

>[1,2]*2

```
[2,  4]
```

Fungsi Matriks Lainnya

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

>v=1:3; v_v

```
Real 2 x 3 matrix

                     1     ...
                     1     ...
```

>A=random(3,4)

```
Real 3 x 4 matrix

     0.6554163483485531     ...
     0.5250003829714993     ...
     0.2826898577756425     ...
```

>A|1

```
Real 3 x 5 matrix

     0.6554163483485531     ...
     0.5250003829714993     ...
     0.2826898577756425     ...
```

>[v,v]

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

>[v;v]

```
Real 2 x 3 matrix

                     1     ...
                     1     ...
```

>[v',v']

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

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

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

>length(2:10)

```
9
```

>ones(2,2)

```
                     1                     1
                     1                     1
```

>zeros(2,2)

```
                     0                     0
                     0                     0
```

>ones(5)*6

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

>random(1,2)

```
[0.217693385437102,  0.4453627564273003]
```

Berikut adalah fungsi lain yang berguna, yang merestrukturisasi elemen matriks menjadi matriks lain.

>redim(1:9,3,3)

```
Real 3 x 3 matrix

                    1     ...
                    4     ...
                    7     ...
```

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

>rep(1:3,5)

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

>multdup(1:3,3)

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

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

Keduanya tidak memiliki jumlah kolom yang sama, kolom yang lebih pendek akan diisi dengan 0.

>flipx(1:5)

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

>rotleft(1:5)

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

>rotright(1:5)

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

Sebuah fungsi khusus adalah drop(v,i), yang menghilangkan elemen dengan indeks di i dari vektor v

>drop(10:20,3)

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

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

>A=id(3)

```
Real 3 x 3 matrix

                    1     ...
                    0     ...
                    0     ...
```

Vektorisasi

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

>sqrt(1:4)

```
[1,  1.414213562373095,  1.732050807568877,  2]
```

Jadi, kamu dapat dengan mudah membuat tabel nilai. Ini adalah salah satu cara untuk memplot suatu fungsi(alternatifnya menggunakan ekspresi).

>x=3:0.05:6; y=log(x)

```
[1.09861228866811,  1.11514159061932,  1.1314021114911,
1.147402452837541,  1.163150809805681,  1.178654996341646,
1.193922468472434,  1.208960345836975,  1.223775431622115,
1.238374231043268,  1.252762968495367,  1.266947603487324,
1.280933845462064,  1.294727167594399,  1.308332819650178,
1.321755839982319,  1.335001066732339,  1.348073148299692,
1.3609765531356,  1.37371557891303,  1.38629436111989,
1.398716881118447,  1.410986973710261,  1.423108334242606,
1.435084525289322,  1.446918982936324,  1.458615022699516,
```

```
1.470175845100592,  1.481604540924214,  1.492904096178148,
1.504077396776273,  1.515127232962858,  1.526056303495048,
1.536867219599264,  1.547562508716012,  1.558144618046549,
1.568615917913844,  1.57897870494939,  1.589235205116579,
1.599387576580598,  1.609437912434099,  1.619388243287267,
1.629240539730279,  1.638996714675643,  1.64865862558738,
1.658228076603531,  1.667706820558075,  1.677096560907914,
1.686398953570227,  1.695615608675151,  1.704748092238424,
1.713797927758341,  1.722766597741102,  1.731655545158348,
1.740466174840503,  1.749199854809257,  1.757857917552372,
1.766441661243763,  1.774952350911672,  1.783391219557537,
 ... ]
```

Dengan ini dan operator titik dua a:delta:b, vektor nilai fungsi dapat dihasilkan dengan mudah. Pada contoh berikut, kita membangkitkan vektor nilai t[i] dengan spasi 0,1 dari -1 hingga 3. Kemudian kita membangkitkan vektor nilai fungsi.

lateks:s=t^3-t

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

```
[0,  0.1709999999999999,  0.2879999999999999,  0.357,  0.384,  0.375,
0.3360000000000001,  0.2730000000000001,  0.1920000000000001,
0.09900000000000014,  1.387778780781446e-16,  -0.09899999999999987,
-0.1919999999999999,  -0.2729999999999999,  -0.336,  -0.375,  -0.384,
-0.3570000000000001,  -0.2880000000000001,  -0.1710000000000003,
-4.440892098500626e-16,  0.2309999999999997,  0.5279999999999998,
0.897,  1.344000000000001,  1.875000000000001,  2.496000000000002,
3.213000000000004,  4.032000000000004,  4.959000000000006,
6.000000000000005,  7.161000000000006,  8.448000000000008,
9.86700000000001,  11.42400000000001,  13.12500000000001,
14.97600000000002,  16.98300000000003,  19.15200000000003,
21.48900000000003,  24.00000000000004]
```

>shortest (1:7)*(1:7)'

```
     1     2     3     4     5     6     7
     2     4     6     8    10    12    14
     3     6     9    12    15    18    21
     4     8    12    16    20    24    28
     5    10    15    20    25    30    35
     6    12    18    24    30    36    42
     7    14    21    28    35    42    49
```

Perhatikan, bahwa ini sangat berbeda dari produk matriks. Produk matriks dilambangkan dengan titik "." di EMT.

>(1:7).(1:7)'

```
140
```

Secara default, vektor baris dicetak dalam format yang ringkas.

>[6,7,8,9]

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

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

>v:=[2,3]; v.v', %^2

```
13
169
```

Untuk mentranspos matriks kita menggunakan apostrof

>v=3:6; v'

```
                     3
                     4
                     5
                     6
```

>A=[1,2,3,4]; A.v'

```
50
```

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

>v'.v

```
Real 4 x 4 matrix

                     9     ...
                    12     ...
                    15     ...
                    18     ...
```

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

>v.v'

```
86
```

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

>norm(v)^3

```
797.5311906126306
```

Operator dan fungsi mematuhi bahasa matriks Euler.

Berikut ringkasan aturannya.

- Fungsi yang diterapkan ke vektor atau matriks diterapkan ke setiap elemen.
- Operator yang beroperasi pada dua matriks dengan ukuran yang sama diterapkan berpasangan ke elemen matriks.
- jika kedua matriks memiliki dimensi yang berbeda, keduanya diperluas dengan cara yang masuk akal, sehingga memiliki ukuran yang sama.

Misalnya, nilai skalar kali vektor mengalikan nilai dengan setiap elemen vektor. Atau matriks kali vektor (dengan *, bukan.) memperluas vektor ke ukuran matriks dengan menduplikasikan.

Berikut ini adalah kasus sederhana dengan operator^.

>[2,3,6]^2

```
[4,  9,  36]
```

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

>v:=[2,3,6]; v*v'

```
Real 3 x 3 matrix

                    4     ...
                    6     ...
                   12     ...
```

Perhatikan bahwa produk skalar menggunakan produk matriks, bukan *!

>v.v'

```
49
```

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

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

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

```
[4,  9,  36]
49
[4,  13,  49]
```

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

>3:9, 0:2:8

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

>[1,2]|[3,4,5], [1,2]_5

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

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

>A:=[4,5,6,7;8,9,10,11]; A[2,2]

```
9
```

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

>v:=[1,3,5,7]; v[1], A[1]

```
1
[4,  5,  6,  7]
```

Indeks juga bisa menjadi vektor baris dari indeks. : menunjukkan semua indeks.

>v[2:4], A[:,2]

```
[3,  5,  7]
                    5
                    9
```

Bentuk singkat untuk : adalah menghilangkan indeks sepenuhnya.

>A[,2:4]

```
Real 2 x 3 matrix

                    5     ...
                    9     ...
```

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

>A{5}

```
8
```

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

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

```
[4,  5,  6,  7,  8,  9,  10,  11]
[4,  5,  6,  7,  8,  9,  10,  11]
```

Untuk menggunakan matriks untuk tabel, mari kita reset 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 menghasilkan beberapa tabel dari beberapa fungsi sekali□gus.

Dalam contoh berikut, kita menghitung t[j]^i untuk i dari 1 hingga n. Kami mendapatkan matriks, di mana

setiap baris adalah tabel t^i untuk satu i. Yaitu, matriks memiliki elemen lateks: a_{i,j} = t_j^i, j

, i n

Fungsi yang tidak berfungsi untuk input vektor harus "divektorkan". Ini dapat dicapai dengan kata kunci

"peta" dalam definisi fungsi. Kemudian fungsi tersebut akan dievaluasi untuk setiap elemen dari parameter vektor.

Integrasi numerik terintegrasi() hanya berfungsi untuk batas interval skalar. Jadi kita perlu membuat vektor.

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

Kata kunci “peta” membuat vektor fungsi. Fungsinya sekarang akan bekerja untuk vektor bilangan.

>f([1:3])

```
[0,  2.05045,  13.7251]
```

Sub-Matriks dan Matriks-Elemen

Untuk mengakses elemen matriks, gunakan notasi braket.

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

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

Kita dapat mengakses satu baris matriks yang lengkap.

>A[1]

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

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

>v=1:8; v[2]

```
2
```

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

>A[3,]

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

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

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

Kita bahkan dapat menyusun ulang A menggunakan vektor indeks. Tepatnya, kami tidak mengubah A disini, tetapi menghitung versi A yang disusun ulang.

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

Trik indeks bekerja dengan kolom juga.

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

>A[1:2,2:3]

```
            2            3
            5            6
```

Untuk singkatan “:” menunjukkan semua indeks baris atau kolom.

>A[:,2]

```
            2
            5
            8
```

Atau, biarkan indeks pertama kosong.

>A[,1:3]

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

Kita juga bisa mendapatkan baris terakhir dari A.

>A[3]

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

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

>A[2,3]=9

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

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

>A[1:2,1:2]=[4,5;6,7]

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

Selain itu, beberapa jalan pintas diperbolehkan.

>A[1:2,1:2]=-1

```
           -1           -1            3
           -1           -1            9
            7            8            9
```

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

>A[5]

```
Row index 5 out of bounds!
Error in:
A[5] ...
    ^
```

Menyortir dan Mengacak

Fungsi sort() mengurutkan vektor baris.

>sort([2,9,5,7,3,1])

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

Seringkali perlu untuk mengetahui indeks dari vektor yang diurutkan dalam vektor aslinya. Ini dapat digu□nakan untuk menyusun ulang vektor lain dengan cara yang sama.

Mari kita mengacak vektor.

>v=shuffle(1:8)

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

Indeks berisi urutan yang tepat dari v.

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

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

>s=[“d”,“f”,“c”,“b”,“aa”,“g”]

```
d
f
c
b
aa
g
```

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

```
aa
b
c
d
f
g
```

```
    endfunction
```

Aljabar Linier

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

Untuk sistem linier Ax=b,dengan A adalah matriks koefisien, x adalah vektor solusi yang ingin kita cari dan b adalah vektor hasil yang diberikan.

Anda dapat menggunakan algoritma Gauss, matriks invers atau kecocokan linier.

Operator Aenggunakan versi algoritma Gauss.

Operator backslash digunakan untuk menyelesaikan sistem persamaan linier ini. Ketika menulis A perangkat lunak akan menghitung solusi yang memenuhi persamaan Ax=b.

Operator ini secara otomatis menggunakan algoritma eliminasi Gauss atau metode numerik serupa untuk menemukan solusi.

>A=[5,6;7,8]; b= [4;3]; A\b

```
  -7
 6.5
```

Untuk contoh lain, kami membuat matriks 100x100 dan jumlah barisnya. Kemudian kita selesaikan Ax=b

menggunakan matriks invers. Kami mengukur kesalahan sebagai deviasi maksimal semua elemen dari 1, yang tentu saja merupakan solusi yang benar.

>A=normal(100,100); b=sum(A); longest totalmax(abs(inv(A).b-1))

```
4.585221091701897e-14
```

Jika sistem tidak memiliki solusi, kecocokan linier meminimalkan norma

kesalahan Ax-b.

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

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

Determinan matriks ini adalah ()

> det(A)

```
-34
```

Matriks Simbolik

Maxima memiliki matriks simbolis. Tentu saja, Maxima dapat digunakan untuk masalah aljabar linier sederhana seperti itu.

Kita dapat mendefinisikan matriks untuk Euler dan Maxima dengan &:=, dan kemudian menggunakannya dalam ekspresi simbolis. Bentuk […] biasa 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(%)

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

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

>$&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

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

dengan

`det(A)` menghitung determinan matriks A.

`factor(% )` memberikan faktor-faktor dari determinan.

`invert(A)` memberikan invers dari matriks A.

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

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

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

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

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

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

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

Untuk mengekstrak vektor eigen tertentu perlu 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 ekspresi simbolik lainnya.

>A(a=6,b=7)

```
1              6
7              2
```

Dalam ekspresi simbolik, gunakan dengan.

>$&A with [a=6,b=7]

$$\begin{pmatrix} 1 & 6 \\ 7 & 2 \end{pmatrix}$$

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

>$&A[1]

$$[1, a]$$

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

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

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

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

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

>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 fungsi xinv() yang kuat, yang membuat upaya lebih besar dan mendapatkan hasil yang lebih tepat.

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

>longest B.xinv(B)

```
                1                    0
                0                    1
```

Misalnya. nilai eigen dari A dapat dihitung secara numerik.

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

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

Atau secara simbolis. Lihat tutorial tentang Maxima untuk detailnya.

>$&eigenvalues((**A?**))

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

Nilai Numerik dalam Ekspresi simbolis

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

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

```
            1       3.14159
            4             5
```

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

>$&A

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

Untuk menghindarinya, 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 floating besar dengan 32 digit. Namun, evaluasinya jauh lebih lambat.

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

$$1.4142135623730950488016887242097_B \times 10^0$$

$$1.414213562373095$$

Ketepatan angka floating point besar dapat diubah

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

```
        3.14159265358979323846264338327950288419716939937510582097494\
459230781640628620899862803482534211706 8b0
```

Variabel numerik dapat digunakan dalam ekspresi simbolis apa pun 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 perhitungan 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 4000 (katakanlah dalam dolar).

>M=4000

```
4000
```

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

>M*1.03

```
4120
```

Euler akan memahami sintaks berikut juga.

>M+M*3%

```
4120
```

Tetapi lebih mudah menggunakan faktornya.

>q=1+3%, M*q

```
1.03
4120
```

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

>M*q^10

```
5375.66551738
```

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

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

```
    5375.67
```

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

>“Starting from” + M + “$ you get” + round(M*q^10,2) + “$.”

```
Starting from 4000$ you get 5375.67$.
```

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

>M*q^(0:10)

```
Real 1 x 11 matrix

    4000.00     4120.00     4243.60     4370.91     ...
```

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

>short 0:10

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

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

>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 sampai q^10. Ini dikalikan dengan M, dan kami mendapatkan vektor nilai.

>VM=M*q^(0:10)

```
Real 1 x 11 matrix

    4000.00     4120.00     4243.60     4370.91     ...
```

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

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

Mari kita bandingkan dua hasil, dengan dan tanpa pembulatan.

>longest oneyear(1234.57), longest 1234.57*q

```
               1271.61
             1271.6071
```

Sekarang tidak ada rumus sederhana untuk tahun ke-n, dan kita harus mengulang selama bertahun-tahun.

Euler memberikan banyak solusi untuk ini.

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

>VMr=iterate(“oneyear”,4000,10)

```
Real 1 x 11 matrix

    4000.00     4120.00     4243.60     4370.91     ...
```

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

>VMr’

```
    4000.00
    4120.00
    4243.60
    4370.91
    4502.04
    4637.10
    4776.21
    4919.50
```

```
    5067.09
    5219.10
    5375.67
```

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

>VMr[2], VMr[1:3]

```
    4120.00
    4000.00     4120.00     4243.60
```

Anehnya, kita juga bisa menggunakan vektor indeks. Ingat bahwa 1:3 menghasilkan vektor [1,2,3].

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

>VMr[-2], VM[-2]

```
    5219.10
    5219.09
```

Perbedaannya sangat kecil.

Memecahkan Persamaan

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

>function onepay (M) := M*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”,4000,10)

```
Real 1 x 11 matrix

    4000.00     4320.00     4649.60     4989.09     ...
```

Bagaimana jika kita menghapus jumlah yang sama setiap tahun?

>R=-200; iterate(“onepay”,4000,10)

```
Real 1 x 11 matrix

    4000.00     3920.00     3837.60     3752.73     ...
```

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

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

>VMR=iterate(“onepay”,4000,50)

```
Real 1 x 51 matrix

    4000.00     3920.00     3837.60     3752.73     ...
```

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

>min(nonzeros(VMR<0))

```
      32.00
```

Alasan untuk ini adalah bahwa bukan nol(VKR<0) mengembalikan vektor indeks i, di mana VKR[i]<0, dan min menghitung indeks minimal. Karena vektor selalu dimulai dengan indeks 1, jawabannya adalah 31 tahun.

Fungsi iterate() memiliki satu trik lagi. Itu bisa mengambil kondisi akhir sebagai argumen. Kemudian akan mengembalikan nilai dan jumlah iterasi.

>{x,n}=iterate(“onepay”,4000,till=“x<0”); x, n

```
      -0.21
      31.00
```

Mari kita coba menjawab pertanyaan yang lebih ambigu. Asumsikan kita tahu bahwa nilainya adalah 0 setelah 50 tahun. Apa yang akan menjadi tingkat bunga?

Ini adalah pertanyaan yang hanya bisa dijawab dengan angka. Di bawah ini, kita akan mendapatkan formula yang diperlukan. Kemudian Anda akan melihat bahwa tidak ada formula yang mudah untuk tingkat bunga.

Tapi untuk saat ini, kami bertujuan untuk solusi numerik.

Langkah pertama adalah mendefinisikan fungsi yang melakukan iterasi sebanyak n kali. Kami menam□bahkan semua parameter ke fungsi ini.

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

Iterasinya sama seperti di atas.

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

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

Mari kita coba tes.

>f(4000,-200,3,31)

```
-0.21
```

Sekarang kita bisa menyelesaikan masalah kita.

>solve("f(4000,-200,x,50)",3)

```
 4.43
```

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

Kita dapat menggunakan fungsi yang sama untuk menyelesaikan pertanyaan berikut:

Berapa banyak yang dapat kita keluarkan per tahun sehingga modal awal habis setelah 20 tahun dengan asumsi tingkat bunga 3% per tahun.

>solve("f(4000,x,3,20)",-200)

```
-268.86
```

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

Solusi Simbolik untuk Masalah Suku Bunga

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

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

$$R + qM$$

Kita sekarang dapat mengulangi ini.

>$&op(op(op(op(M)))), $&expand(%)

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

$$q^3 R + q^2 R + qR + R + q^4 M$$

Kami melihat sebuah pola. Setelah n periode yang kita miliki

Rumusnya adalah rumus untuk jumlah geometri, yang diketahui Maxima.

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

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

Ini agak rumit. Jumlahnya dievaluasi dengan bendera "simpsum" untuk menguranginya menjadi hasil bagi.

Mari kita membuat fungsi untuk ini.

Rumusnya adalah rumus untuk jumlah geometri, yang diketahui Maxima.

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

$$\frac{100\,((\frac{P}{100} + 1)^n - 1)\,R}{P} + M\,(\frac{P}{100} + 1)^n$$

Fungsi tersebut melakukan hal yang sama seperti fungsi f kita sebelumnya. Tapi itu lebih efektif.

>longest f(4000,-200,3,31), longest fs(4000,-200,3,31)

```
-0.2142542009293322
-0.2142542009369208
```

Kita sekarang dapat menggunakannya untuk menanyakan waktu n. Kapan modal kita habis? Dugaan awal kami adalah 30 tahun.

fungsi untuk ini.

Rumusnya adalah rumus untuk jumlah geometri, yang diketahui Maxima.

>solve("fs(4000,-330,3,x)",30)

```
15.29
```

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

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

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

Rumus untuk ini jelas.

>equ &= fs(M,R,P,n)=Mn; $&equ

$$\frac{100\,((\frac{P}{100}+1)^n-1)\,R}{P}+M\,(\frac{P}{100}+1)^n=Mn$$

Biasanya rumus ini diberikan dalam bentuk

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

$$\frac{((i+1)^n-1)\,R}{i}+(i+1)^n\,M=Mn$$

Kita dapat memecahkan tingkat R secara simbolis.

>$&solve(equ,R)

$$[\,R=\frac{i\,Mn-i\,(i+1)^n\,M}{(i+1)^n-1}\,]$$

Seperti yang Anda lihat dari rumus, fungsi ini mengembalikan kesalahan titik mengambang untuk i=0.

Euler tetap merencanakannya.

Tentu saja, kami memiliki batas berikut.

>$&limit(R(4000,0,x,10),x,0)

$$\lim_{x\to 0} R\,(4000,\,0,\,x,\,10)$$

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

Persamaan juga dapat diselesaikan untuk n. Kelihatannya lebih bagus, jika kita menerapkan beberapa penyederhanaan untuk itu.

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

$$[\,n=\frac{\log(\frac{R+i\,Mn}{R+i\,M})}{\log(i+1)}\,]$$