

## **Aplikasi Komputer**

---

Nama: Davina Safa Felisa

NIM: 23030630032

Kelas: Matematika E

## EMT untuk Perhitungan Aljabar

---

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

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

### Contoh pertama

---

Menyederhanakan bentuk aljabar:

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

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

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

Menjabarkan:

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

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

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

**Baris Perintah**

---

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

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

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

16.7551608191

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

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

50.2654824574  
100.530964915

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

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

0.540302305868

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

0.857553215846

Jika dua garis terhubung dengan "..." kedua garis 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 menyebarkan perintah panjang 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.

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

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

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

81

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

```
1.5  
1.41666666667  
1.41421568627  
1.41421356237  
1.41421356237
```

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

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

```
1.41421356237
```

Struktur bersyarat juga berfungsi.

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

Thought so!

Saat Anda menjalankan perintah, kursor dapat berada di posisi mana pun di baris perintah. Anda dapat kembali ke perintah sebelumnya atau melompat ke perintah berikutnya dengan tombol panah. Atau Anda dapat mengklik 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. 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(10°)/cos(20°))
```

0.429875017772

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

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

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

2.5

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

---

Sintaks Dasar

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

Untuk menyetel variabel, gunakan "=" atau ":=". Demi kejelasan, pengantar ini menggunakan bentuk yang terakhir. Spasi tidak masalah. Tapi ruang antara perintah diharapkan.

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

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

30.65625

EMT menggunakan sintaks pemrograman untuk ekspresi. Memasuki

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

Anda harus mengatur tanda kurung yang benar dan menggunakan / untuk pecahan. Perhatikan tanda kurung yang disorot untuk bantuan. Perhatikan bahwa konstanta Euler e diberi nama 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 harus memasukkannya dalam bentuk baris.

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

23.2671801626

Letakkan tanda kurung dengan hati-hati di sekitar sub-ekspresi yang perlu dihitung terlebih dahulu. EMT membantu Anda dengan menyorot ekspresi 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.

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

0.47619047619  
10/21

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

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

14.4978445072

Operator numerik Euler meliputi

```
+ unary atau operator plus
- unary atau operator minus
*, /
. 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. Untuk log.

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

2  
0.5

```
>sin(30°)
```

0.5

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

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

2.41785163923e+24  
4096  
2.41785163923e+24

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

```
>longest 1/3
```

```
0.3333333333333333
```

Representasi ganda internal membutuhkan 8 byte.

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

```
1.01010101010101010101010101010101010101010101010101010101010101*2^-2
```

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

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

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.

```
>"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<sup>2</sup>.

Fungsi print juga mengonversi angka menjadi string. Ini dapat mengambil sejumlah angka dan jumlah tempat (0 untuk keluaran padat), dan secara optimal satu unit.

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

Golden Ratio : 1.61803

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

```
>none
```

Untuk mengonversi string menjadi angka, cukup evaluasi saja. Ini juga berfungsi 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 dilambangkan dengan [none]. Vektor string dapat digabungkan.

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

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

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

String Unicode dapat digabungkan seperti string lainnya.

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

= 45°

I

Dalam komentar, entitas yang sama seperti , dll dapat digunakan. Ini mungkin alternatif cepat untuk Lateks. (Lebih detail di komentar 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 =.
```

Dimungkinkan juga untuk menggunakan entitas numerik.

```
>u"\u00d6hnliches"
```

Ähnliches

## Nilai Boolean

---

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

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

0  
1

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

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

Operator Boolean mematuhi aturan bahasa matriks.

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

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

Anda dapat menggunakan fungsi bukan nol() untuk mengekstrak elemen tertentu dari vektor. Dalam contoh, kami menggunakan isprime bersyarat(n).

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

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

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

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

## Format Keluaran

---

Format output default EMT mencetak 12 digit. Untuk memastikan bahwa kami melihat default, kami mengatur ulang format.

```
>defformat; pi
```

```
3.14159265359
```

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

```
>longest pi
```

```
3.141592653589793
```

Berikut adalah representasi heksadesimal internal dari bilangan ganda.

```
>printhex(pi)
```

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

Format output dapat diubah secara permanen dengan perintah format.

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

```
0.33333  
3.14159  
0.84147
```

Standarnya adalah format (12).

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

```
0.333333333333
```

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

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

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

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

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

3.1416

Fungsi "format terpanjang" mengatur format skalar juga.

```
>longestformat; pi
```

3.141592653589793

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

```
format terpendek format pendek format panjang, format terpanjang  
format(panjang,digit) format baik(panjang)  
fracformat (panjang)  
mengubah bentuk
```

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

Tetapi format output EMT dapat diatur dengan cara yang fleksibel.

```
>longestformat; pi,
```

3.141592653589793

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

3.14159

The default is deformat().

```
>deformat; // default
```

Ada operator pendek yang hanya mencetak satu nilai. Operator "terpanjang" akan mencetak semua digit angka yang valid.

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

4.934802200544679

Ada juga operator pendek untuk mencetak hasil dalam format pecahan. Kami sudah menggunakan di atas.

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

25/12

Karena format internal menggunakan cara biner untuk menyimpan angka, nilai 0,1 tidak akan direpresentasikan dengan tepat. Kesalahan bertambah sedikit, seperti yang Anda lihat dalam perhitungan berikut.

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

-1.110223024625157e-16

Tetapi dengan "format panjang" default Anda tidak akan melihat ini. Untuk kenyamanan, output dari angka yang sangat kecil adalah 0.

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

0

String atau nama dapat digunakan untuk menyimpan ekspresi matematika, yang dapat dievaluasi oleh EMT. Untuk ini, gunakan tanda kurung setelah ekspresi. Jika Anda bermaksud menggunakan string sebagai ekspresi, gunakan konvensi untuk menamainya "fx" atau "fxy" dll. Ekspresi lebih diutamakan daripada fungsi.

Variabel global dapat digunakan dalam evaluasi.

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

12.56637061435917

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

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

-0.919535764538

Perhatikan bahwa ekspresi akan selalu menggunakan variabel global, bahkan jika ada variabel dalam fungsi dengan nama yang sama. (Jika tidak, evaluasi ekspresi dalam fungsi dapat memberikan 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

Untuk referensi, kami berkomentar bahwa koleksi panggilan (dibahas di tempat lain) dapat berisi ekspresi. Jadi kita bisa membuat contoh di atas sebagai berikut.

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

45

Ekspresi dalam x sering digunakan seperti fungsi.

Perhatikan bahwa mendefinisikan fungsi dengan nama yang sama seperti ekspresi simbolik global menghapus variabel ini untuk menghindari kebingungan antara ekspresi simbolik dan fungsi.

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

12

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

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

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

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) ...".

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

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

41

Ini memungkinkan untuk memanipulasi ekspresi dalam variabel lain untuk fungsi EMT yang membutuhkan ekspresi dalam "x".

Cara paling dasar untuk mendefinisikan fungsi sederhana adalah dengan menyimpan rumusnya dalam ekspresi simbolis atau numerik. Jika variabel utama adalah x, ekspresi 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 ditegaskan.

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

-0.425

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

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

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.

```
>$&44!
```

```
2658271574788448768043625811014615890319638528000000000
```

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

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

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

```
2481256778
```

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

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

2481256778

Untuk mempelajari lebih lanjut tentang fungsi tertentu klik dua kali 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.

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

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

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

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

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

120

Dengan begitu Anda dapat menggunakan solusi persamaan dalam persamaan lain.

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

Seperti yang akan Anda lihat pada contoh sebelumnya dan berikut, jika Anda telah menginstal LaTeX, Anda dapat mencetak ekspresi simbolis dengan Lateks. Jika tidak, perintah berikut akan mengeluarkan pesan kesalahan.

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

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

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

Ekspresi simbolik diuraikan oleh Euler. Jika Anda membutuhkan sintaks yang kompleks dalam satu ekspresi, Anda dapat menyertakan ekspresi dalam "...". Untuk menggunakan lebih dari ekspresi sederhana adalah mungkin, tetapi sangat tidak disarankan.

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

25

Untuk kelengkapan, kami menyatakan bahwa ekspresi simbolik dapat digunakan dalam program, tetapi perlu diapit 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
```

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

Sekali lagi, % mengacu pada hasil sebelumnya.

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

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

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

Ekspresi simbolik dapat digunakan dalam ekspresi simbolik lainnya.

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

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

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

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

2432902008176640000

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

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

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

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

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

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

```
2  
g
```

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

$$\begin{matrix} 3 & x \\ x & E \end{matrix}$$

$$x^3 e^x$$

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

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

$$\begin{matrix} 5 \\ 125 E \end{matrix}$$

$$125 e^5$$

$$18551.64488782208$$

```
>fx(5)
```

$$18551.6448878$$

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

$$\begin{array}{r} 10 \\ 1000 \text{ E} \\ - 125 \text{ E} \\ \hline 5 \end{array}$$

2.20079141499189e+7

Untuk evaluasi 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(%)
```

$$\begin{array}{r} 10 \\ 1000 \text{ E} \\ - 125 \text{ E} \\ \hline 5 \end{array}$$

2.20079141499189e+7

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

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

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

```
>tex(fx)
```

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

Ekspresi simbolik dapat dievaluasi seperti ekspresi numerik.

```
>fx(0.5)
```

$$0.206090158838$$

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

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

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

Penugasan juga bisa bersifat simbolis.

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

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

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

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

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

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

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

1.56155281281

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

```
>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 simbolis tertentu, seseorang dapat menggunakan "dengan" dan indeks.

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

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

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

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

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

2

Ekspresi 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)")

```
>$& 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 EMT, fungsi adalah program yang didefinisikan dengan perintah "fungsi". Ini bisa berupa fungsi satu baris atau fungsi multibaris.

Fungsi satu baris dapat berupa numerik atau simbolis. Fungsi satu baris numerik didefinisikan oleh ":=".

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

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

```
>f(2)
```

4.472135955

Fungsi ini akan bekerja untuk vektor juga, dengan mematuhi bahasa matriks Euler, karena ekspresi yang digunakan dalam fungsi divektorkan.

```
>f(0:0.1:1)
```

```
[0, 0.100499, 0.203961, 0.313209, 0.430813, 0.559017, 0.699714,
0.854459, 1.0245, 1.21083, 1.41421]
```

Fungsi dapat diplot. Alih-alih ekspresi, kita hanya perlu memberikan nama fungsi.

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

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

0.786151377757

Secara default, jika Anda perlu menimpa fungsi bawaan, Anda harus menambahkan kata kunci "menimpa". Menimpa fungsi bawaan berbahaya dan dapat menyebabkan masalah untuk fungsi lain tergantung pada fungsi tersebut.

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

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

0.707106781187

Lebih baik kita menghapus redefinisi dosa ini.

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

0.707106781187

Fungsi numerik dapat memiliki parameter default.

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

Menghilangkan parameter ini menggunakan nilai default.

```
>f(4)
```

16

Menyetelnya akan menimpa nilai default.

```
>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 harus global. Fungsi satu baris dapat melihat variabel global.

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

24

Tetapi parameter yang ditetapkan menimpa 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 dalam Euler dan Maxima, dan bekerja di kedua dunia. Ekspresi yang mendefinisikan dijalankan melalui Maxima sebelum definisi.

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

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

Fungsi simbolik dapat digunakan dalam ekspresi simbolik.

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

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

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

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 atau ekspresi simbolis 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

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)
```

0.703467422498

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

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

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

$$(x + 2)^n$$

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

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

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

625

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

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

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

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

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

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

```
>${&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 `&=` fungsinya simbolis, dan dapat digunakan dalam ekspresi simbolik lainnya.

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

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

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

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

yang tidak dapat dinilai secara simbolis.

Jika kita mendefinisikan kembali fungsi dengan kata kunci "peta" 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)
```

```
[-0.783431, -0.410816, 0, 0.676863, 2.05045]
```

Fungsi dapat memiliki nilai default untuk parameter.

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

Sekarang fungsi dapat dipanggil dengan atau tanpa parameter "basis".

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

2  
6.7

Selain itu, dimungkinkan untuk menggunakan parameter yang ditetapkan.

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

2

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

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

$$y^2 - x y + y + x^2$$

Fungsi simbolik seperti itu dapat digunakan untuk variabel simbolik.

Tetapi fungsi tersebut juga dapat digunakan untuk vektor numerik.

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

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

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

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

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

0

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

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

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

Untuk meringkas

- $\&=$  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)
```

1.41421356237

Ini juga berfungsi untuk ekspresi simbolis. Ambil fungsi berikut.

```
>$&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)
```

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

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

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

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

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

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.

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

0.966715594851

2

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
```

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

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

```
>longest sol()
```

```
-0.6180339887498949 1.618033988749895
```

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

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

0

Memecahkan sistem persamaan secara simbolis dapat dilakukan dengan vektor persamaan dan solver simbolis solve(). Jawabannya adalah daftar 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 melihat variabel global. Namun seringkali kita ingin menggunakan parameter lokal.

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

dengan a=3.

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

Salah satu cara untuk meneruskan parameter tambahan ke f() adalah dengan menggunakan daftar dengan nama fungsi dan parameter (sebaliknya adalah parameter titik koma).

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

$$2.54116291558$$

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

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

2.54116291558

## Menyelesaikan Pertidaksamaan

---

Untuk menyelesaikan pertidaksamaan, EMT tidak akan dapat 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
```

$$\text{universal set}$$

```
> $&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}([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) \text{ and } (x - y > 8),[x,y])
```

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

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

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

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

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

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

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

**Bahasa Matriks**

---

Dokumentasi inti EMT berisi diskusi terperinci tentang bahasa matriks Euler.

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

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

1	2
3	4

Produk matriks dilambangkan dengan titik.

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

3
4

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

[3,	4]
-----	----

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

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

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

```
11  
25
```

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

```
1 0  
0 1
```

Poin utama dari bahasa matriks adalah bahwa semua fungsi dan operator bekerja elemen untuk elemen.

```
>A.A
```

```
7 10  
15 22
```

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

1	4
9	16

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

37	54
81	118

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

37	54
81	118

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

1	1
1	1

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

```
0.333333    0.666667  
0.75          1
```

```
>A\b // hasil kali 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
```

$$\begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}$$

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

$$\begin{matrix} 1 & 4 \\ 9 & 16 \end{matrix}$$

Ini bukan produk matriks, tetapi perkalian elemen demi elemen. Hal yang sama berlaku untuk vektor.

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

$$\begin{matrix} 9 \\ 16 \end{matrix}$$

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

```
>2*A
```

$$\begin{matrix} 2 & 4 \\ 6 & 8 \end{matrix}$$

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

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

$$\begin{matrix} 1 & 4 \\ 3 & 8 \end{matrix}$$

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

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

$$\begin{matrix} 2 & 6 \\ 6 & 12 \end{matrix}$$

Seseorang dapat 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)
```

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

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

1	4
3	8

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

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

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

Sekali lagi, ingat bahwa ini bukan produk matriks!

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

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

55

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

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

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

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

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

5

Euler memiliki operator perbandingan, seperti "`==`", yang memeriksa kesetaraan. Kami mendapatkan vektor 0 dan 1, di mana 1 berarti benar.

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

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

Dari vektor seperti itu, "bukan nol" memilih elemen bukan nol.

Dalam hal ini, kami mendapatkan indeks semua elemen 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, yaitu 5 modulo 11 dan 3 modulo 13.

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

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

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

Kita dapat memeriksa keutamaan. Mari kita cari tahu, berapa banyak kuadrat ditambah 1 adalah bilangan prima.

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

112

Fungsi bukan nol() hanya berfungsi untuk vektor. Untuk matriks, ada mnonzeros().

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

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

Ini mengembalikan indeks elemen, yang bukan nol.

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

1	4
2	1
2	2
3	2

Indeks ini dapat digunakan untuk mengatur elemen ke beberapa nilai.

```
>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 ke entri dari beberapa matriks lainnya.

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

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

Dan dimungkinkan untuk mendapatkan elemen dalam vektor.

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

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

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

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

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

Kita dapat menggunakan ini untuk mengekstrak nilai maksimal di setiap baris.

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

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

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

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

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

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

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

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

## Fungsi Matriks Lainnya (Membangun Matriks)

---

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
```

1	2	3
1	2	3

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

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

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

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

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

```
>A|1
```

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

Dimungkinkan untuk membuat matriks vektor baris dan kolom.

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

1	2	3
1	2	3

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

1	1
2	2
3	3

Tujuan utama dari ini adalah untuk menafsirkan vektor ekspresi untuk vektor kolom.

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

1	1
2	4
3	9

Untuk mendapatkan ukuran A, kita dapat menggunakan fungsi berikut.

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

2	
4	
[2,	4]
4	

Untuk vektor, ada panjang().

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

Ada banyak fungsi lain, yang menghasilkan matriks.

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

1	1
1	1

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

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

[6, 6, 6, 6, 6]

Juga matriks bilangan acak dapat dihasilkan dengan acak (distribusi seragam) atau normal (distribusi Gau).

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

0.66566	0.831835
0.977	0.544258

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

```
>redim(1:9,3,3) // menyusun elemen 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 n kali.

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

Let us test.

```
>rep(1:3,5)
```

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

Fungsi multdup() menduplikasi elemen vektor.

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

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

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

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

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

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

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

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

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]
```

Perhatikan bahwa vektor `i` di `drop(v,i)` mengacu pada indeks elemen di `v`, bukan nilai elemen. Jika Anda ingin menghapus elemen, Anda harus menemukan elemennya terlebih dahulu. Fungsi `indexof(v,x)` dapat digunakan untuk mencari elemen `x` dalam vektor terurut `v`.

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

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

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

```
>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 kami tidak mengubah matriks A. Kami mendapatkan matriks baru sebagai hasil dari setdiag().

Berikut adalah fungsi, yang mengembalikan matriks tri-diagonal.

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

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

Diagonal suatu matriks juga dapat diekstraksi dari matriks tersebut. Untuk mendemonstrasikan ini, kami merestrukturisasi vektor 1:9 menjadi matriks 3x3.

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

1	2	3
4	5	6
7	8	9

Sekarang kita dapat mengekstrak diagonal.

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

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

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

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

$$\begin{matrix} 1 & 2 & 3 \\ \frac{4}{5} & 1 & \frac{6}{5} \\ \frac{7}{9} & \frac{8}{9} & 1 \end{matrix}$$

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(1:3)
```

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

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

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

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 1. Kemudian kita membangkitkan vektor nilai fungsi

$$s = t^3 - t$$

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

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

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

Misalnya, vektor kolom dikalikan vektor baris menjadi matriks, jika operator diterapkan. Berikut ini,  $v'$  adalah vektor yang ditransposisikan (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 sangat berbeda dari produk matriks. Produk matriks dilambangkan dengan titik “.” di 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 . menunjukkan perkalian matriks, dan A' menunjukkan transpos. Matriks 1x1 dapat digunakan seperti bilangan real.

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

5  
25

Untuk mentranspos matriks kita menggunakan apostrof.

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

```
1  
2  
3  
4
```

Jadi kita dapat menghitung matriks A kali vektor b.

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

```
30  
70
```

Perhatikan bahwa v masih merupakan vektor baris. Jadi  $v' \cdot v$  berbeda dari  $v \cdot v'$ .

```
>v'.v
```

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

$v \cdot v'$  menghitung norma v kuadrat untuk vektor baris v. Hasilnya adalah vektor  $1 \times 1$ , yang bekerja seperti bilangan real.

```
>v.v'
```

30

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

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

30

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 menduplikasinya.

Berikut ini adalah kasus sederhana dengan operator ^.

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

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

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

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

1	2	3
2	4	6
3	6	9

Perhatikan bahwa produk skalar menggunakan produk matriks, bukan \*!

```
>v.v'
```

14

Ada banyak fungsi 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)
```

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

Operator : menghasilkan vektor baris 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 ada operator "|" dan "\_" .

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

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

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

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

6

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

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

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

Indeks juga bisa menjadi 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 matriks dapat diakses seolah-olah mereka adalah vektor.

```
>A{4}
```

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

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

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

Untuk menggunakan matriks untuk tabel, mari kita reset ke format default, dan menghitung tabel nilai sinus dan 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 sekaligus.

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

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

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([1:5])
```

```
[0, 2.05045, 13.7251, 113.336, 1241.03]
```

## Sub-Matriks dan Matriks-Elemen

---

Untuk mengakses elemen matriks, gunakan notasi braket.

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

1	2	3
4	5	6
7	8	9
5		

Kita dapat mengakses satu baris matriks yang lengkap.

```
>A[2]
```

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

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

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

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

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

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

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

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, kami tidak mengubah A di sini, tetapi menghitung versi A yang disusun ulang.

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

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

2	3
5	6
8	9

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

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

3
6
9

Atau, biarkan indeks pertama kosong.

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

2	3
5	6
8	9

Kita juga bisa mendapatkan baris terakhir dari A.

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

[7, 8, 9]

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

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

4	2	3
4	5	6
7	8	9

Kami juga dapat menetapkan nilai ke baris A.

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

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

Kami bahkan dapat menetapkan 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 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[4]
```

```
Row index 4 out of bounds!
```

```
Error in:
```

```
A[4] ...
```

```
^
```

## Menyortir dan Mengacak

---

Fungsi `sort()` mengurutkan vektor baris.

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

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

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

Mari kita mengocok vektor.

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

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

Indeks berisi urutan yang tepat dari `v`.

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

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

Ini bekerja untuk vektor string juga.

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

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

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

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

Seperti yang Anda lihat, posisi entri ganda agak acak.

```
>ind
```

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

Fungsi unik mengembalikan daftar elemen unik vektor yang diurutkan.

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

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

Ini bekerja untuk vektor string juga.

```
>unique(s)
```

```
a  
aa  
d  
e
```

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

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

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

```
-4  
4.5
```

Untuk contoh lain, kami membuat matriks 200x200 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(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]
```

$$\begin{array}{ccc} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{array}$$

Determinan matriks ini adalah 0.

```
>det(A)
```

0

## 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 menggunakan 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(%)
```

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

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

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

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

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

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

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

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

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

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=4,b=5)
```

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

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 dengan matriks numerik.

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

$$[1, a]$$

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

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

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

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

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

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

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

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

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

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

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

$$\begin{array}{cc} -2 & 1 \\ 1.5 & -0.5 \end{array}$$

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)
```

$$\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}$$

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)
```

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

**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 "mxmlset(variable)".

```
>mxmlset(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.

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

```
1.414213562373095
```

Ketepatan angka floating point besar dapat diubah.

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

```
3.14159265358979323846264338327950288419716939937510582097494\  
4592307816406286208998628034825342117068b0
```

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 5000 (katakanlah dalam dolar).

```
>K=5000
```

5000

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

```
>K*1.03
```

5150

Euler akan memahami sintaks berikut juga.

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

5150

Tetapi lebih mudah menggunakan faktornya

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

```
1.03  
5150
```

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

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

```
6719.58189672
```

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

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

```
6719.58
```

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

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

Starting from 5000\$ you get 6719.58\$.

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

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

Real 1 x 11 matrix

5000.00      5150.00      5304.50      5463.64      ...

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

```
>short 0:10
```

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

Kemudian semua operator dan fungsi dalam Euler dapat diterapkan pada elemen vektor untuk 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  $q^0$  sampai  $q^{10}$ . Ini dikalikan dengan K, dan kami mendapatkan vektor nilai.

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

Tentu saja, cara realistik 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 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.

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

Real 1 x 11 matrix

5000.00      5150.00      5304.50      5463.64      ...

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

```
>VKr'
```

5000.00  
5150.00  
5304.50  
5463.64  
5627.55  
5796.38  
5970.27  
6149.38  
6333.86  
6523.88  
6719.60

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

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

```
5150.00  
5000.00      5150.00      5304.50
```

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

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

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

```
6719.60  
6719.58
```

Perbedaannya sangat kecil.

## Memecahkan Persamaan

---

Sekarang kita mengambil 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 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.

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

Real 1 x 51 matrix

```
5000.00      4950.00      4898.50      4845.45      ...
```

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

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

48.00

Alasan untuk ini adalah bahwa 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 47 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",5000,till="x<0"); x, n,
```

```
-19.83  
47.00
```

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 menambahkan semua parameter ke fungsi ini.

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

Iterasinya sama seperti di atas

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

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

-19.83

Sekarang kita bisa menyelesaikan masalah kita.

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

3.15

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(5000,x,3,20)",-200)
```

-336.08

Perhatikan bahwa Anda tidak dapat menyelesaikan 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(K) &= K*q+R; \$&op(K)
```

$$R + q K$$

Kita sekarang dapat mengulangi ini.

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

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

Kami melihat sebuah pola. Setelah n periode yang kita miliki

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

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.

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

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

```
-19.82504734650985  
-19.82504734652684
```

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

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

```
20.51
```

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

Kita juga dapat menggunakan sisi simbolis Euler untuk menghitung 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(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 tingkat R secara simbolis.

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

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

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

Tentu saja, kami memiliki batasan berikut.

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

$$\lim_{x \rightarrow 0} R(5000, 0, x, 10)$$

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
```

$$n = \frac{\log\left(\frac{R+iKn}{R+iK}\right)}{\log(i+1)}$$

Menggambar Grafik 2D dengan EMT

---

Notebook ini menjelaskan tentang cara menggambar berbagai kurva dan grafik 2D dengan software EMT. EMT menyediakan fungsi `plot2d()` untuk menggambar berbagai kurva dan grafik dua dimensi (2D).

---

## Plot Dasar

---

Ada fungsi yang sangat mendasar dari plot. Ada koordinat layar, yang selalu berkisar dari 0 hingga 1024 di setiap sumbu, tidak peduli apakah layarnya persegi atau tidak. Semut ada koordinat plot, yang dapat diatur dengan `setplot()`. Pemetaan antara koordinat tergantung pada jendela plot saat ini. Misalnya, `shrinkwindow()` default menyisakan ruang untuk label sumbu dan judul plot.

Dalam contoh, kita hanya menggambar beberapa garis acak dalam berbagai warna. Untuk detail tentang fungsi ini, pelajari fungsi inti EMT.

```
>clg; // clear screen
>window(0,0,1024,1024); // use all of the window
>setplot(0,1,0,1); // set plot coordinates
>hold on; // start overwrite mode
>n=100; X=random(n,2); Y=random(n,2); // get random points
>colors=rgb(random(n),random(n),random(n)); // get random colors
>loop 1 to n; color(colors[#]); plot(X[#],Y[#]); end; // plot
>hold off; // end overwrite mode
>insimg; // insert to notebook
```



```
>reset;
```

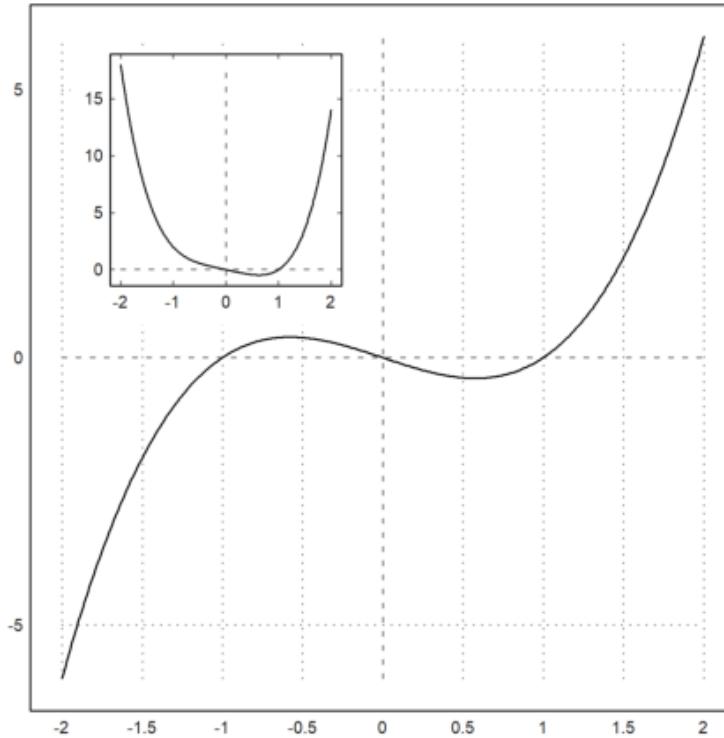
Grafik perlu ditahan, karena perintah plot() akan menghapus jendela plot.

Untuk menghapus semua yang kami lakukan, kami menggunakan reset().

Untuk menampilkan gambar hasil plot di layar notebook, perintah plot2d() dapat diakhiri dengan titik dua (:). Cara lain adalah perintah plot2d() diakhiri dengan titik koma (;), kemudian menggunakan perintah insimg() untuk menampilkan gambar hasil plot.

Untuk contoh lain, kami menggambar plot sebagai sisipan di plot lain. Ini dilakukan dengan mendefinisikan jendela plot yang lebih kecil. Perhatikan bahwa jendela ini tidak menyediakan ruang untuk label sumbu di luar jendela plot. Kita harus menambahkan beberapa margin untuk ini sesuai kebutuhan. Perhatikan bahwa kami menyimpan dan memulihkan jendela penuh, dan menahan plot saat ini saat kami memplot inset.

```
>plot2d("x^3-x");
>xw=200; yw=100; ww=300; hw=300;
>ow>window();
>>window(xw,yw,xw+ww,yw+hw);
>hold on;
>barclear(xw-50,yw-10,ww+60,ww+60);
>plot2d("x^4-x",grid=6);
```



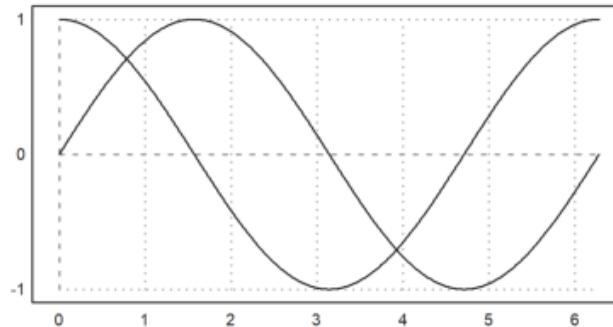
```
>hold off;  
>>window(ow);
```

Plot dengan banyak angka dicapai dengan cara yang sama. Ada fungsi figure() utilitas untuk ini.

Plot default menggunakan jendela plot persegi. Anda dapat mengubah ini dengan fungsi aspek(). Jangan lupa untuk mengatur ulang aspek nanti. Anda juga dapat mengubah default ini di menu dengan "Set Aspect" ke rasio aspek tertentu atau ke ukuran jendela grafis saat ini.

Tetapi Anda juga dapat mengubahnya untuk satu plot. Untuk ini, ukuran area plot saat ini diubah, dan jendela diatur sehingga label memiliki cukup ruang.

```
>aspect(2); // rasio panjang dan lebar 2:1  
>plot2d(["sin(x)","cos(x)"],0,2pi):
```



```
>aspect();  
>reset;
```

Fungsi reset() mengembalikan default plot termasuk rasio aspek.

#### Plot 2D di Euler

EMT Math Toolbox memiliki plot dalam 2D, baik untuk data maupun fungsi. EMT menggunakan fungsi plot2d. Fungsi ini dapat memplot fungsi dan data.

Dimungkinkan untuk membuat plot di Maxima menggunakan Gnuplot atau dengan Python menggunakan Math Plot Lib.

Euler dapat memplot plot 2D dari

- ekspresi
- fungsi, variabel, atau kurva parameter,
- vektor nilai x-y,
- awan titik di pesawat,
- kurva implisit dengan level atau wilayah level.
- Fungsi kompleks

Gaya plot mencakup berbagai gaya untuk garis dan titik, plot batang dan plot berbayang.

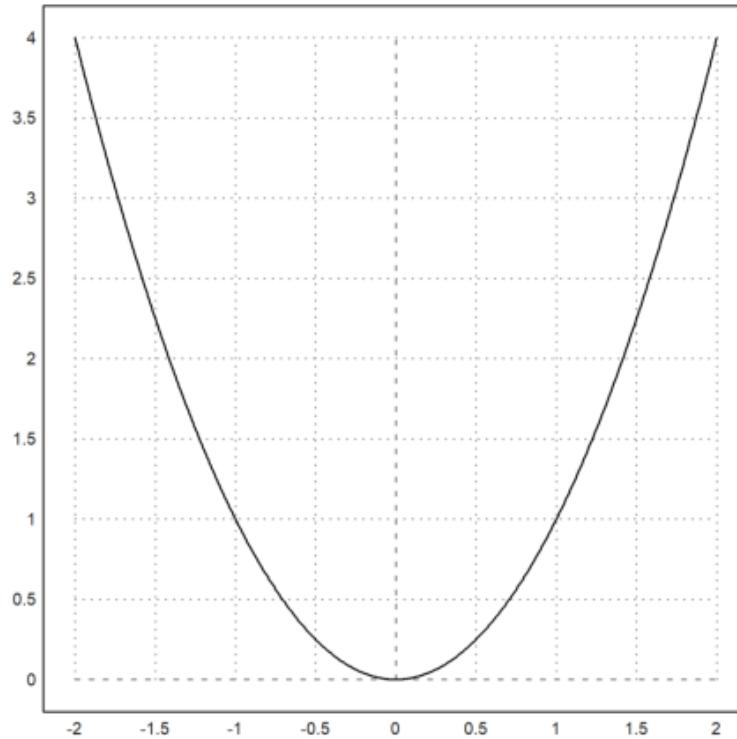
#### Plot Ekspresi atau Variabel

Ekspresi tunggal dalam "x" (mis. "4\*x^2") atau nama fungsi (mis. "f") menghasilkan grafik fungsi.

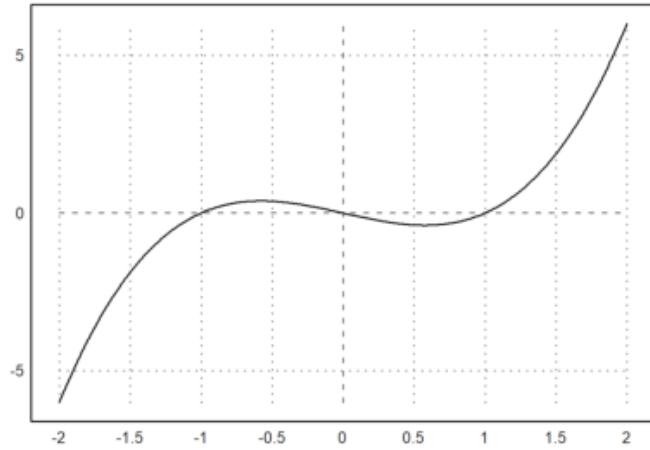
Berikut adalah contoh paling dasar, yang menggunakan rentang default dan menetapkan rentang y yang tepat agar sesuai dengan plot fungsi.

Catatan: Jika Anda mengakhiri baris perintah dengan titik dua ":", plot akan dimasukkan ke dalam jendela teks. Jika tidak, tekan TAB untuk melihat plot jika jendela plot tertutup.

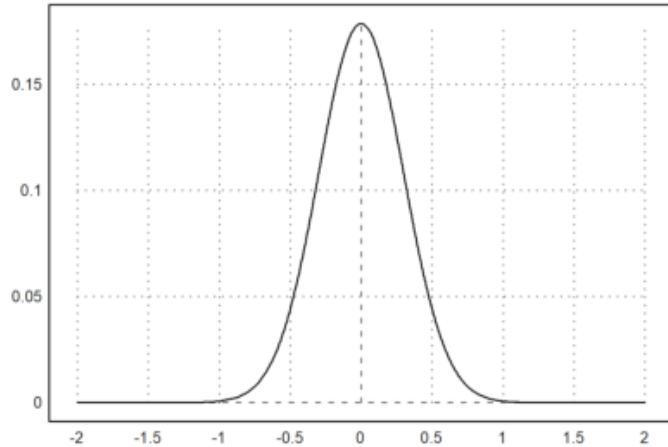
```
>plot2d("x^2"):
```



```
>aspect(1.5); plot2d("x^3-x"):
```



```
>a:=5.6; plot2d("exp(-a*x^2)/a"); insimg(30); // menampilkan gambar hasil plot setinggi 25 baris
```

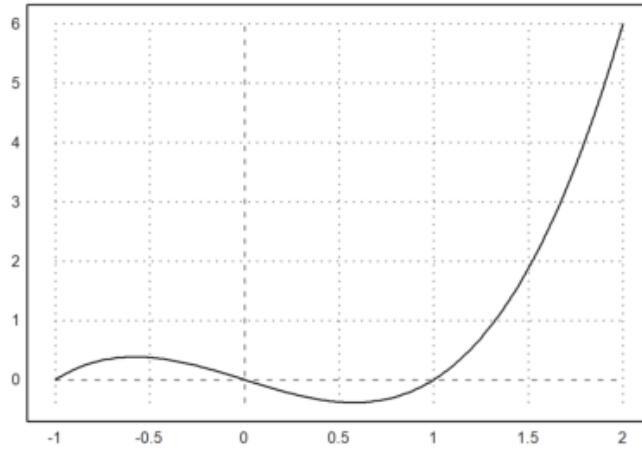


Dari beberapa contoh sebelumnya Anda dapat melihat bahwa Gambaran gambar plot menggunakan sumbu X dengan rentang nilai dari -2 sampai dengan 2. Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan nilai batas X (dan Y) di belakang ekspresi yang digambar.

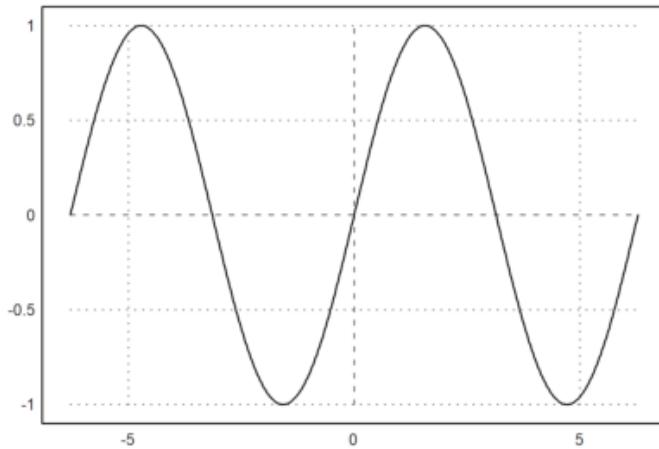
Rentang plot diatur dengan parameter yang ditetapkan berikut:

- a,b: rentang-x (default -2,2)
- c,d: y-range (default: skala dengan nilai)
- r: sebagai alternatif radius di sekitar pusat plot
- cx,cy: koordinat pusat plot (default 0,0)

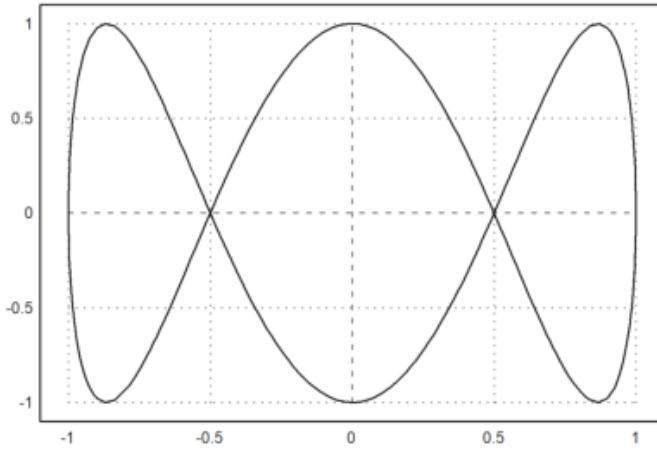
```
>plot2d("x^3-x",-1,2):
```



```
>plot2d("sin(x)",-2*pi,2*pi); // plot sin(x) pada interval [-2pi, 2pi]
```



```
>plot2d("cos(x)","sin(3*x)",xmin=0,xmax=2pi):
```



Alternatif untuk titik dua adalah perintah `insimg`(baris), yang menyisipkan plot yang menempati sejumlah baris teks tertentu.

Dalam opsi, plot dapat diatur untuk muncul

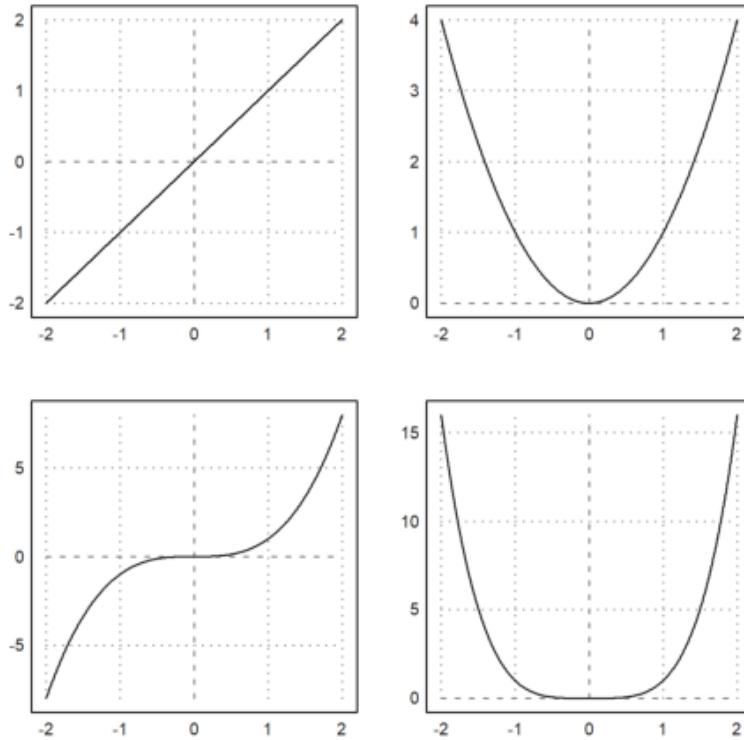
- di jendela terpisah yang dapat diubah ukurannya,
- di jendela buku catatan.

Lebih banyak gaya dapat dicapai dengan perintah plot tertentu.

Bagaimanapun, tekan tombol tabulator untuk melihat plot, jika disembunyikan.

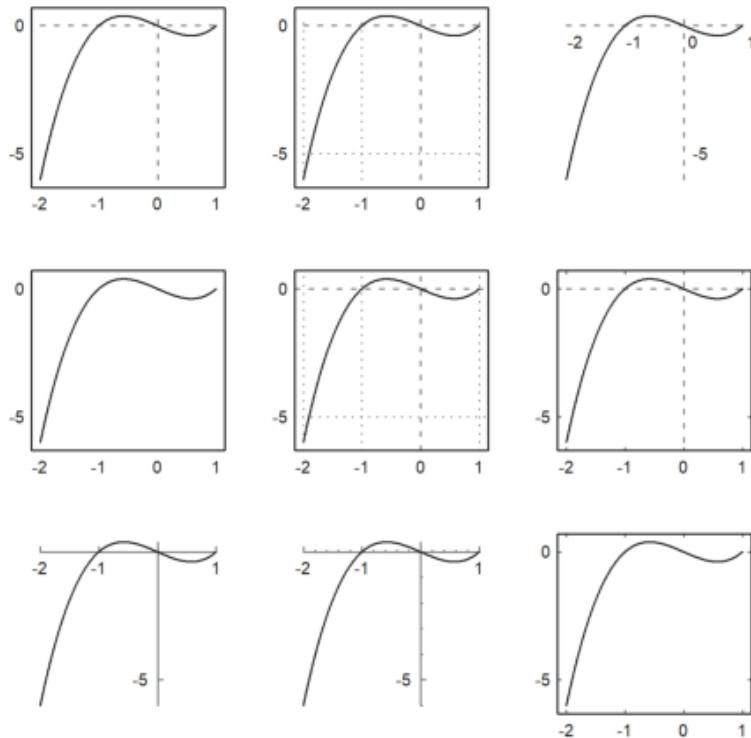
Untuk membagi jendela menjadi beberapa plot, gunakan perintah `figure()`. Dalam contoh, kami memplot  $x^1$  hingga  $x^4$  menjadi 4 bagian jendela. `figure(0)` mengatur ulang jendela default.

```
>reset;
>figure(2,2); ...
>for n=1 to 4; figure(n); plot2d("x^"+n); end; ...
>figure(0);
```



Di `plot2d()`, ada gaya alternatif yang tersedia dengan `grid=x`. Untuk gambaran umum, kami menunjukkan berbagai gaya kisi dalam satu gambar (lihat di bawah untuk perintah `figure()`). Gaya `kisi=0` tidak disertakan. Ini menunjukkan tidak ada grid dan tidak ada bingkai.

```
>figure(3,3); ...
>for k=1:9; figure(k); plot2d("x^3-x",-2,1,grid=k); end; ...
>figure(0):
```

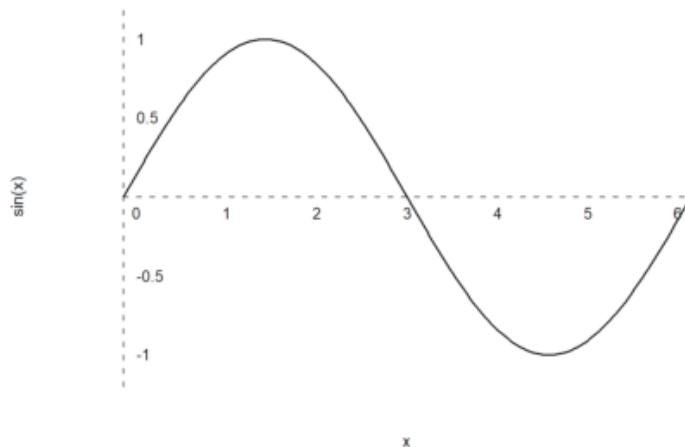


Jika argumen ke plot2d() adalah ekspresi yang diikuti oleh empat angka, angka-angka ini adalah rentang x dan y untuk plot.

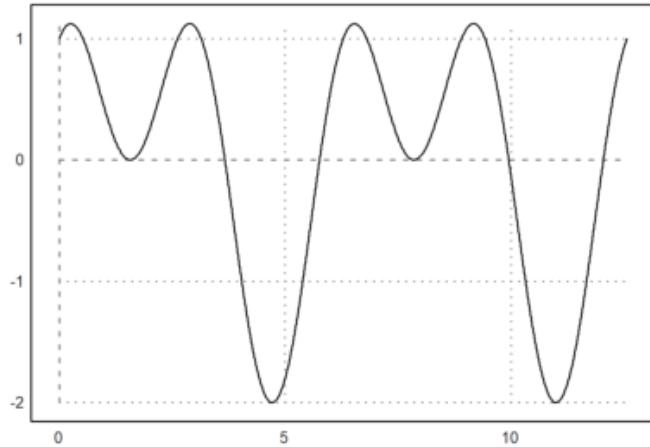
Atau, a, b, c, d dapat ditentukan sebagai parameter yang ditetapkan sebagai a=... dll.

Dalam contoh berikut, kita mengubah gaya kisi, menambahkan label, dan menggunakan label vertikal untuk sumbu y.

```
>aspect(1.5); plot2d("sin(x)",0,2pi,-1.2,1.2,grid=3,xl="x",yl="sin(x)":
```



```
>plot2d("sin(x)+cos(2*x)",0,4pi):
```

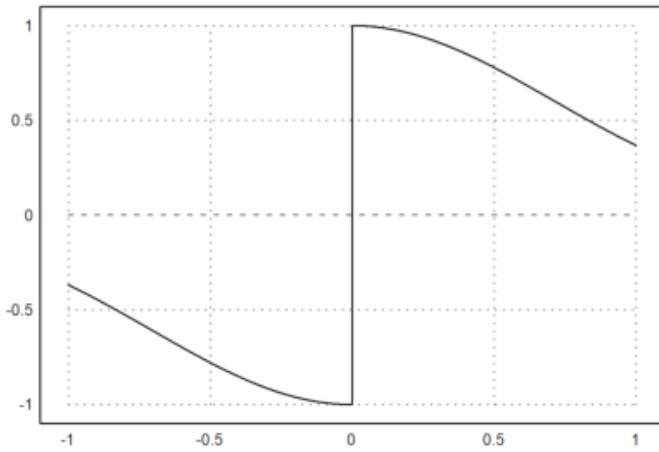


Gambar yang dihasilkan dengan memasukkan plot ke dalam jendela teks disimpan di direktori yang sama dengan buku catatan, secara default di subdirektori bernama "gambar". Mereka juga digunakan oleh ekspor HTML.

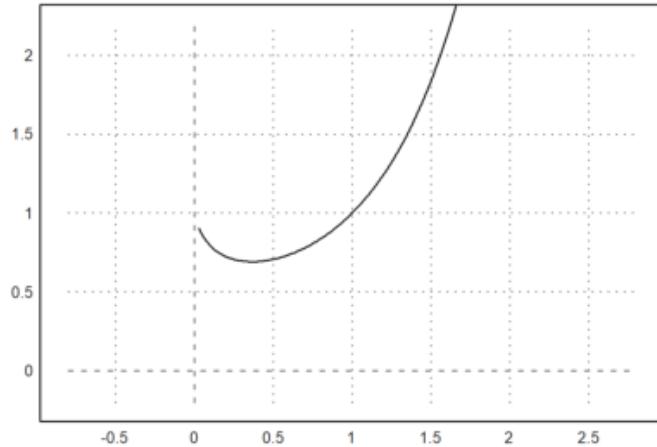
Anda cukup menandai gambar apa saja dan menyalinnya ke clipboard dengan Ctrl-C. Tentu saja, Anda juga dapat mengekspor grafik saat ini dengan fungsi di menu File.

Fungsi atau ekspresi dalam plot2d dievaluasi secara adaptif. Untuk kecepatan lebih, matikan plot adaptif dengan <adaptive dan tentukan jumlah subinterval dengan n=... Ini hanya diperlukan dalam kasus yang jarang terjadi.

```
>plot2d("sign(x)*exp(-x^2)",-1,1,<adaptive,n=10000):
```

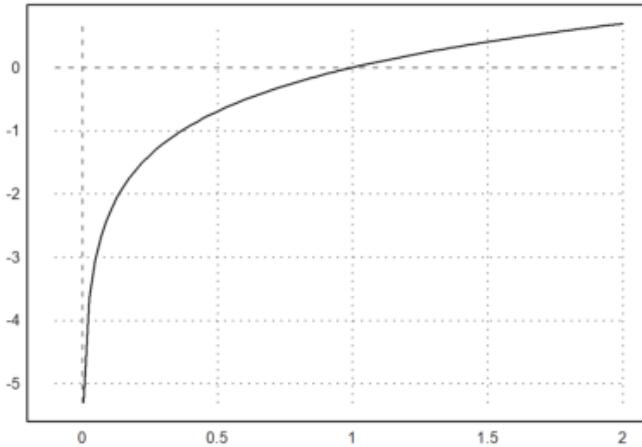


```
>plot2d("x^x",r=1.2,cx=1,cy=1):
```



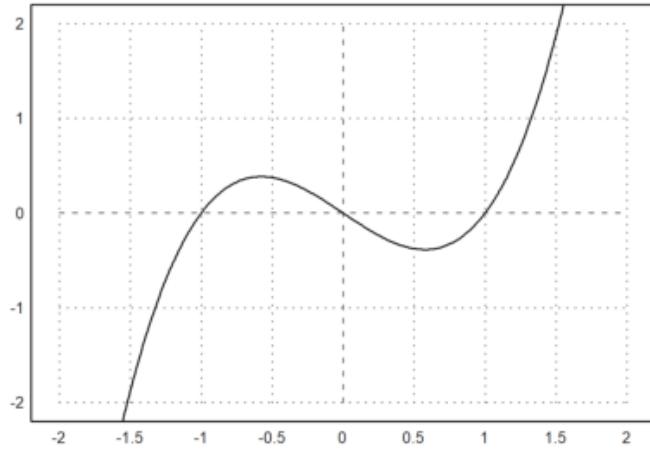
Perhatikan bahwa  $x^x$  tidak didefinisikan untuk  $x \leq 0$ . Fungsi `plot2d` menangkap kesalahan ini, dan mulai merencanakan segera setelah fungsi didefinisikan. Ini berfungsi untuk semua fungsi yang mengembalikan NAN keluar dari jangkauan definisinya.

```
>plot2d("log(x)",-0.1,2):
```

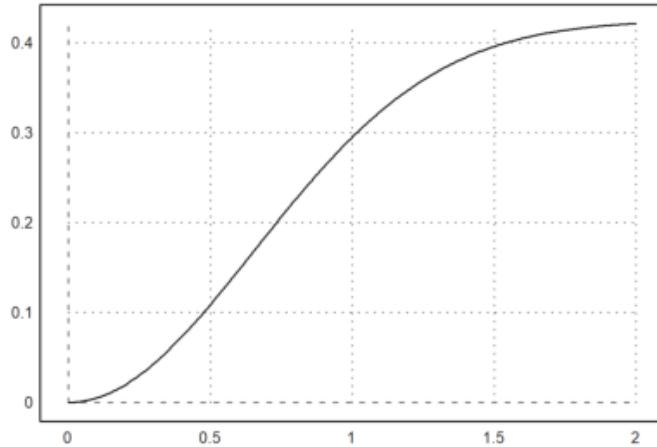


Parameter `square=true` (atau `>square`) memilih y-range secara otomatis sehingga hasilnya adalah jendela plot persegi. Perhatikan bahwa secara default, Euler menggunakan ruang persegi di dalam jendela plot.

```
>plot2d("x^3-x",>square):
```

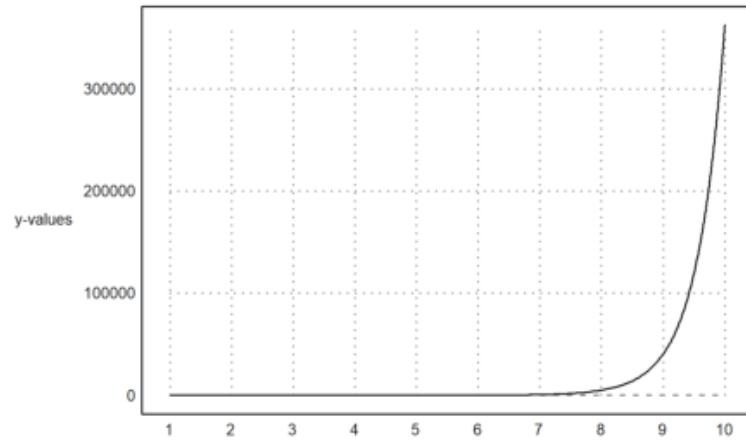


```
>plot2d(''integrate("sin(x)*exp(-x^2)",0,x)',0,2); // plot integral
```



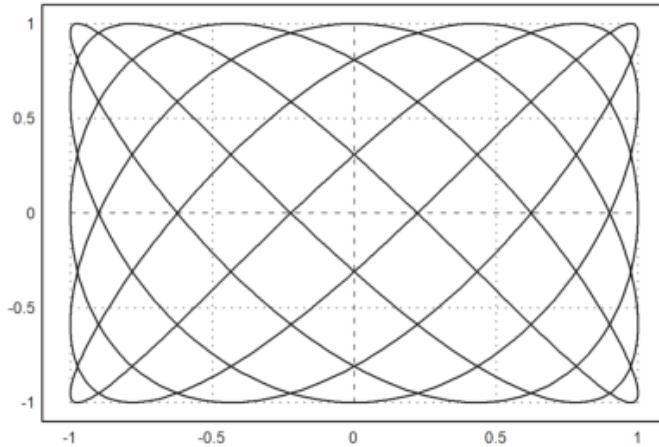
Jika Anda membutuhkan lebih banyak ruang untuk label-y, panggil shrinkwindow() dengan parameter yang lebih kecil, atau tetapkan nilai positif untuk "lebih kecil" di plot2d().

```
>plot2d("gamma(x)",1,10,yl="y-values",smaller=6,<vertical):
```

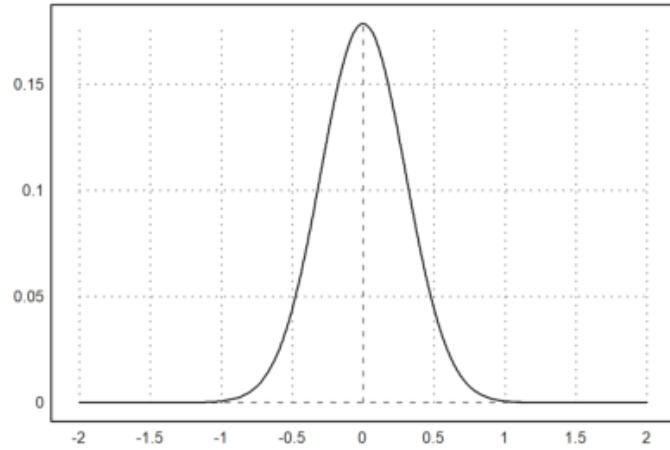


Ekspresi simbolik juga dapat digunakan, karena disimpan sebagai ekspresi string sederhana.

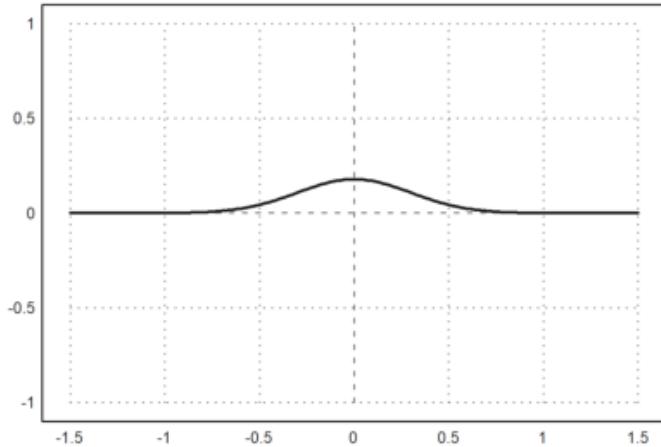
```
>x=linspace(0,2pi,1000); plot2d(sin(5x),cos(7x));
```



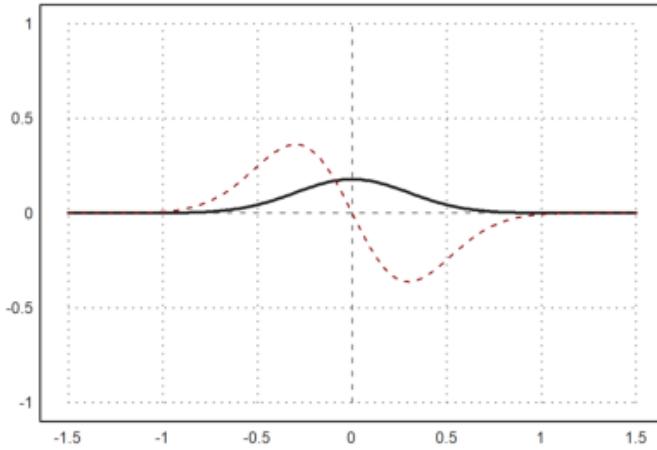
```
>a:=5.6; expr &= exp(-a*x^2)/a; // define expression  
>plot2d(expr,-2,2); // plot from -2 to 2
```



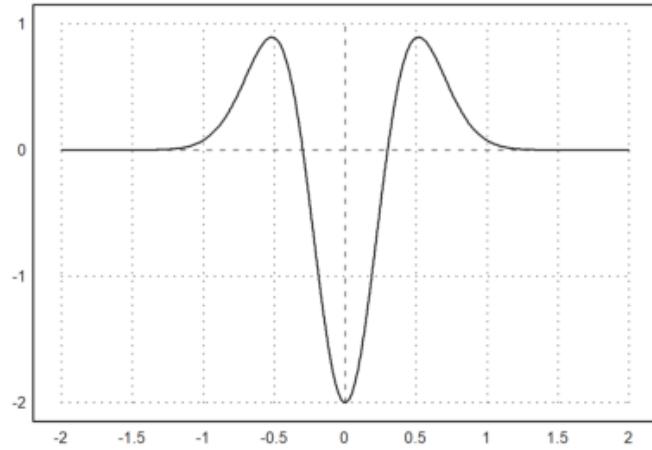
```
>plot2d(expr,r=1,thickness=2); // plot in a square around (0,0)
```



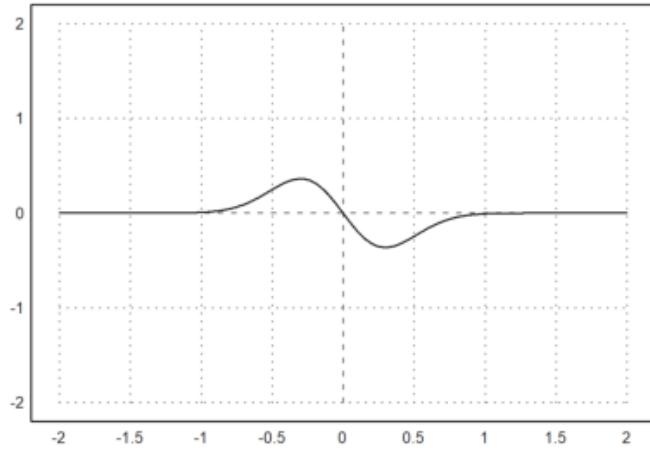
```
>plot2d(&diff(expr,x),>add,style="--",color=red); // add another plot
```



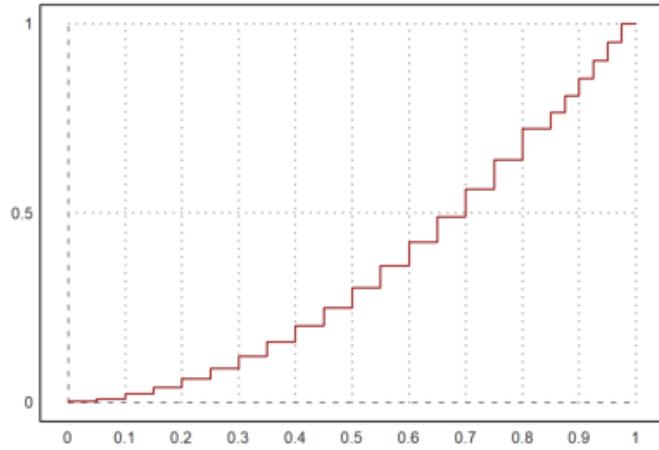
```
>plot2d(&diff(expr,x,2),a=-2,b=2,c=-2,d=1); // plot in rectangle
```



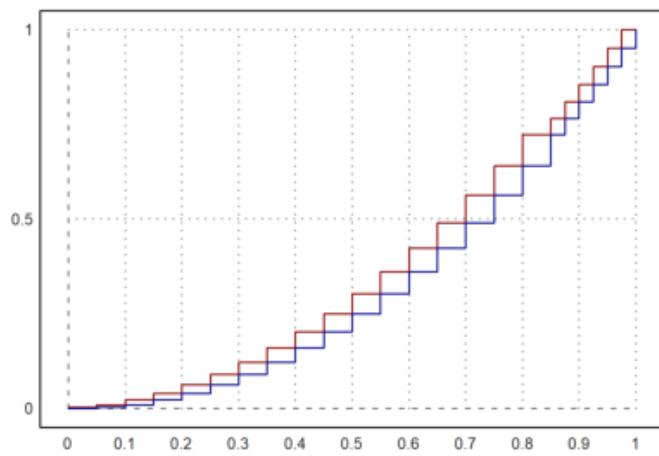
```
>plot2d(&diff(expr,x),a=-2,b=2,>square): // keep plot square
```



```
>plot2d("x^2",0,1,steps=1,color=red,n=10):
```



```
>plot2d("x^2",>add,steps=2,color=blue,n=10):
```



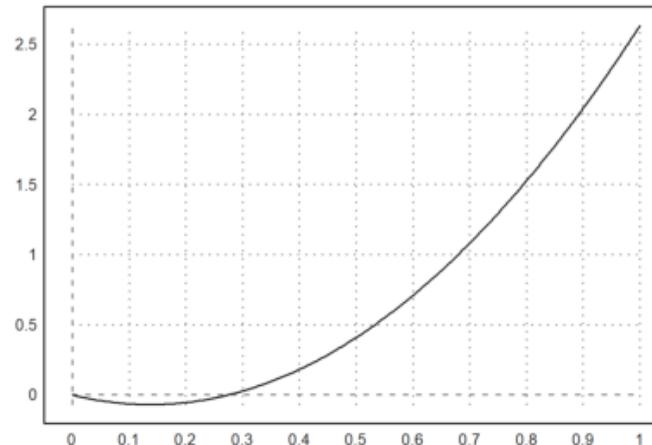
## Fungsi dalam satu Parameter

---

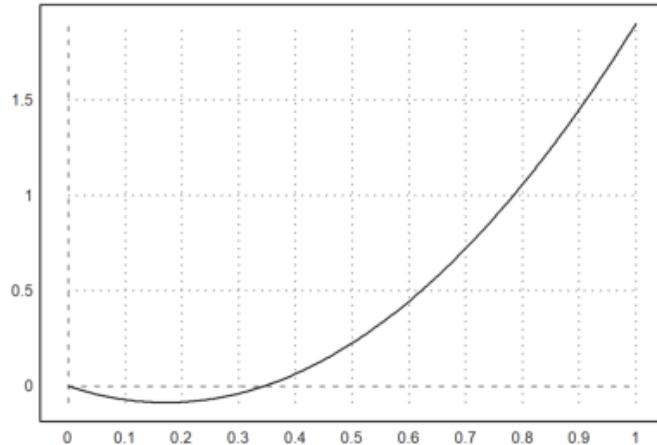
Fungsi plot yang paling penting untuk plot planar adalah `plot2d()`. Fungsi ini diimplementasikan dalam bahasa Euler dalam file "plot.e", yang dimuat di awal program.

Berikut adalah beberapa contoh menggunakan fungsi. Seperti biasa di EMT, fungsi yang berfungsi untuk fungsi atau ekspresi lain, Anda dapat meneruskan parameter tambahan (selain  $x$ ) yang bukan variabel global ke fungsi dengan parameter titik koma atau dengan koleksi panggilan.

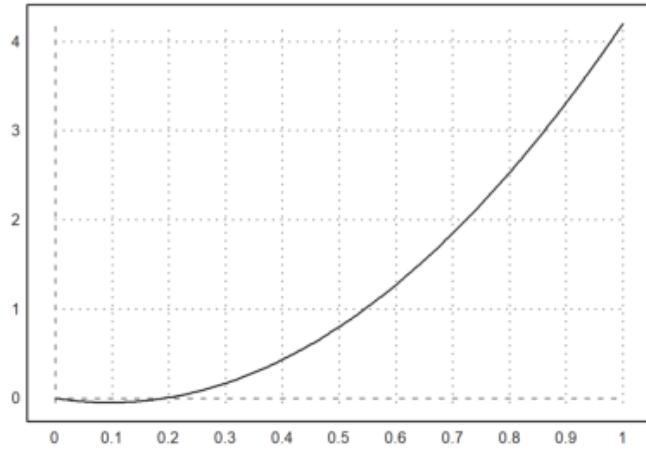
```
>function f(x,a) := x^2/a+a*x^2-x; // define a function  
>a=0.3; plot2d("f",0,1;a); // plot with a=0.3
```



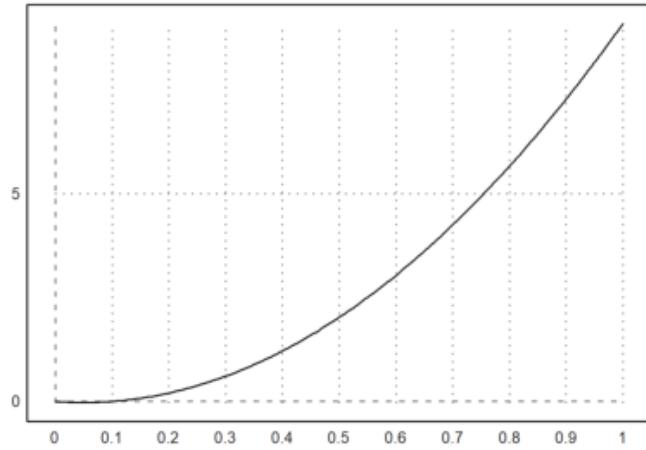
```
>plot2d("f",0,1;0.4): // plot with a=0.4
```



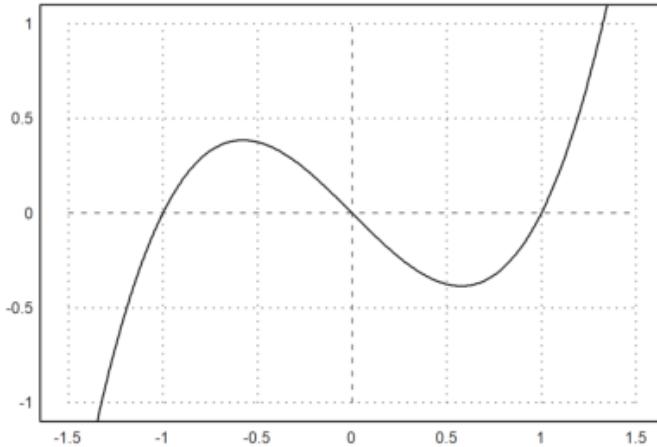
```
>plot2d({{"f",0.2}},0,1): // plot with a=0.2
```



```
>plot2d({{"f(x,b)",b=0.1}},0,1); // plot with 0.1
```



```
>function f(x) := x^3-x; ...
>plot2d("f",r=1):
```



Berikut adalah ringkasan dari fungsi yang diterima

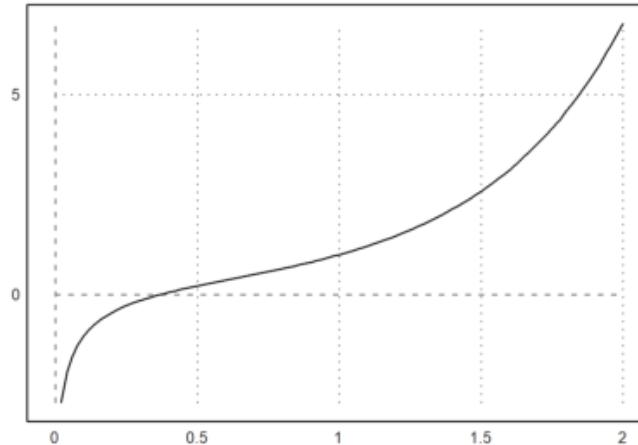
- ekspresi atau ekspresi simbolik dalam x
- fungsi atau fungsi simbolis dengan nama sebagai "f"
- fungsi simbolis hanya dengan nama f

Fungsi plot2d() juga menerima fungsi simbolis. Untuk fungsi simbolis, nama saja yang berfungsi.

```
>function f(x) &= diff(x^x,x)
```

$$\frac{x}{x} (\log(x) + 1)$$

```
>plot2d(f,0,2):
```

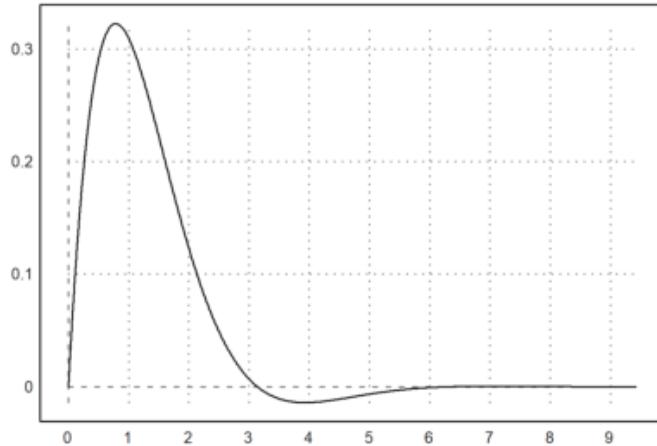


Tentu saja, untuk ekspresi atau ekspresi simbolik, nama variabel sudah cukup untuk memplotnya.

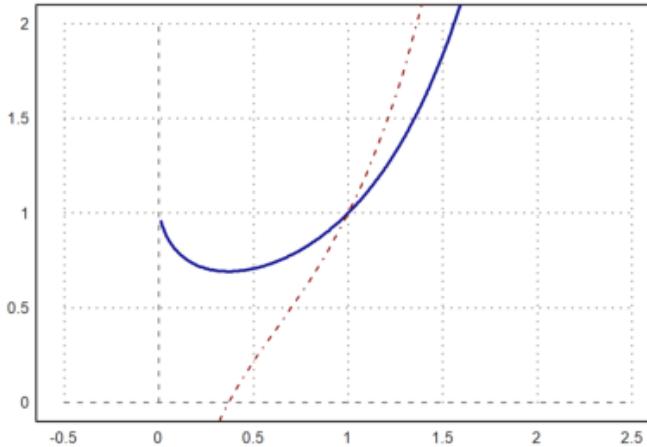
```
>expr &= sin(x)*exp(-x)
```

$$E^{-x} \sin(x)$$

```
>plot2d(expr,0,3pi):
```



```
>function f(x) &= x^x;  
>plot2d(f,r=1,cx=1,cy=1,color=blue,thickness=2);  
>plot2d(&diff(f(x),x),>add,color=red,style="- . -"):
```



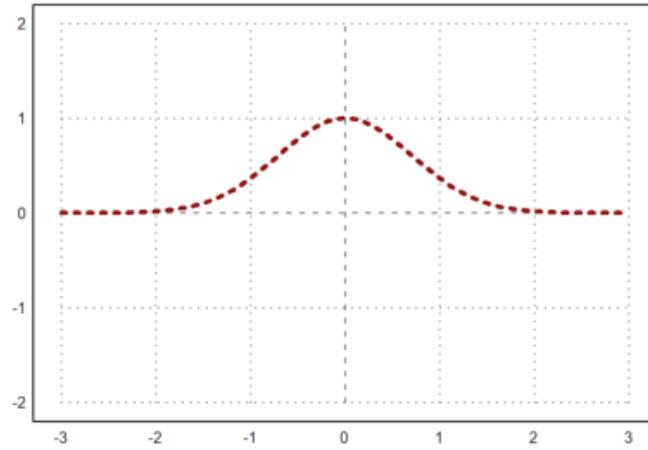
Untuk gaya garis ada berbagai pilihan.

- `gaya="..."`. Pilih dari `"-`, `"--"`, `"-.."`, `".-."`, `"-.-"`.
- warna: Lihat di bawah untuk warna.
- ketebalan: Default adalah 1.

Warna dapat dipilih sebagai salah satu warna default, atau sebagai warna RGB.

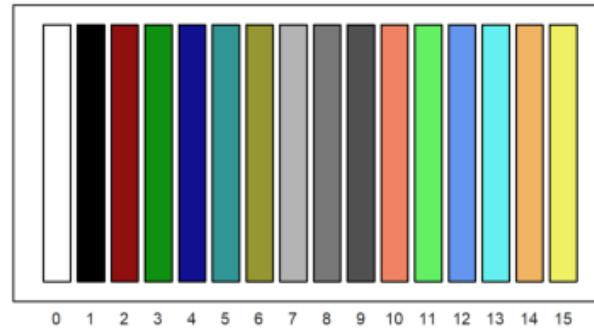
- 0.15: indeks warna default.
- konstanta warna: putih, hitam, merah, hijau, biru, cyan, zaitun, abu-abu muda, abu-abu, abu-abu tua, oranye, hijau muda, pirus, biru muda, oranye terang, kuning
- `rgb(merah, hijau, biru)`: parameter adalah real dalam [0,1].

```
>plot2d("exp(-x^2)",r=2,color=red,thickness=3,style="--"):
```



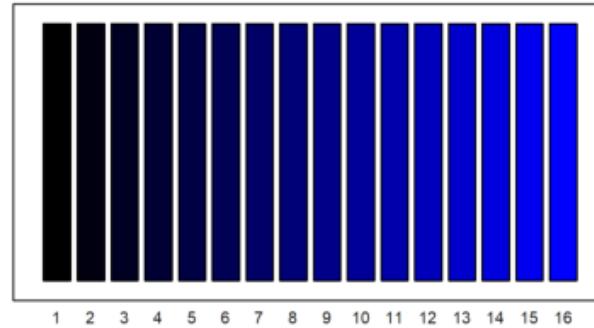
Berikut adalah tampilan warna EMT yang telah ditentukan sebelumnya.

```
>aspect(2); columnsplot(ones(1,16),lab=0:15,grid=0,color=0:15):
```



But you can use any color.

```
>columnsplot(ones(1,16),grid=0,color=rgb(0,0,linspace(0,1,15))):
```

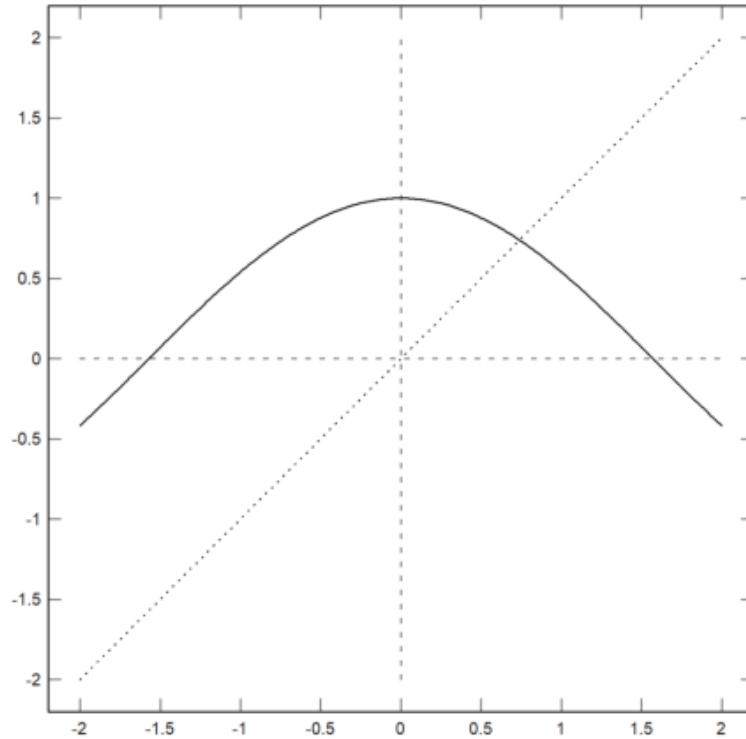


## Menggambar Beberapa Kurva pada bidang koordinat yang sama

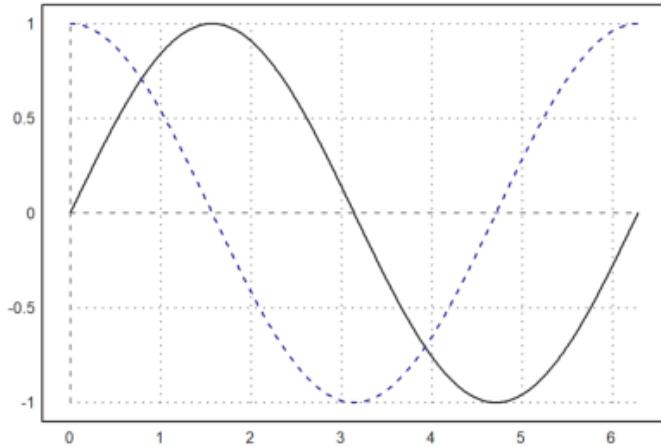
---

Plot lebih dari satu fungsi (multiple function) ke dalam satu jendela dapat dilakukan dengan berbagai cara. Salah satu metode menggunakan >add untuk beberapa panggilan ke plot2d secara keseluruhan, tetapi panggilan pertama. Kami telah menggunakan fitur ini dalam contoh di atas.

```
>aspect(); plot2d("cos(x)",r=2,grid=6); plot2d("x",style=".",>add):
```

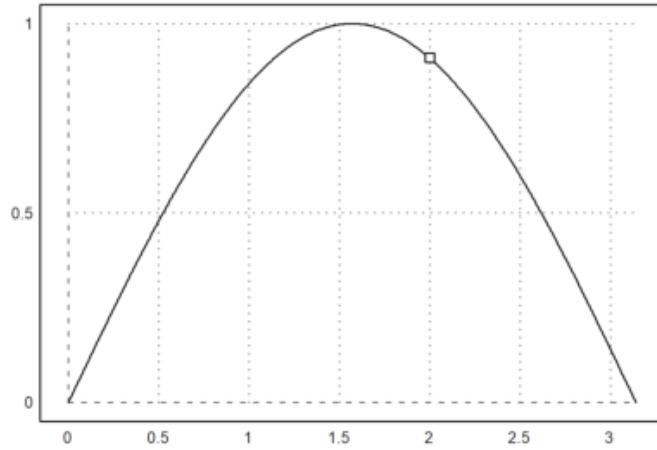


```
>aspect(1.5); plot2d("sin(x)",0,2pi); plot2d("cos(x)",color=blue,style="--",>add):
```



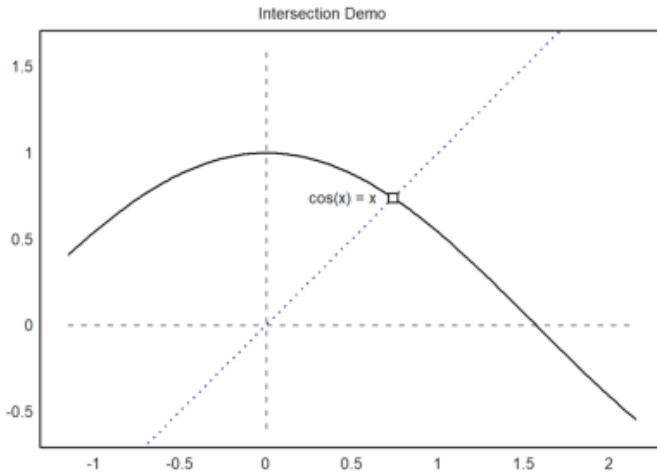
Salah satu kegunaan `>add` adalah untuk menambahkan titik pada kurva.

```
>plot2d("sin(x)",0,pi); plot2d(2,sin(2),>points,>add):
```



Kami menambahkan titik persimpangan dengan label (pada posisi "cl" untuk kiri tengah), dan memasukkan hasilnya ke dalam notebook. Kami juga menambahkan judul ke plot.

```
>plot2d(["cos(x)","x"],r=1.1,cx=0.5,cy=0.5, ...
> color=[black,blue],style=["-","."], ...
> grid=1);
>x0=solve("cos(x)-x",1); ...
> plot2d(x0,x0,>points,>add,title="Intersection Demo"); ...
> label("cos(x) = x",x0,x0,pos="cl",offset=20):
```



Dalam demo berikut, kami memplot fungsi  $\text{sinc}(x)=\sin(x)/x$  dan ekspansi Taylor ke-8 dan ke-16. Kami menghitung ekspansi ini menggunakan Maxima melalui ekspresi simbolis.

Plot ini dilakukan dalam perintah multi-baris berikut dengan tiga panggilan ke `plot2d()`. Yang kedua dan yang ketiga memiliki set flag `>add`, yang membuat plot menggunakan rentang sebelumnya.

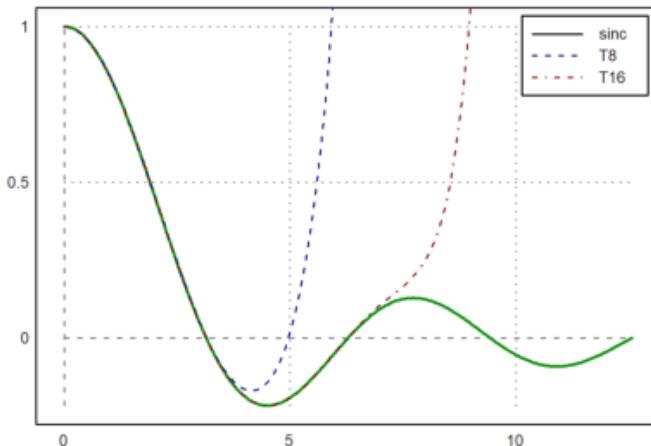
Kami menambahkan kotak label yang menjelaskan fungsi.

```
>$taylor(sin(x)/x,x,0,4)
```

$$\frac{x^4}{120} - \frac{x^2}{6} + 1$$

```

>plot2d("sinc(x)",0,4pi,color=green,thickness=2); ...
> plot2d(&taylor(sin(x)/x,x,0,8),>add,color=blue,style="--"); ...
> plot2d(&taylor(sin(x)/x,x,0,16),>add,color=red,style="-."); ...
> labelbox(["sinc","T8","T16"],styles=["-","--","-."], ...
> colors=[black,blue,red]):
```



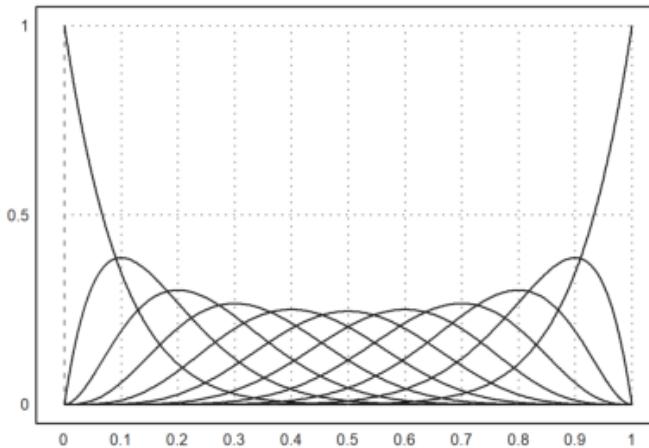
Dalam contoh berikut, kami menghasilkan Bernstein-Polinomial.

$$B_i(x) = \binom{n}{i} x^i (1-x)^{n-i}$$

```

>plot2d("(1-x)^10",0,1); // plot first function
>for i=1 to 10; plot2d("bin(10,i)*x^i*(1-x)^(10-i)",>add); end;
>insimg;

```



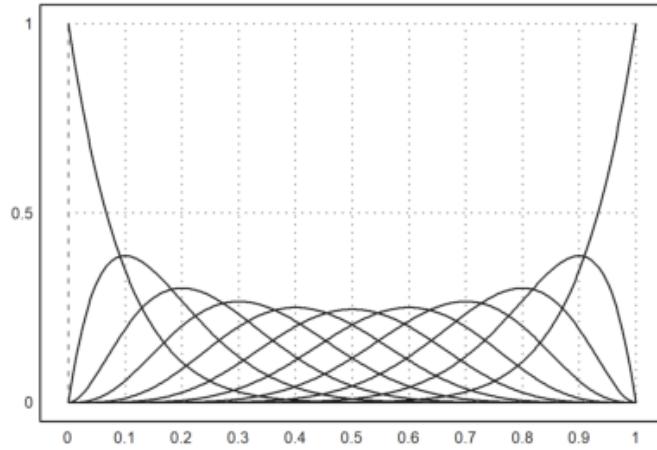
Metode kedua menggunakan pasangan matriks nilai-x dan matriks nilai-y yang berukuran sama.

Kami menghasilkan matriks nilai dengan satu Polinomial Bernstein di setiap baris. Untuk ini, kita cukup menggunakan vektor kolom  $i$ . Lihat pengantar tentang bahasa matriks untuk mempelajari lebih detail.

```

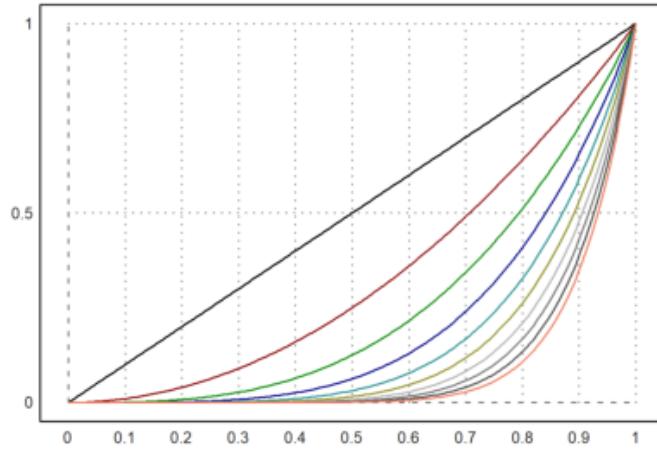
>x=linspace(0,1,500);
>n=10; k=(0:n)'; // n is row vector, k is column vector
>y=bin(n,k)*x.^k*(1-x).^(n-k); // y is a matrix then
>plot2d(x,y):

```



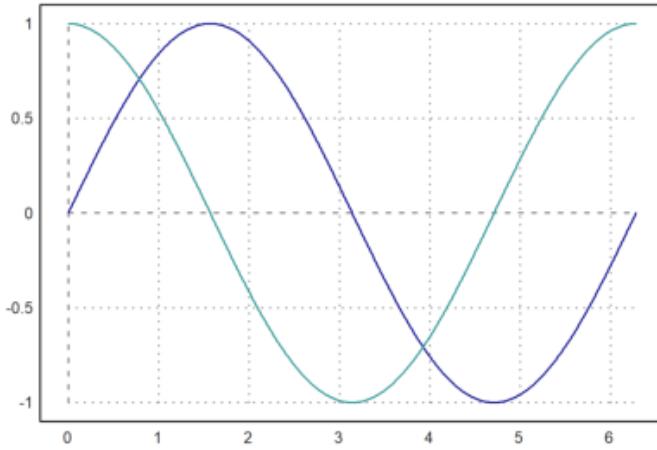
Perhatikan bahwa parameter warna dapat berupa vektor. Kemudian setiap warna digunakan untuk setiap baris matriks.

```
>x=linspace(0,1,200); y=x^(1:10)'; plot2d(x,y,color=1:10):
```

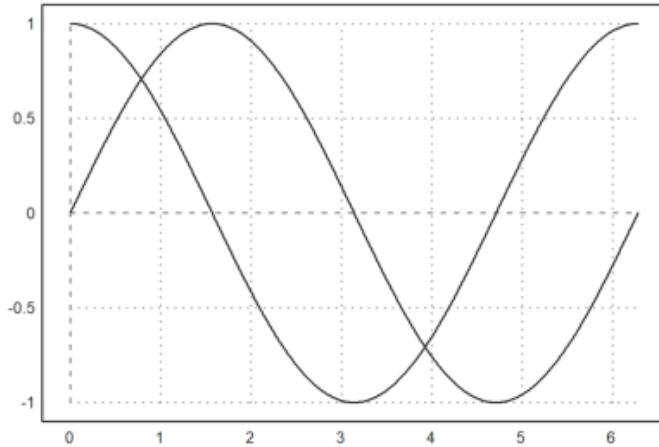


Metode lain adalah menggunakan vektor ekspresi (string). Anda kemudian dapat menggunakan larik warna, larik gaya, dan larik ketebalan dengan panjang yang sama.

```
>plot2d(["sin(x)","cos(x)"],0,2pi,color=4:5):
```



```
>plot2d(["sin(x)","cos(x")],0,2pi); // plot vector of expressions
```



Kita bisa mendapatkan vektor seperti itu dari Maxima menggunakan makelist() dan mxm2str().

```
>v &= makelist(binomial(10,i)*x^i*(1-x)^(10-i),i,0,10) // make list
```

```

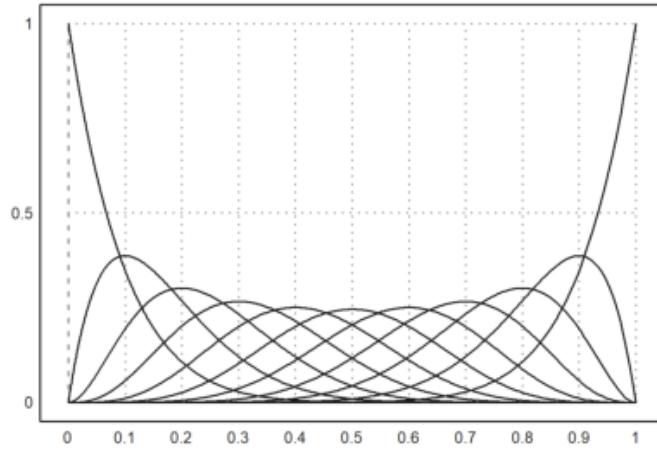
          10      9      8  2      7  3
[(1 - x) , 10 (1 - x) x, 45 (1 - x) x , 120 (1 - x) x ,
   6  4      5  5      4  6      3  7
210 (1 - x) x , 252 (1 - x) x , 210 (1 - x) x , 120 (1 - x) x ,
   2  8      9  10
45 (1 - x) x , 10 (1 - x) x , x ]

```

```
>mxm2str(v) // get a vector of strings from the symbolic vector
```

```
(1-x)^10  
10*(1-x)^9*x  
45*(1-x)^8*x^2  
120*(1-x)^7*x^3  
210*(1-x)^6*x^4  
252*(1-x)^5*x^5  
210*(1-x)^4*x^6  
120*(1-x)^3*x^7  
45*(1-x)^2*x^8  
10*(1-x)*x^9  
x^10
```

```
>plot2d(mxm2str(v),0,1); // plot functions
```

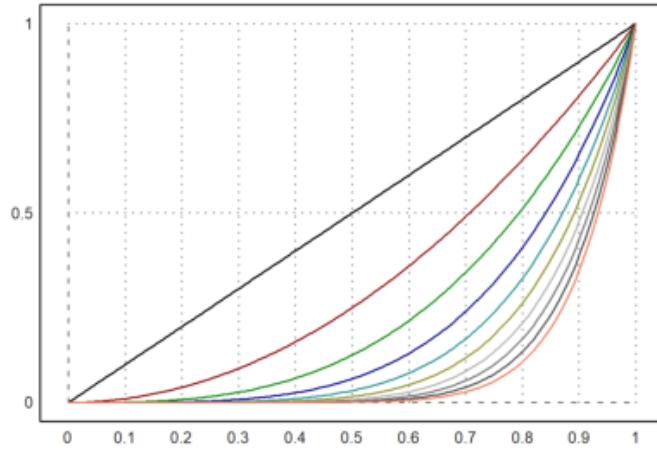


Alternatif lain adalah dengan menggunakan bahasa matriks Euler.

Jika ekspresi menghasilkan matriks fungsi, dengan satu fungsi di setiap baris, semua fungsi ini akan diplot ke dalam satu plot.

Untuk ini, gunakan vektor parameter dalam bentuk vektor kolom. Jika array warna ditambahkan, itu akan digunakan untuk setiap baris plot.

```
>n=(1:10)'; plot2d("x^n",0,1,color=1:10):
```

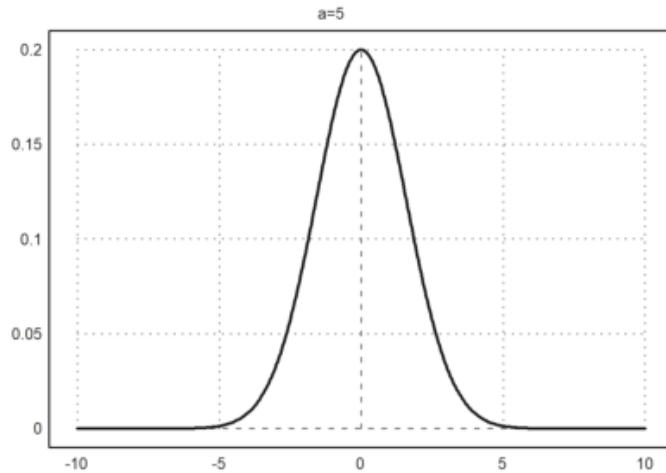


Ekspresi dan fungsi satu baris dapat melihat variabel global.

Jika Anda tidak dapat menggunakan variabel global, Anda perlu menggunakan fungsi dengan parameter tambahan, dan meneruskan parameter ini sebagai parameter titik koma.

Berhati-hatilah, untuk meletakkan semua parameter yang ditetapkan di akhir perintah plot2d. Dalam contoh kita meneruskan a=5 ke fungsi f, yang kita plot dari -10 hingga 10.

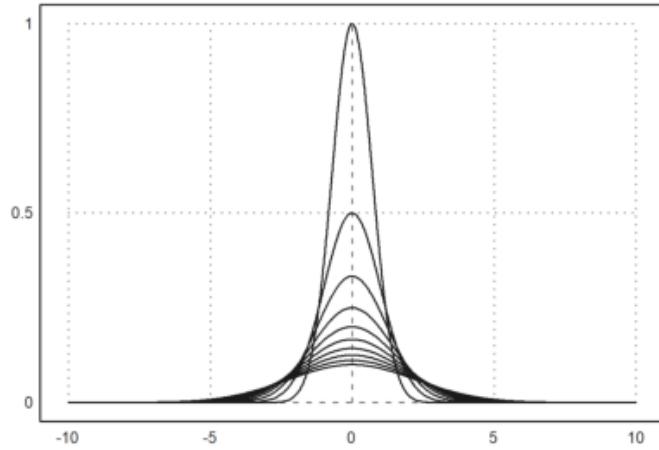
```
>function f(x,a) := 1/a*exp(-x^2/a); ...
>plot2d("f",-10,10;5,thickness=2,title="a=5");
```



Atau, gunakan koleksi dengan nama fungsi dan semua parameter tambahan. Daftar khusus ini disebut koleksi panggilan, dan itu adalah cara yang lebih disukai untuk meneruskan argumen ke fungsi yang dengan sendirinya diteruskan sebagai argumen ke fungsi lain.

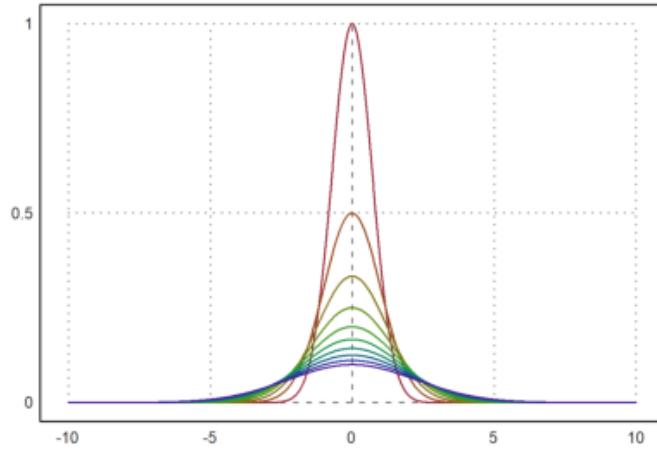
Dalam contoh berikut, kami menggunakan loop untuk memplot beberapa fungsi (lihat tutorial tentang pemrograman untuk loop).

```
>plot2d({{"f",1}},-10,10); ...
>for a=2:10; plot2d({{"f",a}},>add); end:
```



Kami dapat mencapai hasil yang sama dengan cara berikut menggunakan bahasa matriks EMT. Setiap baris matriks  $f(x,a)$  adalah satu fungsi. Selain itu, kita dapat mengatur warna untuk setiap baris matriks. Klik dua kali pada fungsi getspectral() untuk penjelasannya.

```
>x=-10:0.01:10; a=(1:10)'; plot2d(x,f(x,a),color=getspectral(a/10)):
```



## Label Teks

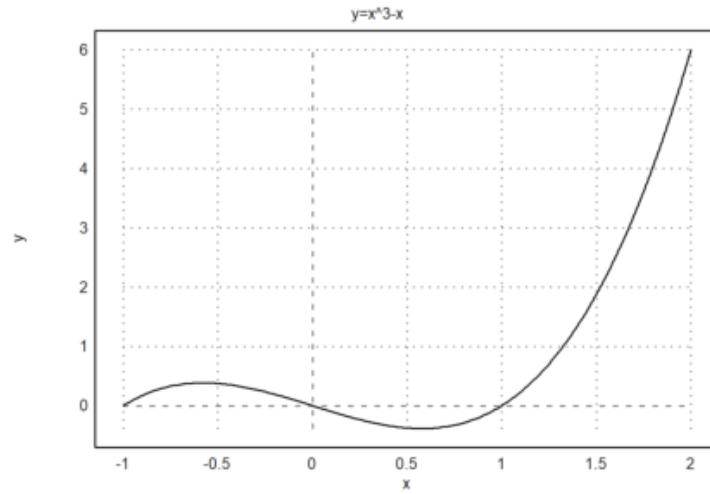
---

Dekorasi sederhana bisa

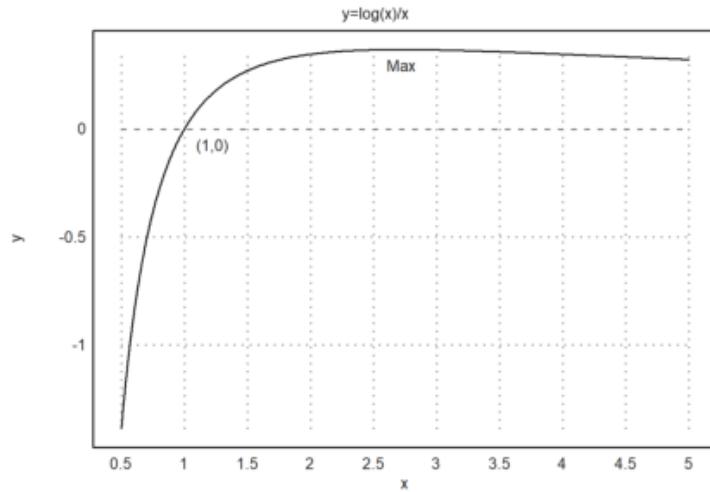
- judul dengan judul="..."
- x- dan y-label dengan xl="...", yl="..."
- label teks lain dengan label("...",x,y)

Perintah label akan memplot ke dalam plot saat ini pada koordinat plot (x,y). Itu bisa mengambil argumen posisi.

```
>plot2d("x^3-x",-1,2,title="y=x^3-x",yl="y",xl="x"):
```

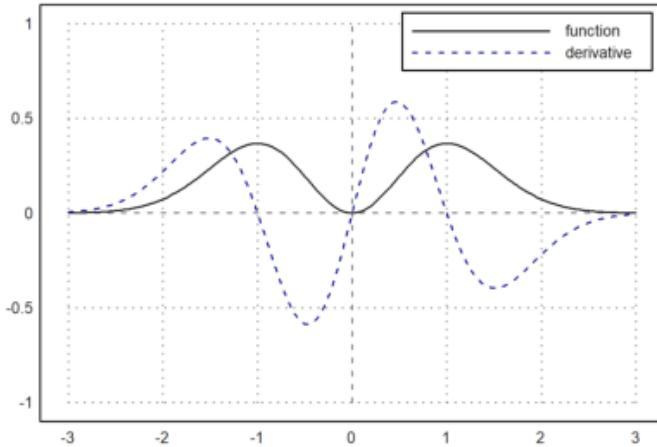


```
>expr := "log(x)/x"; ...
> plot2d(expr,0.5,5,title="y="+expr,xl="x",yl="y"); ...
> label("(1,0)",1,0); label("Max",E,expr(E),pos="lc");
```



Ada juga fungsi `labelbox()`, yang dapat menampilkan fungsi dan teks. Dibutuhkan vektor string dan warna, satu item untuk setiap fungsi.

```
>function f(x) &= x^2*exp(-x^2); ...
>plot2d(&f(x),a=-3,b=3,c=-1,d=1); ...
>plot2d(&diff(f(x),x),>add,color=blue,style="--"); ...
>labelbox(["function","derivative"],styles=["-","--"], ...
>    colors=[black,blue],w=0.4):
```

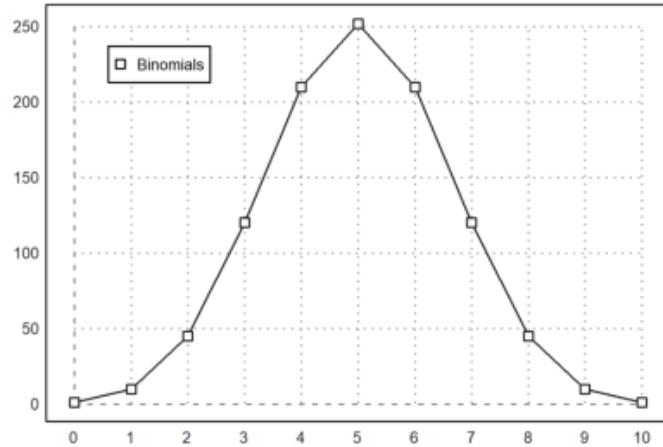


Kotak ditambatkan di kanan atas secara default, tetapi > kiri menambatkannya di kiri atas. Anda dapat memindahkannya ke tempat yang Anda suka. Posisi jangkar adalah sudut kanan atas kotak, dan angkanya adalah pecahan dari ukuran jendela grafik. Lebarnya otomatis.

Untuk plot titik, kotak label juga berfungsi. Tambahkan parameter >points, atau vektor flag, satu untuk setiap label.

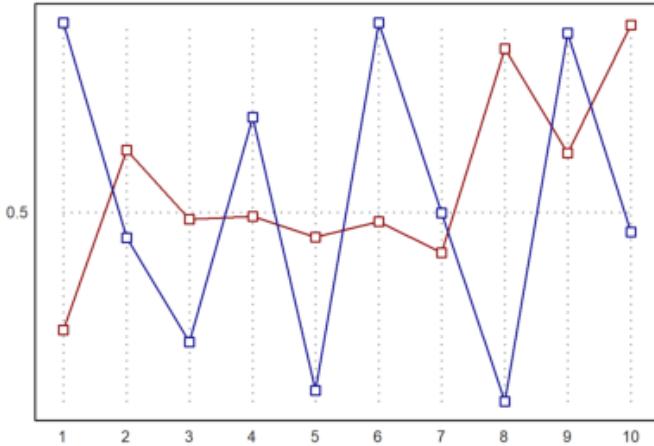
Dalam contoh berikut, hanya ada satu fungsi. Jadi kita bisa menggunakan string sebagai pengganti vektor string. Kami mengatur warna teks menjadi hitam untuk contoh ini.

```
>n=10; plot2d(0:n,bin(n,0:n),>addpoints); ...
>labelbox("Binomials",styles="[]",>points,x=0.1,y=0.1, ...
>tcolor=black,>left):
```



Gaya plot ini juga tersedia di statplot(). Seperti di plot2d() warna dapat diatur untuk setiap baris plot. Ada lebih banyak plot khusus untuk keperluan statistik (lihat tutorial tentang statistik).

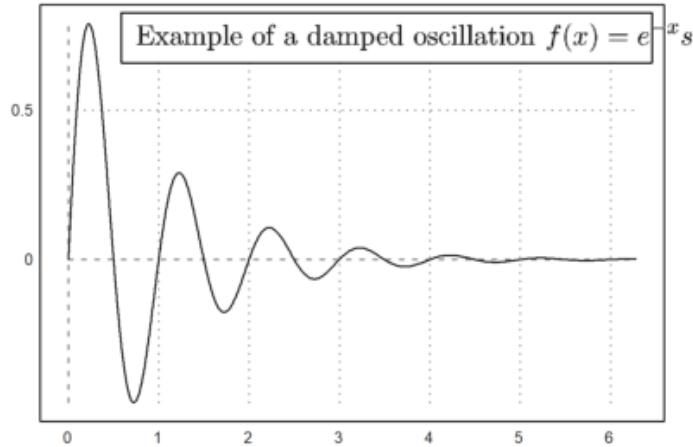
```
>statplot(1:10,random(2,10),color=[red,blue]):
```



Fitur serupa adalah fungsi `textbox()`.

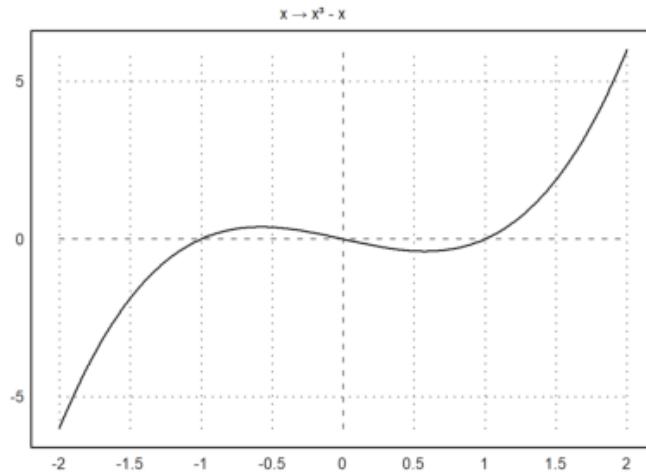
Lebar secara default adalah lebar maksimal dari baris teks. Tapi itu bisa diatur oleh pengguna juga.

```
>function f(x) &= exp(-x)*sin(2*pi*x); ...
>plot2d("f(x)",0,2pi); ...
>textbox(latex("\text{Example of a damped oscillation}\\" f(x)=e^{-x}\sin(2\pi x)'),w=0.85):
```



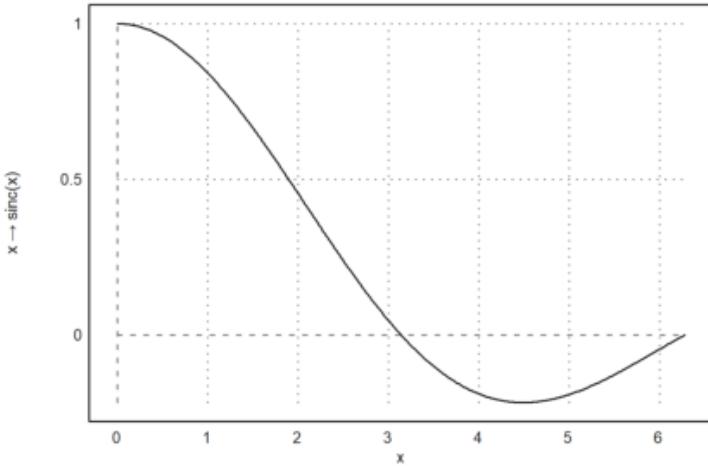
Label teks, judul, kotak label, dan teks lainnya dapat berisi string Unicode (lihat sintaks EMT untuk mengetahui lebih lanjut tentang string Unicode).

```
>plot2d("x^3-x",title=u"x &rarr; x3 - x"):
```



Label pada sumbu x dan y bisa vertikal, begitu juga sumbunya.

```
>plot2d("sinc(x)",0,2pi,xl="x",yl=u"x &rarr; sinc(x)",>vertical):
```



## LaTeX

---

Anda juga dapat memplot rumus LaTeX jika Anda telah menginstal sistem LaTeX. Saya merekomendasikan MiKTeX. Jalur ke biner "lateks" dan "dvipng" harus berada di jalur sistem, atau Anda harus mengatur LaTeX di menu opsi.

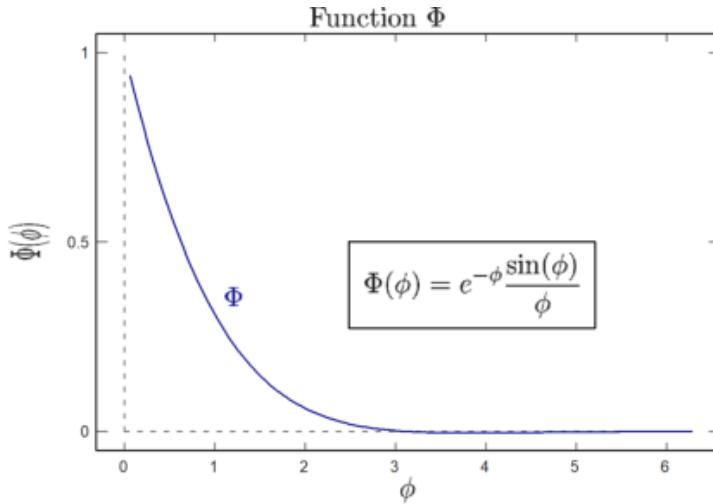
Perhatikan, bahwa penguraian LaTeX lambat. Jika Anda ingin menggunakan LaTeX dalam plot animasi, Anda harus memanggil `latex()` sebelum loop sekali dan menggunakan hasilnya (gambar dalam matriks RGB).

Dalam plot berikut, kami menggunakan LaTeX untuk label x dan y, label, kotak label, dan judul plot.

```

>plot2d("exp(-x)*sin(x)/x",a=0,b=2pi,c=0,d=1,grid=6,color=blue, ...
>  title=latex("\text{Function } \Phi"), ...
>  xl=latex("\phi"),yl=latex("\Phi(\phi)"); ...
>textbox( ...
>  latex("\Phi(\phi) = e^{-\phi} \frac{\sin(\phi)}{\phi}"),x=0.8,y=0.5); ...
>label(latex("\Phi",color=blue),1,0.4):

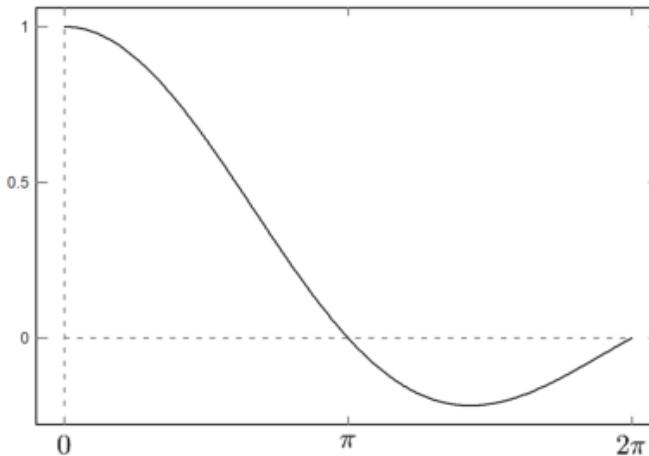
```



Seringkali, kami menginginkan spasi dan label teks non-konformal pada sumbu x. Kita dapat menggunakan `xaxis()` dan `yaxis()` seperti yang akan kita tunjukkan nanti.

Cara termudah adalah dengan membuat plot kosong dengan bingkai menggunakan `grid=4`, lalu menambahkan grid dengan `ygrid()` dan `xgrid()`. Dalam contoh berikut, kami menggunakan tiga string LaTeX untuk label pada sumbu x dengan `xtick()`.

```
>plot2d("sinc(x)",0,2pi,grid=4,<ticks); ...
>ygrid(-2:0.5:2,grid=6); ...
>xgrid([0:2]*pi,<ticks,grid=6); ...
>xlabel([0,pi,2pi],["0"," $\pi$ "," $2\pi$ "],>tex):
```



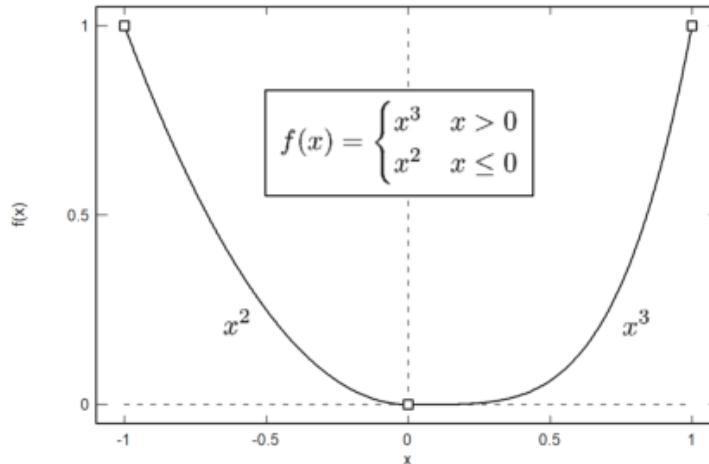
Tentu saja, fungsi juga dapat digunakan.

```
>function map f(x) ...
if x>0 then return x^4
else return x^2
endif
endfunction
```

Parameter "peta" membantu menggunakan fungsi untuk vektor. Untuk plot, itu tidak perlu. Tetapi untuk mendemonstrasikan vektorisasi itu berguna, kami menambahkan beberapa poin kunci ke plot di  $x=-1$ ,  $x=0$  dan  $x=1$ .

Pada plot berikut, kami juga memasukkan beberapa kode LaTeX. Kami menggunakannya untuk dua label dan kotak teks. Tentu saja, Anda hanya akan dapat menggunakan LaTeX jika Anda telah menginstal LaTeX dengan benar.

```
>plot2d("f", -1, 1, xl="x", yl="f(x)", grid=6); ...
>plot2d([-1,0,1], f([-1,0,1]), >points, >add); ...
>label(latex("x^3"), 0.72, f(0.72)); ...
>label(latex("x^2"), -0.52, f(-0.52), pos="ll"); ...
>textbox( ...
>  latex("f(x)=\begin{cases} x^3 & x>0 \\ x^2 & x \leq 0 \end{cases}"), ...
>  x=0.7, y=0.2);
```



## Interaksi pengguna

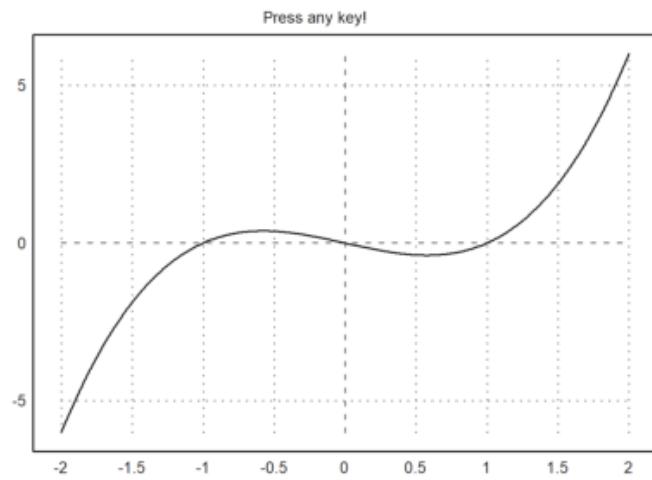
Saat memplot fungsi atau ekspresi, parameter `>user` memungkinkan pengguna untuk memperbesar dan menggeser plot dengan tombol kursor atau mouse. Pengguna dapat

- perbesar dengan + atau -
- pindahkan plot dengan tombol kursor
- pilih jendela plot dengan mouse
- atur ulang tampilan dengan spasi
- keluar dengan kembali

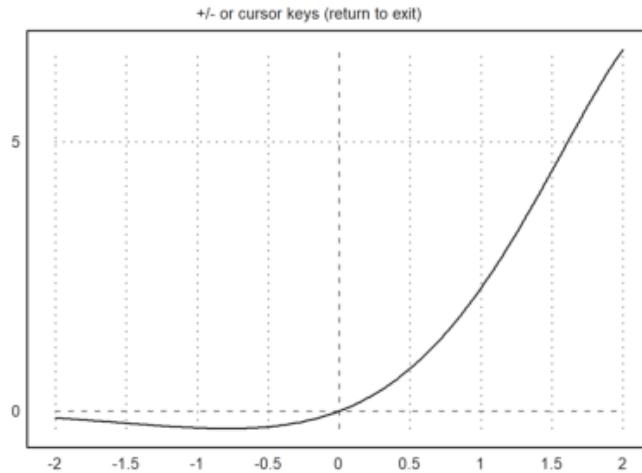
Tombol spasi akan mengatur ulang plot ke jendela plot asli.

Saat memplot data, flag `>user` hanya akan menunggu penekanan tombol.

```
>plot2d({{"x^3-a*x",a=1}},>user,title="Press any key!":
```



```
>plot2d("exp(x)*sin(x)",user=true, ...
> title="+/- or cursor keys (return to exit)":
```



Berikut ini menunjukkan cara interaksi pengguna tingkat lanjut (lihat tutorial tentang pemrograman untuk detailnya).

Fungsi bawaan mousedrag() menunggu event mouse atau keyboard. Ini melaporkan mouse ke bawah, mouse dipindahkan atau mouse ke atas, dan penekanan tombol. Fungsi dragpoints() memanfaatkan ini, dan memungkinkan pengguna menyeret titik mana pun dalam plot.

Kita membutuhkan fungsi plot terlebih dahulu. Sebagai contoh, kita interpolasi dalam 5 titik dengan polinomial. Fungsi harus diplot ke area plot tetap.

```
>function plotf(xp,yp,select) ...
```

```
d=interp(xp,yp);
plot2d("interpval(xp,d,x)" ; d, xp, r=2);
plot2d(xp,yp,>points,>add);
if select>0 then
```

```

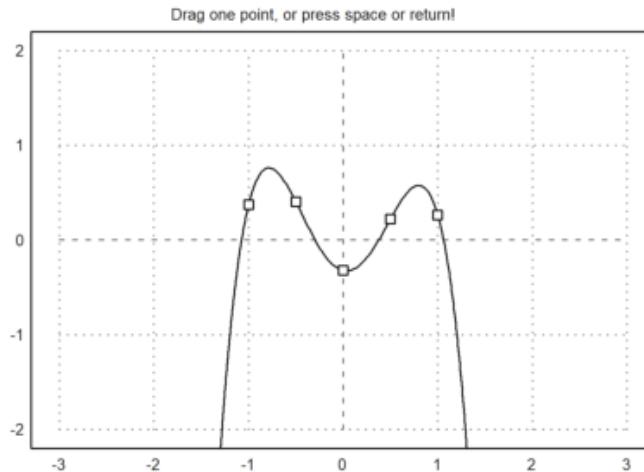
    plot2d(xp[select],yp[select],color=red,>points,>add);
endif;
title("Drag one point, or press space or return!");
endfunction

```

Perhatikan parameter titik koma di plot2d (d dan xp), yang diteruskan ke evaluasi fungsi interp(). Tanpa ini, kita harus menulis fungsi plotinterp() terlebih dahulu, mengakses nilai secara global.

Sekarang kita menghasilkan beberapa nilai acak, dan membiarkan pengguna menyeret poin.

```
>t=-1:0.5:1; dragpoints("plotf",t,random(size(t))-0.5):
```



Ada juga fungsi, yang memplot fungsi lain tergantung pada vektor parameter, dan memungkinkan pengguna menyesuaikan parameter ini.

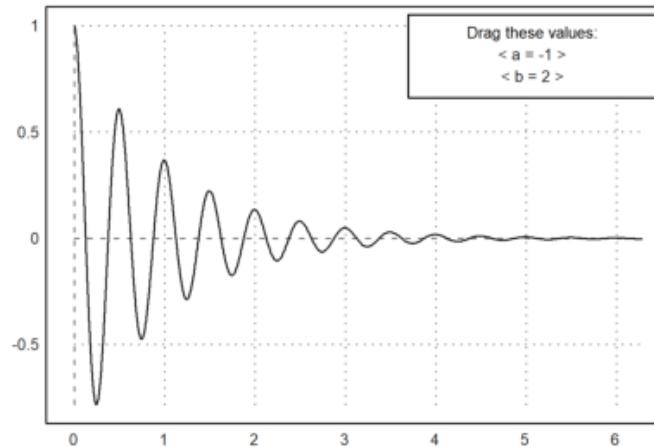
Pertama kita membutuhkan fungsi plot.

```
>function plotf([a,b]) := plot2d("exp(a*x)*cos(2pi*b*x)",0,2pi;a,b);
```

Kemudian kita membutuhkan nama untuk parameter, nilai awal dan matriks rentang nx2, opsional baris judul.

Ada slider interaktif, yang dapat mengatur nilai oleh pengguna. Fungsi dragvalues() menyediakan ini.

```
>dragvalues("plotf",["a","b"],[-1,2], [[-2,2];[1,10]], ...
> heading="Drag these values:",hcolor=black):
```

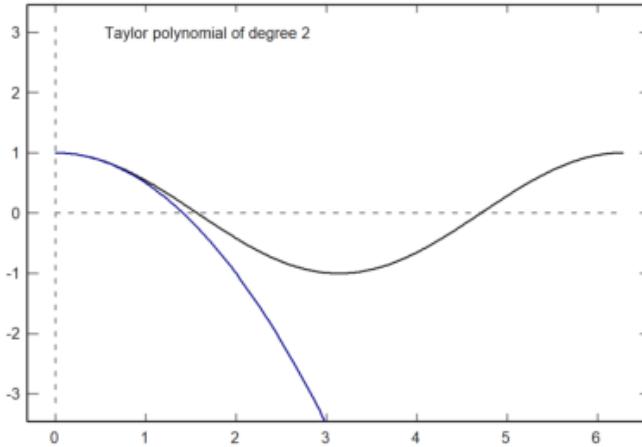


Dimungkinkan untuk membatasi nilai yang diseret ke bilangan bulat. Sebagai contoh, kita menulis fungsi plot, yang memplot polinomial Taylor derajat n ke fungsi kosinus.

```
>function plotf(n) ...
    plot2d("cos(x)",0,2pi,>square,grid=6);
    plot2d(&"taylor(cos(x),x,0,@n)",color=blue,>add);
    textbox("Taylor polynomial of degree "+n,0.1,0.02,style="t",>left);
endfunction
```

Sekarang kami mengizinkan derajat n bervariasi dari 0 hingga 20 dalam 20 pemberhentian. Hasil drag-values() digunakan untuk memplot sketsa dengan n ini, dan untuk memasukkan plot ke dalam buku catatan.

```
>nd=dragvalues("plotf","degree",2,[0,20],20,y=0.8, ...
>   heading="Drag the value:"); ...
>plotf(nd);
```



Berikut ini adalah demonstrasi sederhana dari fungsi tersebut. Pengguna dapat menggambar di atas jendela plot, meninggalkan jejak poin.

```
>function dragtest ...  
  
plot2d(None,r=1,title="Drag with the mouse, or press any key!");  
start=0;  
repeat  
    {flag,m,time}=mousedrag();  
    if flag==0 then return; endif;  
    if flag==2 then  
        hold on; mark(m[1],m[2]); hold off;  
        endif;  
    end  
endfunction
```

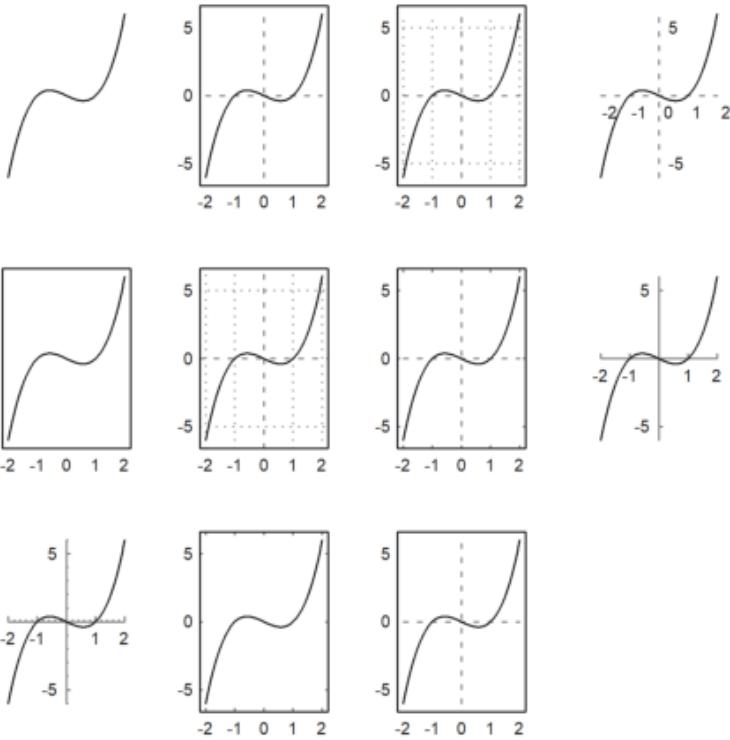
```
>dragtest // lihat hasilnya dan cobalah lakukan!
```

## Gaya Plot 2D

---

Secara default, EMT menghitung tick sumbu otomatis dan menambahkan label ke setiap tick. Ini dapat diubah dengan parameter grid. Gaya default sumbu dan label dapat dimodifikasi. Selain itu, label dan judul dapat ditambahkan secara manual. Untuk mengatur ulang ke gaya default, gunakan reset().

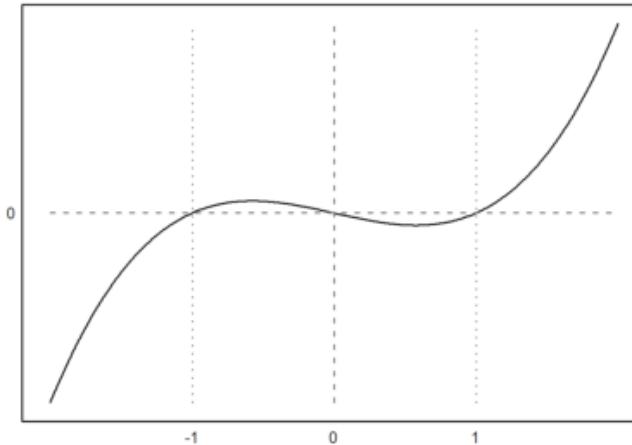
```
>aspect();
>figure(3,4); ...
> figure(1); plot2d("x^3-x",grid=0); ... // no grid, frame or axis
> figure(2); plot2d("x^3-x",grid=1); ... // x-y-axis
> figure(3); plot2d("x^3-x",grid=2); ... // default ticks
> figure(4); plot2d("x^3-x",grid=3); ... // x-y- axis with labels inside
> figure(5); plot2d("x^3-x",grid=4); ... // no ticks, only labels
> figure(6); plot2d("x^3-x",grid=5); ... // default, but no margin
> figure(7); plot2d("x^3-x",grid=6); ... // axes only
> figure(8); plot2d("x^3-x",grid=7); ... // axes only, ticks at axis
> figure(9); plot2d("x^3-x",grid=8); ... // axes only, finer ticks at axis
> figure(10); plot2d("x^3-x",grid=9); ... // default, small ticks inside
> figure(11); plot2d("x^3-x",grid=10); .... // no ticks, axes only
> figure(0):
```



Parameter <frame mematikan frame, dan framecolor=blue mengatur frame ke warna biru.

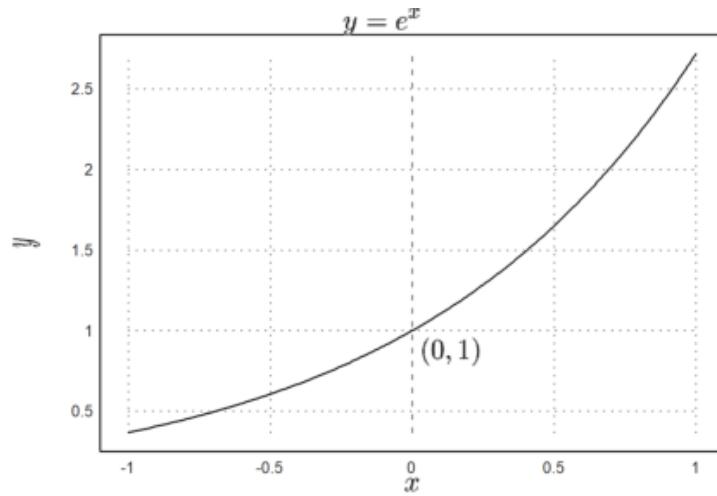
Jika Anda ingin centang sendiri, Anda dapat menggunakan style=0, dan menambahkan semuanya nanti.

```
>aspect(1.5);
>plot2d("x^3-x",grid=0); // plot
>frame; xgrid([-1,0,1]); ygrid(0); // add frame and grid
```



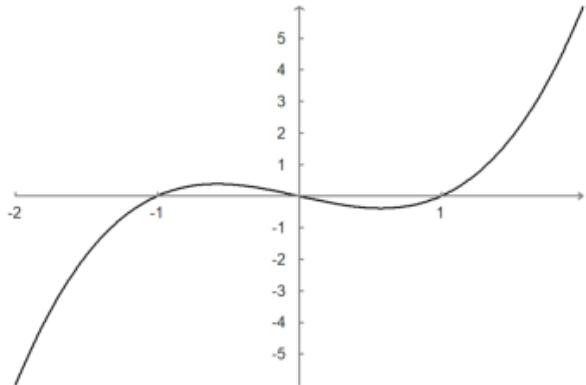
Untuk judul plot dan label sumbu, lihat contoh berikut.

```
>plot2d("exp(x)",-1,1);
>textcolor(black); // set the text color to black
>title(latex("y=e^x")); // title above the plot
>xlabel(latex("x")); // "x" for x-axis
>ylabel(latex("y"),>vertical); // vertical "y" for y-axis
>label(latex("(0,1)'),0,1,color=blue); // label a point
```



Sumbu dapat digambar secara terpisah dengan xaxis() dan yaxis().

```
>plot2d("x^3-x",<grid,<frame);
>xaxis(0,xx=-2:1,style="->"); yaxis(0,yy=-5:5,style="->");
```

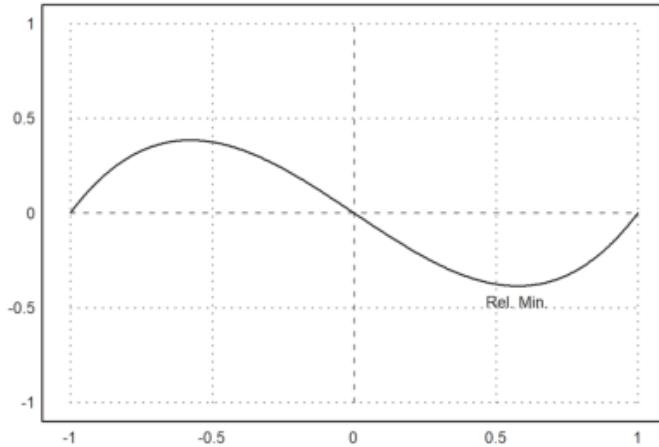


Teks pada plot dapat diatur dengan `label()`. Dalam contoh berikut, "lc" berarti tengah bawah. Ini mengatur posisi label relatif terhadap koordinat plot.

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

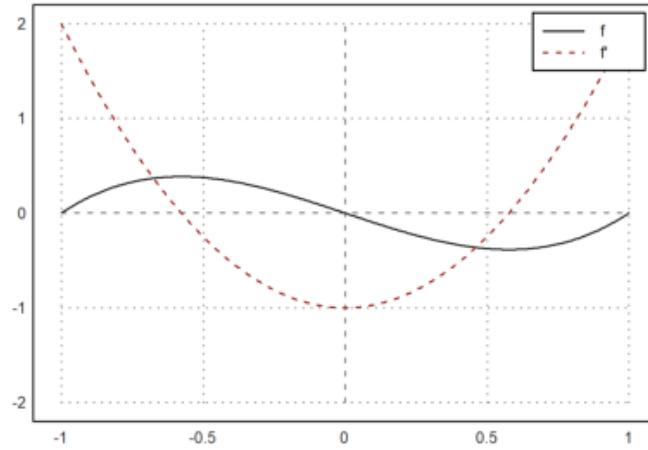
$$x^3 - x$$

```
>plot2d(f,-1,1,>square);
>x0=fmin(f,0,1); // compute point of minimum
>label("Rel. Min.",x0,f(x0),pos="lc"); // add a label there
```

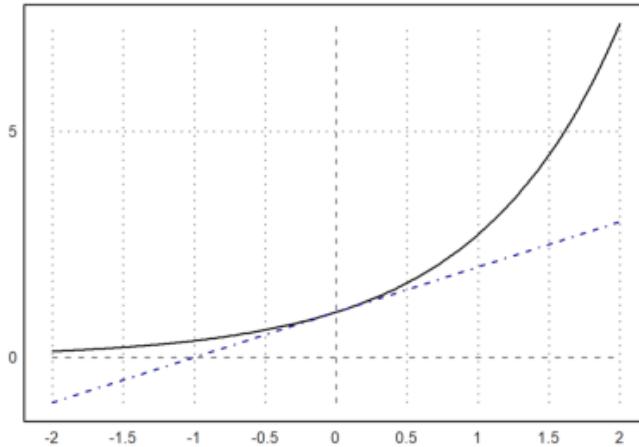


Ada juga kotak teks.

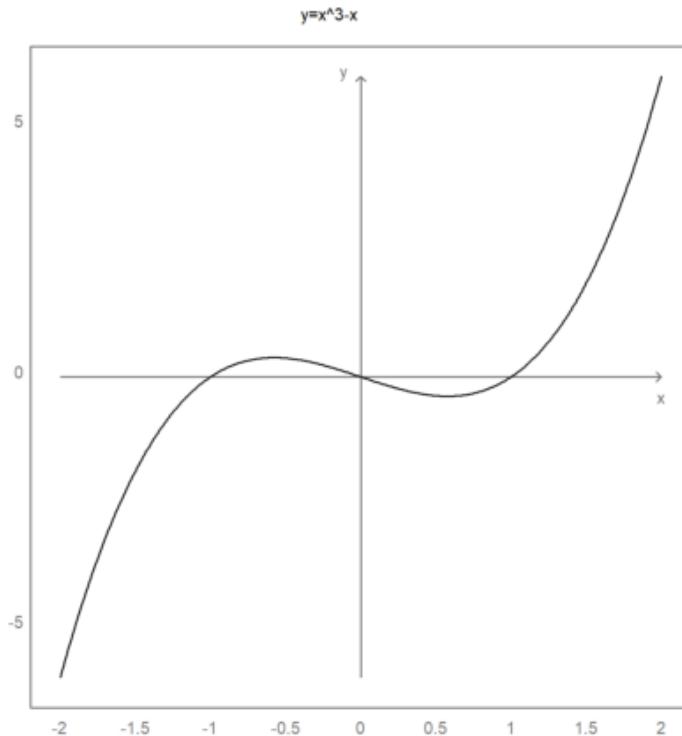
```
>plot2d(&f(x),-1,1,-2,2); // function  
>plot2d(&diff(f(x),x),>add,style="--",color=red); // derivative  
>labelbox(["f","f'"],["-","--"],[black,red]): // label box
```



```
>plot2d(["exp(x)","1+x"],color=[black,blue],style=["-","-.-"]):
```



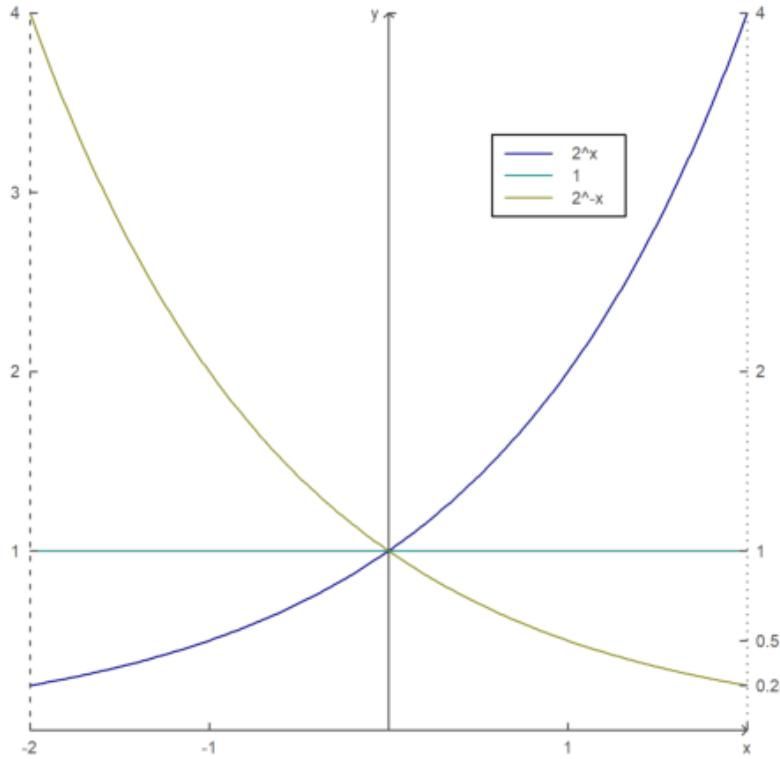
```
>gridstyle("->",color=gray,textcolor=gray,framecolor=gray);  ...
> plot2d("x^3-x",grid=1);    ...
> settitle("y=x^3-x",color=black); ...
> label("x",2,0,pos="bc",color=gray); ...
> label("y",0,6,pos="cl",color=gray); ...
> reset():
```



Untuk kontrol lebih, sumbu x dan sumbu y dapat dilakukan secara manual.

Perintah `fullwindow()` memperluas jendela plot karena kita tidak lagi membutuhkan tempat untuk label di luar jendela plot. Gunakan `shrinkwindow()` atau `reset()` untuk mengatur ulang ke default.

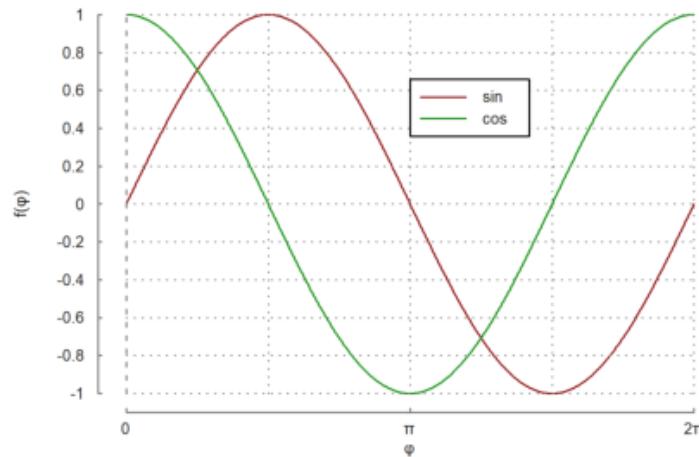
```
>fullwindow; ...
> gridstyle(color=darkgray,textcolor=darkgray); ...
> plot2d(["2^x","1","2^(-x)"],a=-2,b=2,c=0,d=4,<grid,color=4:6,<frame); ...
> xaxis(0,-2:1,style="->"); xaxis(0,2,"x",<axis); ...
> yaxis(0,4,"y",style="->"); ...
> yaxis(-2,1:4,>left); ...
> yaxis(2,2^(-2:2),style=".",<left); ...
> labelbox(["2^x","1","2^-x"],colors=4:6,x=0.8,y=0.2); ...
> reset:
```



Berikut adalah contoh lain, di mana string Unicode digunakan dan sumbu di luar area plot.

```
>aspect(1.5);
>plot2d(["sin(x)","cos(x)"],0,2pi,color=[red,green],<grid,<frame); ...
>xaxis(-1.1,(0:2)*pi,xt=["0",u"\u03c0;","u"2\u03c0;"],style="-",>ticks,>zero); ...
>xgrid((0:0.5:2)*pi,<ticks); ...
```

```
> yaxis(-0.1*pi,-1:0.2:1,style="-",>zero,>grid); ...
> labelbox(["sin","cos"],colors=[red,green],x=0.5,y=0.2,>left); ...
> xlabel(u"\u03c6"); ylabel(u"f(\u03c6)");
```



## Merencanakan Data 2D

---

Jika x dan y adalah vektor data, data ini akan digunakan sebagai koordinat x dan y dari suatu kurva. Dalam hal ini, a, b, c, dan d, atau radius r dapat ditentukan, atau jendela plot akan menyesuaikan secara otomatis dengan data. Atau, >persegi dapat diatur untuk menjaga rasio aspek persegi.

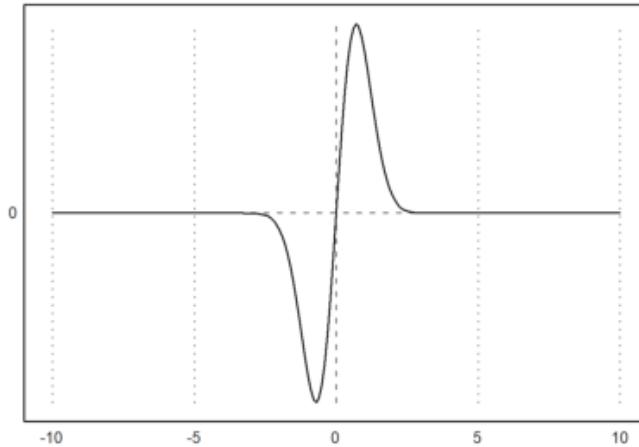
Memplot ekspresi hanyalah singkatan untuk plot data. Untuk plot data, Anda memerlukan satu atau beberapa baris nilai x, dan satu atau beberapa baris nilai y. Dari rentang dan nilai-x, fungsi plot2d akan menghitung data yang akan diplot, secara default dengan evaluasi fungsi yang adaptif. Untuk plot titik gunakan ">titik", untuk garis campuran dan titik gunakan ">tambahan".

Tapi Anda bisa memasukkan data secara langsung.

- Gunakan vektor baris untuk x dan y untuk satu fungsi.
- Matriks untuk x dan y diplot baris demi baris.

Berikut adalah contoh dengan satu baris untuk x dan y.

```
>x=-10:0.1:10; y=exp(-x^2)*x; plot2d(x,y);
```



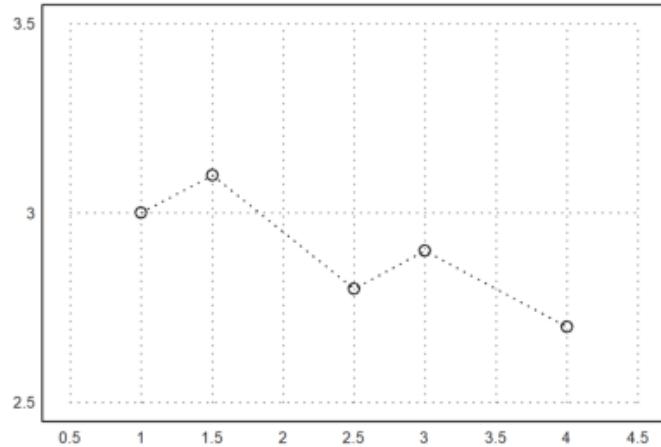
Data juga dapat diplot sebagai titik. Gunakan poin=true untuk ini. Plotnya bekerja seperti poligon, tetapi hanya menggambar sudut-sudutnya.

- style="...": Pilih dari "[", "<>", "o", ".", "..", "+", "\*", "[", "<>", "o", "..", "", "|".

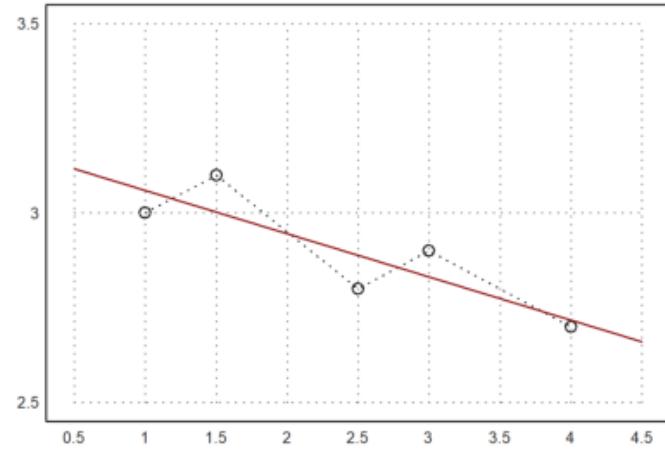
Untuk memplot set poin gunakan >points. Jika warna adalah vektor warna, setiap titik mendapat warna yang berbeda. Untuk matriks koordinat dan vektor kolom, warna berlaku untuk baris matriks.

Parameter >addpoints menambahkan titik ke segmen garis untuk plot data.

```
>xdata=[1,1.5,2.5,3,4]; ydata=[3,3.1,2.8,2.9,2.7]; // data  
>plot2d(xdata,ydata,a=0.5,b=4.5,c=2.5,d=3.5,style="."); // lines  
>plot2d(xdata,ydata,>points,>add,style="o"); // add points
```



```
>p=polyfit(xdata,ydata,1); // get regression line  
>plot2d("polyval(p,x)",>add,color=red); // add plot of line
```



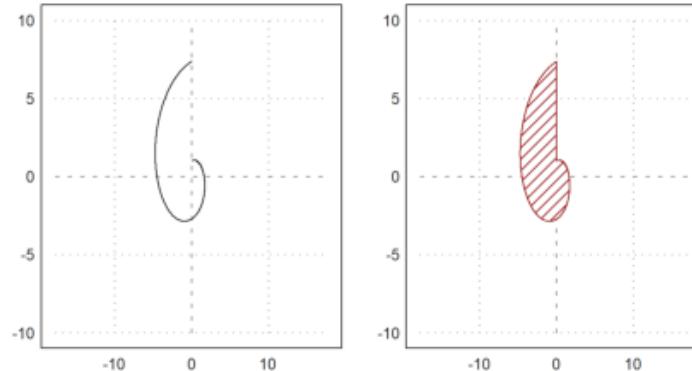
## Menggambar Daerah Yang Dibatasi Kurva

Plot data benar-benar poligon. Kita juga dapat memplot kurva atau kurva terisi.

- `terisi=benar` mengisi plot.
- `style="...":` Pilih dari "", "/", "\", "\/".
- `fillcolor:` Lihat di atas untuk warna yang tersedia.

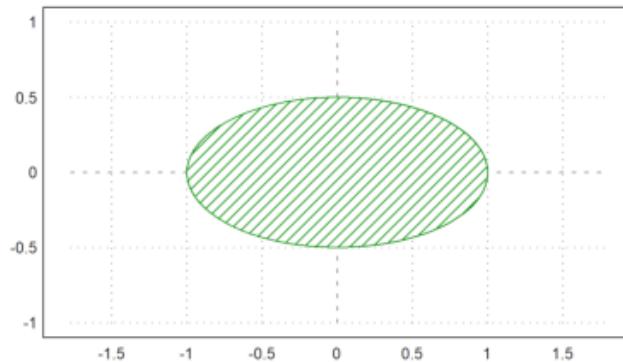
Warna isian ditentukan oleh argumen "fillcolor", dan pada <outline opsional mencegah menggambar batas untuk semua gaya kecuali yang default.

```
>t=linspace(0,2pi,1000); // parameter for curve  
>x=sin(t)*exp(t/pi); y=cos(t)*exp(t/pi); // x(t) and y(t)  
>figure(1,2); aspect(16/9)  
>figure(1); plot2d(x,y,r=10); // plot curve  
>figure(2); plot2d(x,y,r=10,>filled,style="/",fillcolor=red); // fill curve  
>figure(0):
```

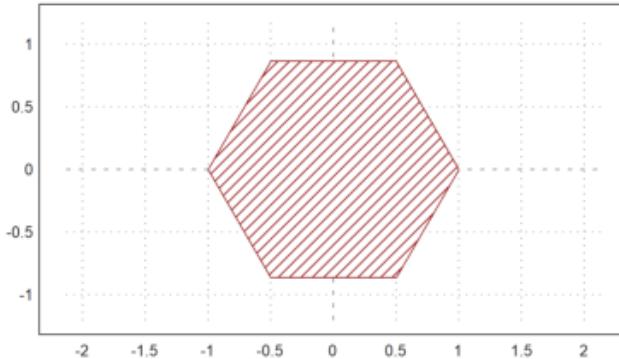


Dalam contoh berikut kami memplot elips terisi dan dua segi enam terisi menggunakan kurva tertutup dengan 6 titik dengan gaya isian berbeda.

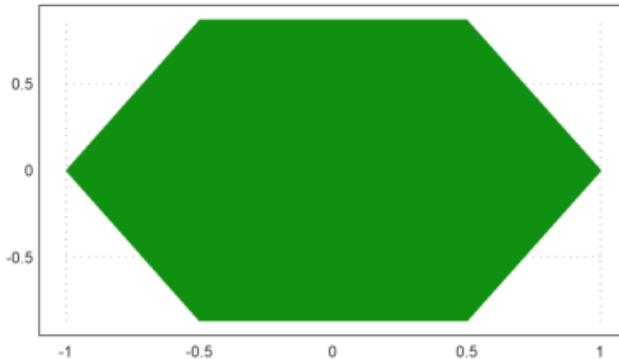
```
>x=linspace(0,2pi,1000); plot2d(sin(x),cos(x)*0.5,r=1,>filled,style="/"):
```



```
>t=linspace(0,2pi,6); ...
>plot2d(cos(t),sin(t),>filled,style="/",fillcolor=red,r=1.2):
```

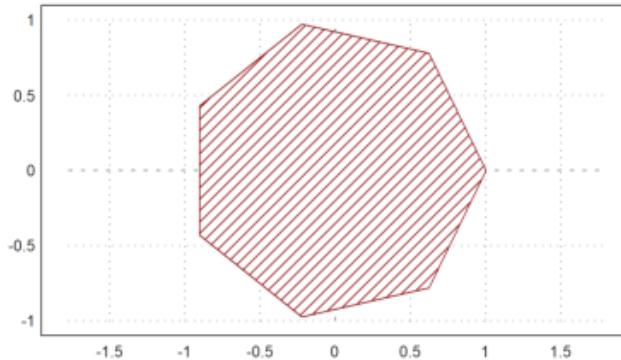


```
>t=linspace(0,2pi,6); plot2d(cos(t),sin(t),>filled,style="#");
```



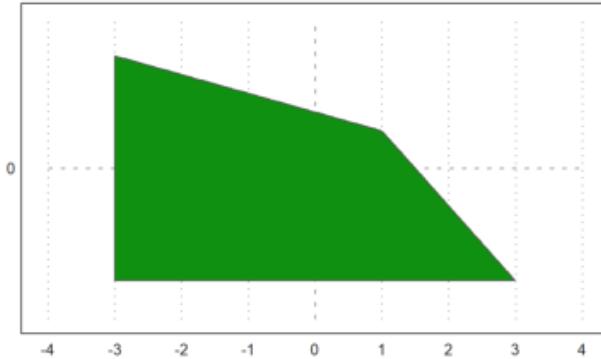
Contoh lainnya adalah segi empat, yang kita buat dengan 7 titik pada lingkaran satuan.

```
>t=linspace(0,2pi,7); ...  
> plot2d(cos(t),sin(t),r=1,>filled,style="/",fillcolor=red):
```



Berikut ini adalah himpunan nilai maksimal dari empat kondisi linier yang kurang dari atau sama dengan 3. Ini adalah  $A[k].v \leq 3$  untuk semua baris  $A$ . Untuk mendapatkan sudut yang bagus, kita menggunakan  $n$  yang relatif besar.

```
>A=[2,1;1,2;-1,0;0,-1];  
>function f(x,y) := max([x,y].A');  
>plot2d("f",r=4,level=[0;3],color=green,n=111):
```

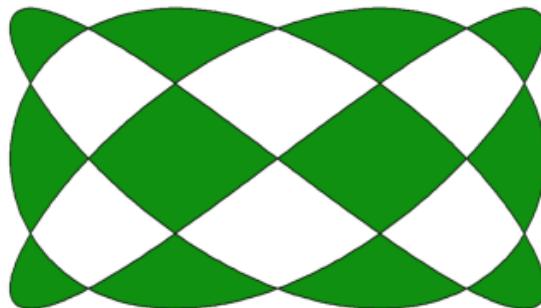


Poin utama dari bahasa matriks adalah memungkinkan untuk menghasilkan tabel fungsi dengan mudah.

```
>t=linspace(0,2pi,1000); x=cos(3*t); y=sin(4*t);
```

Kami sekarang memiliki vektor x dan y nilai. `plot2d()` dapat memplot nilai-nilai ini sebagai kurva yang menghubungkan titik-titik. Plotnya bisa diisi. Pada kasus ini ini menghasilkan hasil yang bagus karena aturan lilitan, yang digunakan untuk isi.

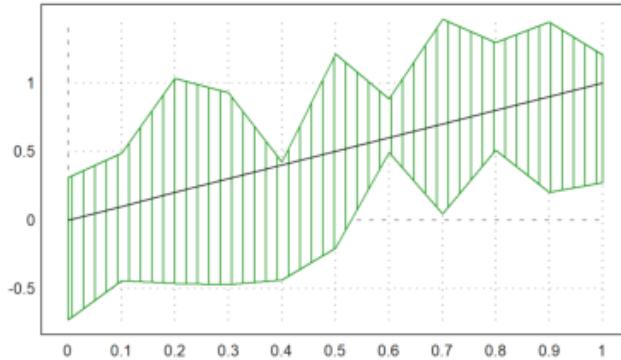
```
>plot2d(x,y,<grid,<frame,>filled):
```



Sebuah vektor interval diplot terhadap nilai x sebagai daerah terisi antara nilai interval bawah dan atas.

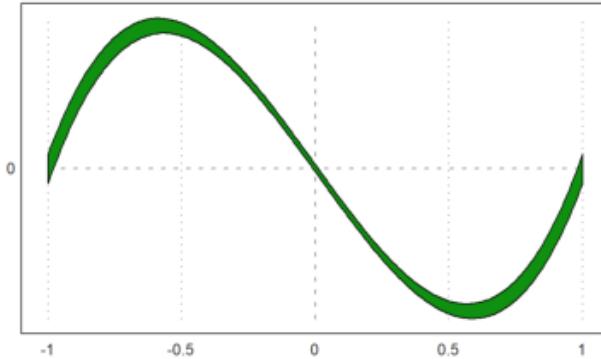
Hal ini dapat berguna untuk memplot kesalahan perhitungan. Tapi itu bisa juga digunakan untuk memplot kesalahan statistik.

```
>t=0:0.1:1; ...
> plot2d(t,interval(t-random(size(t)),t+random(size(t))),style="|");
> plot2d(t,t,add=true):
```



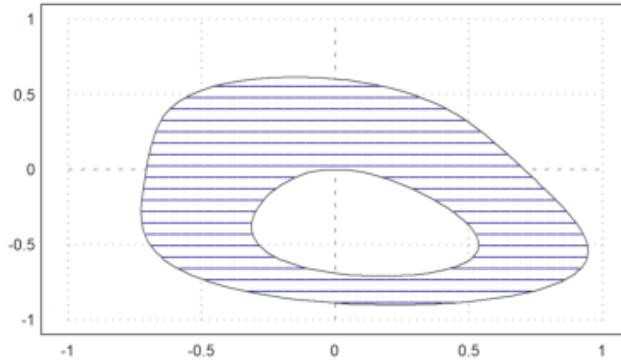
Jika  $x$  adalah vektor yang diurutkan, dan  $y$  adalah vektor interval, maka `plot2d` akan memplot rentang interval yang terisi dalam bidang. Gaya isian sama dengan gaya poligon.

```
>t=-1:0.01:1; x=~-t-0.01,t+0.01~; y=x^3-x;  
>plot2d(t,y);
```



Jika  $x$  adalah vektor yang diurutkan, dan  $y$  adalah vektor interval, maka `plot2d` akan memplot rentang interval yang terisi dalam bidang. Gaya isian sama dengan gaya poligon.

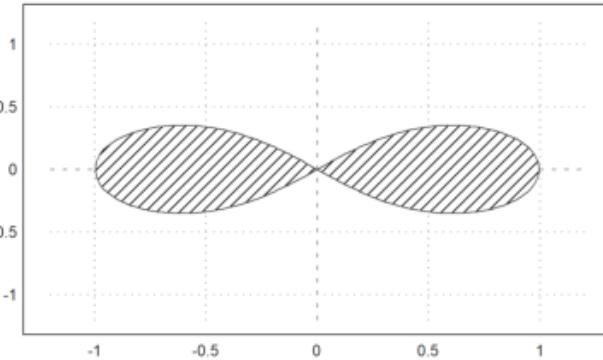
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=[0;1],style="-",color=blue); // 0 <= f(x,y) <= 1
```



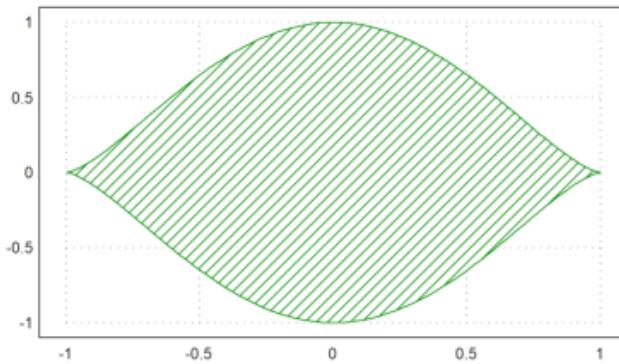
Kami juga dapat mengisi rentang nilai seperti

$$-1 \leq (x^2 + y^2)^2 - x^2 + y^2 \leq 0.$$

```
>plot2d("(x^2+y^2)^2-x^2+y^2",r=1.2,level=[-1;0],style="/"):
```



```
>plot2d("cos(x)","sin(x)^3",xmin=0,xmax=2pi,>filled,style="/"):
```



## Grafik Fungsi Parametrik

---

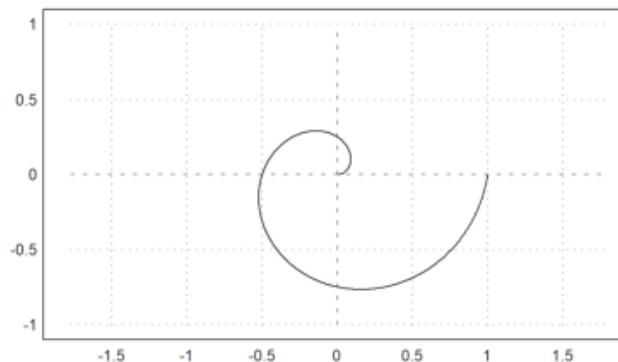
Nilai-x tidak perlu diurutkan. (x,y) hanya menggambarkan kurva. Jika x diurutkan, kurva tersebut merupakan grafik fungsi.

Dalam contoh berikut, kami memplot spiral

$$\gamma(t) = t \cdot (\cos(2\pi t), \sin(2\pi t))$$

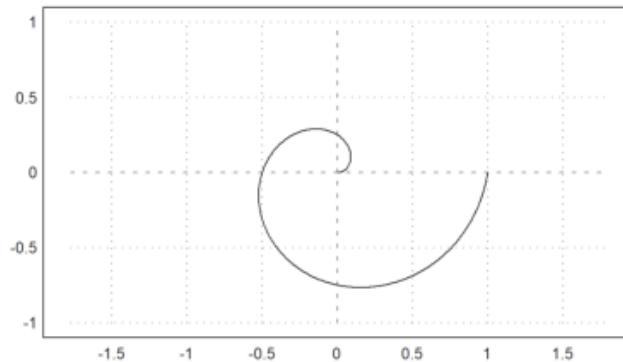
Kita perlu menggunakan banyak titik untuk tampilan yang halus atau fungsi adaptif() untuk mengevaluasi ekspresi (lihat fungsi adaptif() untuk lebih jelasnya).

```
>t=linspace(0,1,1000); ...
>plot2d(t*cos(2*pi*t),t*sin(2*pi*t),r=1):
```

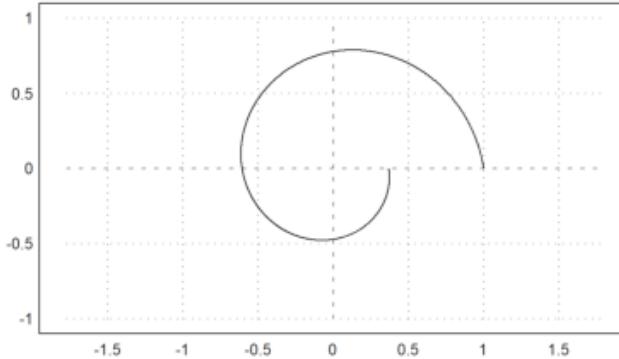


Atau, dimungkinkan untuk menggunakan dua ekspresi untuk kurva. Berikut ini plot kurva yang sama seperti di atas.

```
>plot2d("x*cos(2*pi*x)","x*sin(2*pi*x)",xmin=0,xmax=1,r=1):
```



```
>t=linspace(0,1,1000); r=exp(-t); x=r*cos(2pi*t); y=r*sin(2pi*t);
>plot2d(x,y,r=1):
```



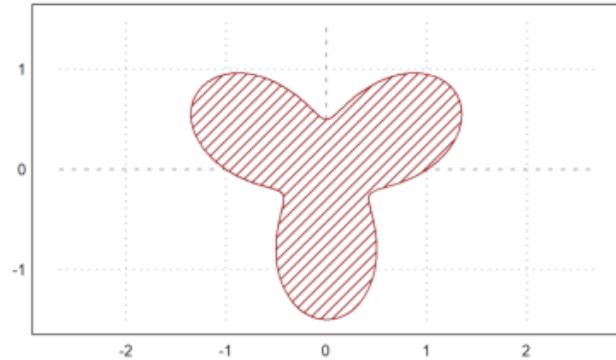
Dalam contoh berikutnya, kami memplot kurva

$$\gamma(t) = (r(t) \cos(t), r(t) \sin(t))$$

dengan

$$r(t) = 1 + \frac{\sin(3t)}{2}.$$

```
>t=linspace(0,2pi,1000); r=1+sin(3*t)/2; x=r*cos(t); y=r*sin(t); ...
>plot2d(x,y,>filled,fillcolor=red,style="/",r=1.5):
```



## Menggambar Grafik Bilangan Kompleks

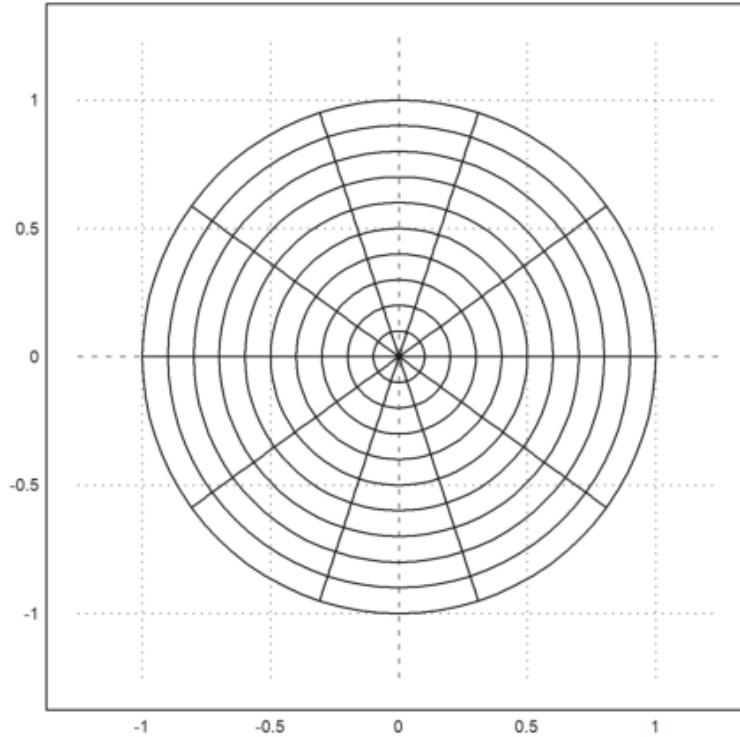
---

Array bilangan kompleks juga dapat diplot. Kemudian titik-titik grid akan terhubung. Jika sejumlah garis kisi ditentukan (atau vektor garis kisi 1x2) dalam argumen cgrid, hanya garis kisi tersebut yang terlihat.

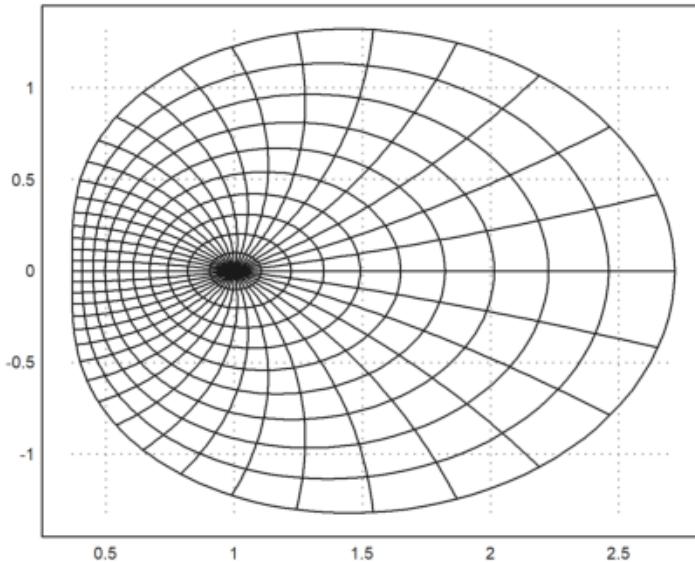
Matriks bilangan kompleks akan secara otomatis diplot sebagai kisi di bidang kompleks.

Dalam contoh berikut, kami memplot gambar lingkaran satuan di bawah fungsi eksponensial. Parameter cgrid menyembunyikan beberapa kurva grid.

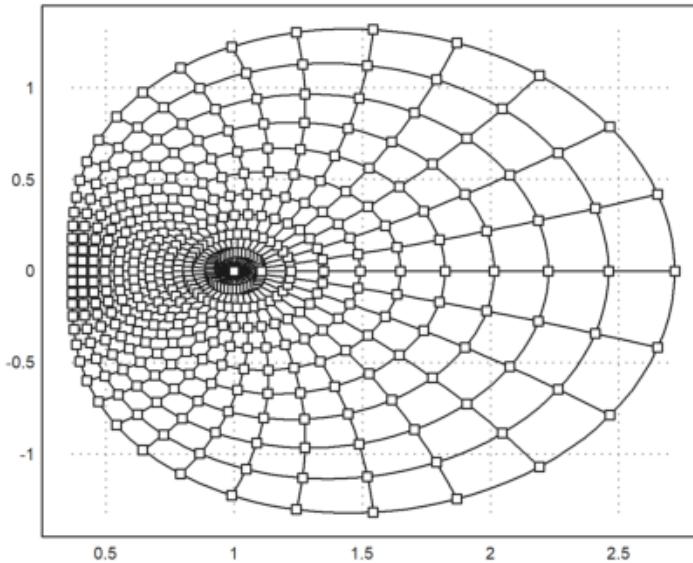
```
>aspect(); r=linspace(0,1,50); a=linspace(0,2pi,80)'; z=r*exp(I*a);...
>plot2d(z,a=-1.25,b=1.25,c=-1.25,d=1.25,cgrid=10);
```



```
>aspect(1.25); r=linspace(0,1,50); a=linspace(0,2pi,200)'; z=r*exp(I*a);
>plot2d(exp(z),cgrid=[40,10]):
```



```
>r=linspace(0,1,10); a=linspace(0,2pi,40)'; z=r*exp(I*a);
>plot2d(exp(z),>points,>add):
```

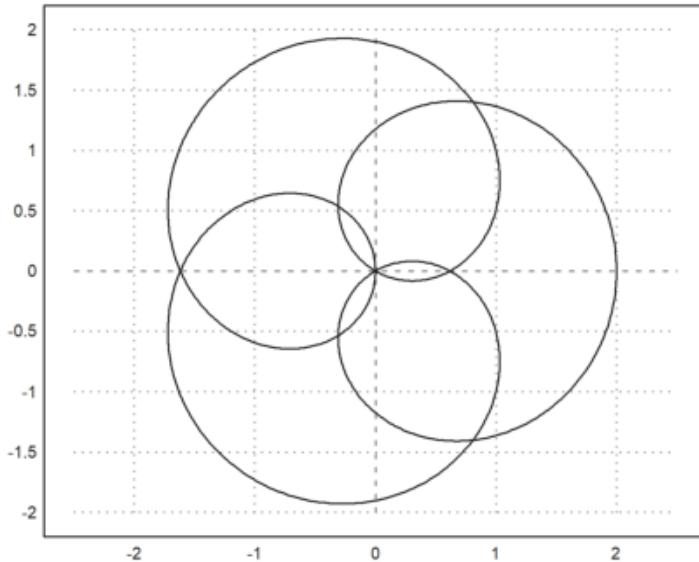


Sebuah vektor bilangan kompleks secara otomatis diplot sebagai kurva pada bidang kompleks dengan bagian real dan bagian imajiner.

Dalam contoh, kami memplot lingkaran satuan dengan

$$\gamma(t) = e^{it}$$

```
>t=linspace(0,2pi,1000); ...
>plot2d(exp(I*t)+exp(4*I*t),r=2):
```



## Plot Statistik

---

Ada banyak fungsi yang dikhususkan pada plot statistik. Salah satu plot yang sering digunakan adalah plot kolom.

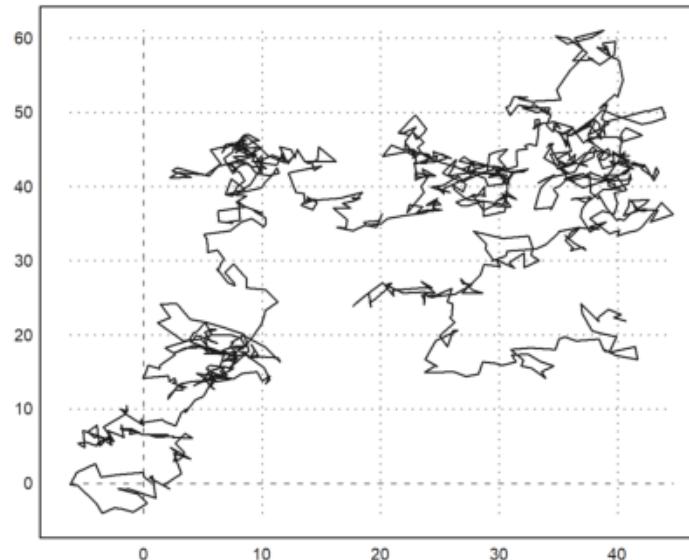
Jumlah kumulatif dari nilai terdistribusi 0-1-normal menghasilkan jalan acak.

```
>plot2d(cumsum(randnormal(1,1000))):
```

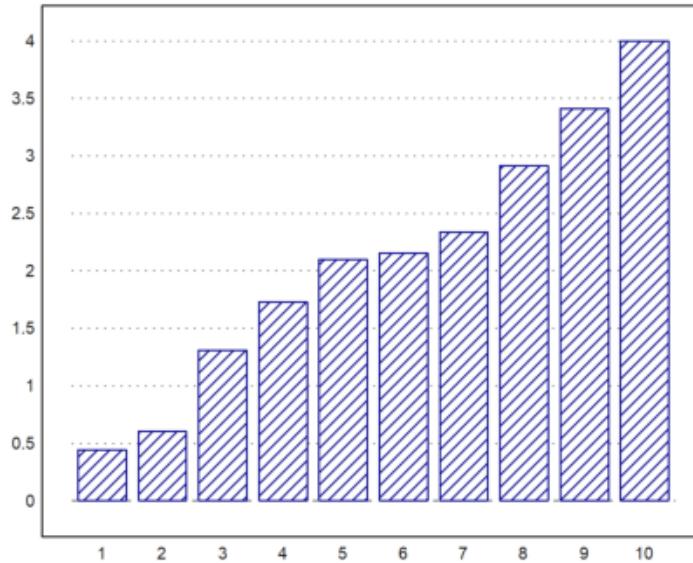


Menggunakan dua baris menunjukkan jalan dalam dua dimensi.

```
>X=cumsum(randnormal(2,1000)); plot2d(X[1],X[2]):
```

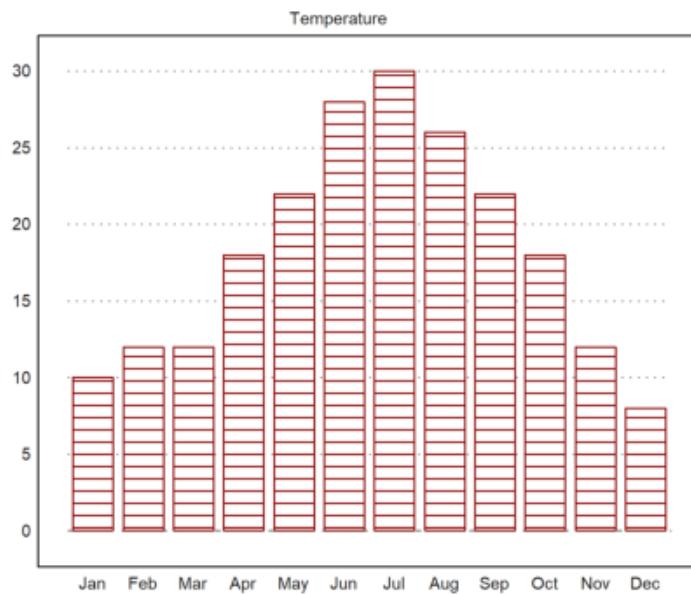


```
>columnsplot(cumsum(random(10)),style="/",color=blue):
```

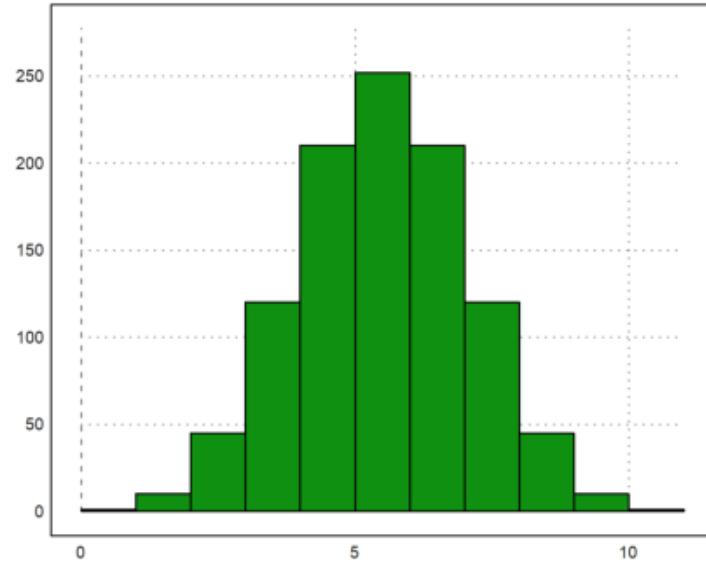


Itu juga dapat menampilkan string sebagai label.

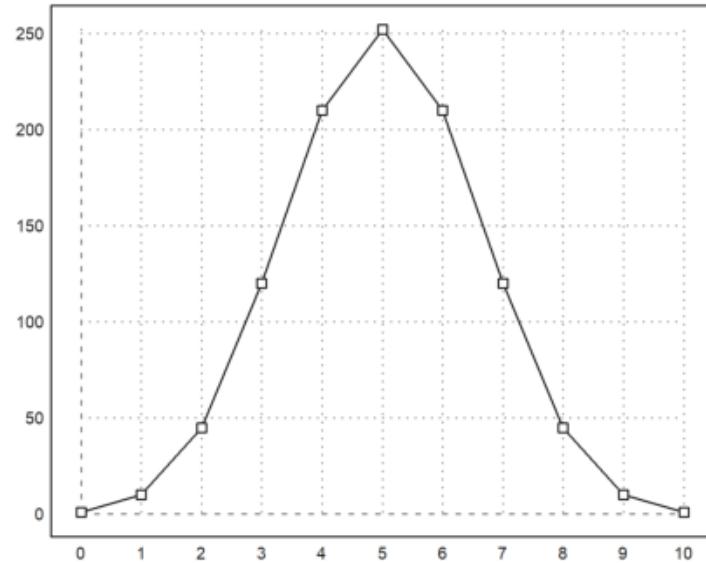
```
>months=["Jan","Feb","Mar","Apr","May","Jun", ...  
> "Jul","Aug","Sep","Oct","Nov","Dec"];  
>values=[10,12,12,18,22,28,30,26,22,18,12,8];  
>columnspplot(values,lab=months,color=red,style="-");  
>title("Temperature");
```



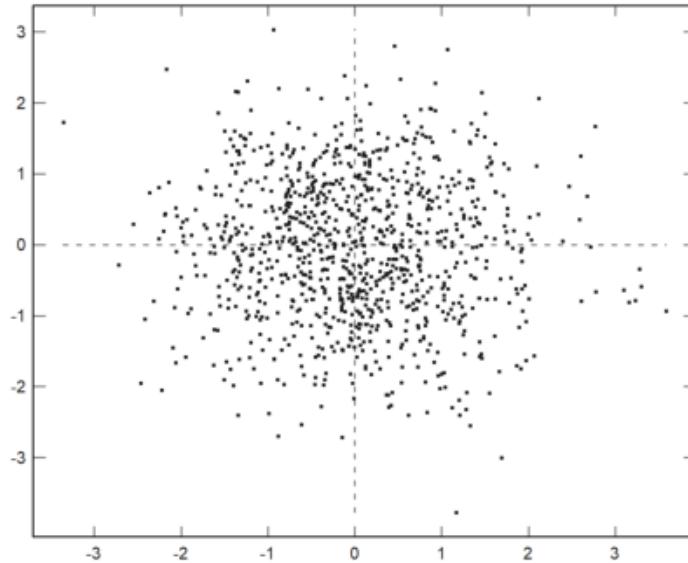
```
>k=0:10;  
>plot2d(k,bin(10,k),>bar):
```



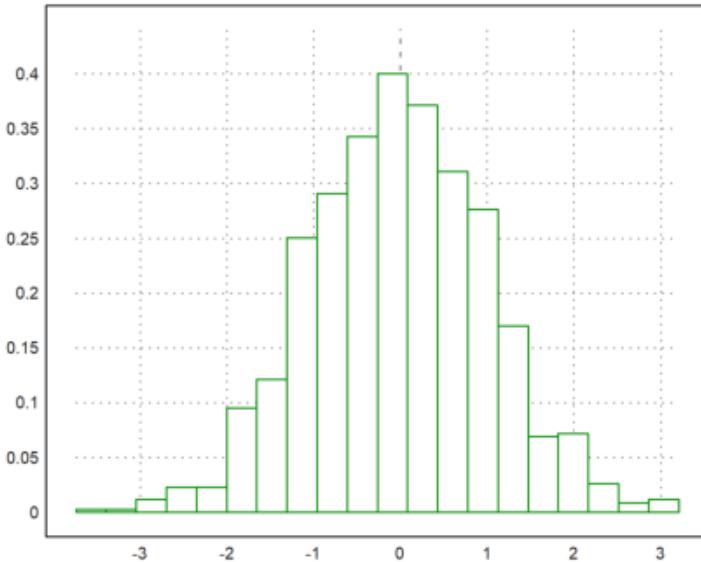
```
>plot2d(k,bin(10,k)); plot2d(k,bin(10,k),>points,>add):
```



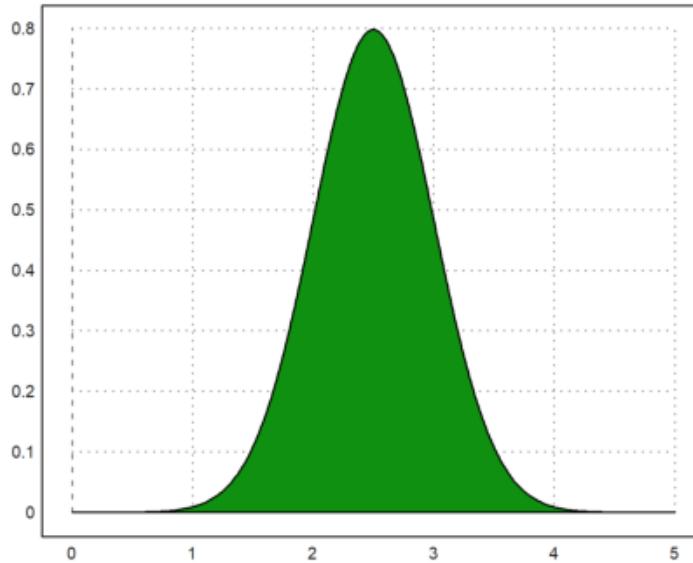
```
>plot2d(normal(1000),normal(1000),>points,grid=6,style=".."):
```



```
>plot2d(normal(1,1000),>distribution,style="0"):
```

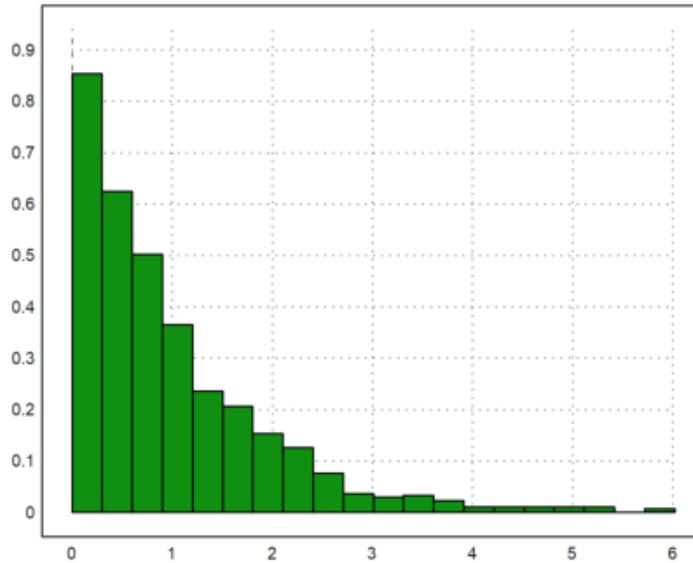


```
>plot2d("qnormal",0,5;2.5,0.5,>filled):
```



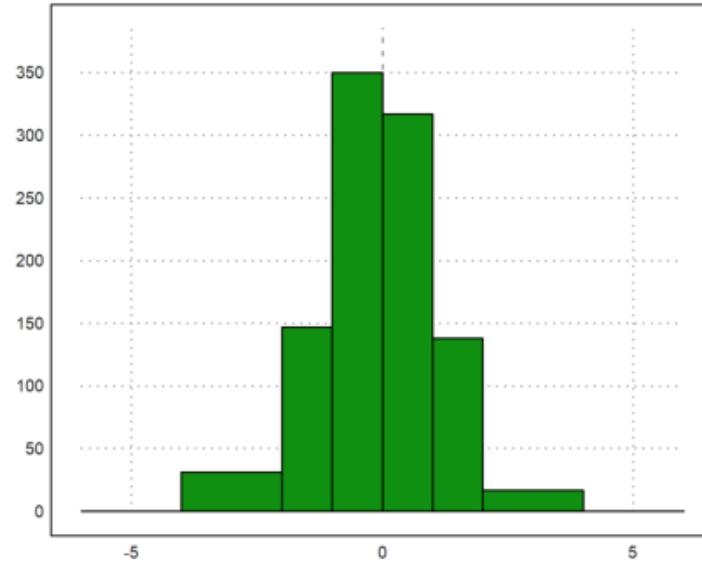
Untuk memplot distribusi statistik eksperimental, Anda dapat menggunakan `distribution=n` dengan `plot2d`.

```
>w=randexponential(1,1000); // exponential distribution  
>plot2d(w,>distribution); // or distribution=n with n intervals
```



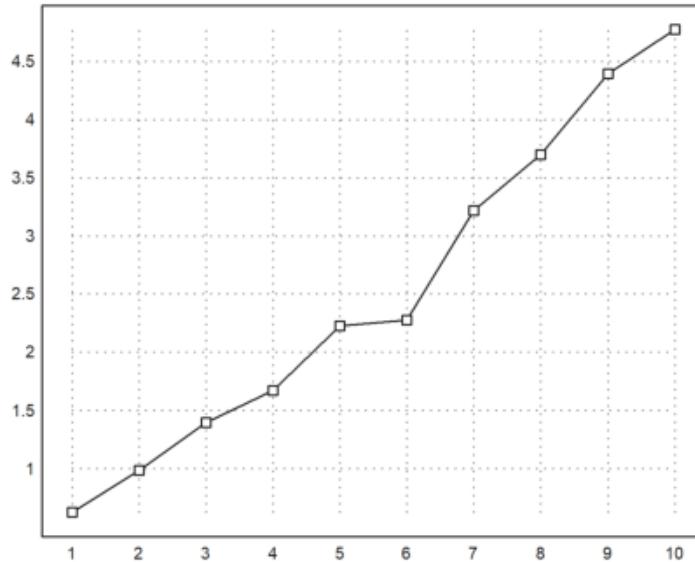
Atau Anda dapat menghitung distribusi dari data dan memplot hasilnya dengan >bar di plot3d, atau dengan plot kolom.

```
>w=normal(1000); // 0-1-normal distribution
>{x,y}=histo(w,10,v=[-6,-4,-2,-1,0,1,2,4,6]); // interval bounds v
>plot2d(x,y,>bar):
```

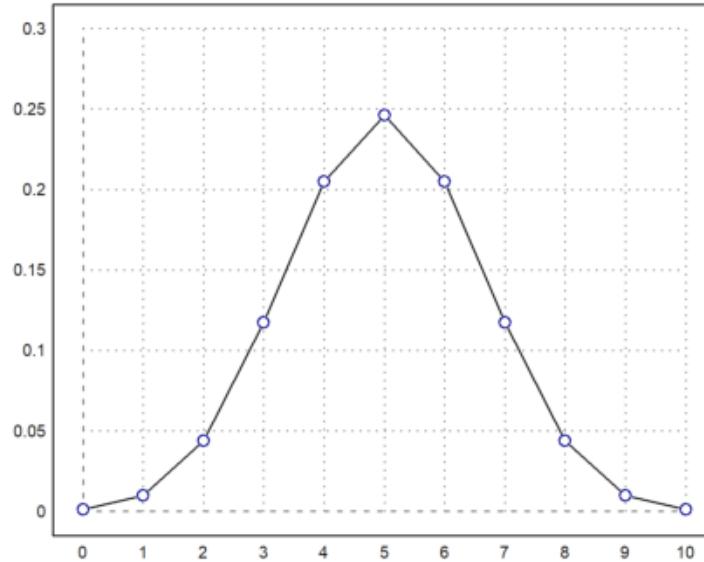


Fungsi statplot() menyetel gaya dengan string sederhana.

```
>statplot(1:10,cumsum(random(10)),"b"):
```



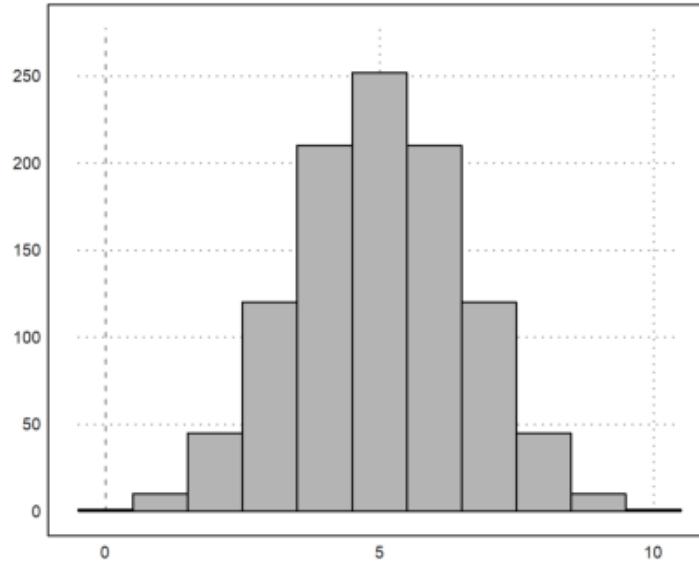
```
>n=10; i=0:n; ...
>plot2d(i,bin(n,i)/2^n,a=0,b=10,c=0,d=0.3); ...
>plot2d(i,bin(n,i)/2^n,points=true,style="ow",add=true,color=blue):
```



Selain itu, data dapat diplot sebagai batang. Dalam hal ini, x harus diurutkan dan satu elemen lebih panjang dari y. Bilah akan memanjang dari  $x[i]$  ke  $x[i+1]$  dengan nilai  $y[i]$ . Jika x memiliki ukuran yang sama dengan y, maka akan diperpanjang satu elemen dengan spasi terakhir.

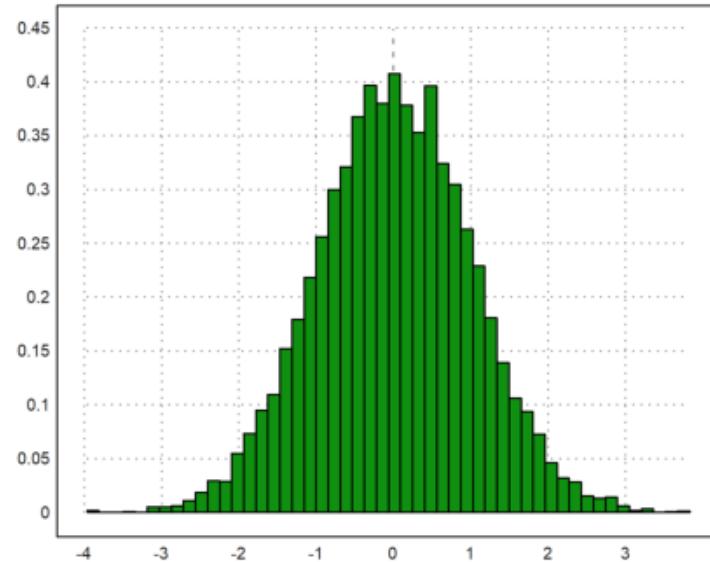
Gaya isian dapat digunakan seperti di atas.

```
>n=10; k=bin(n,0:n); ...
>plot2d(-0.5:n+0.5,k,bar=true,fillcolor=lightgray):
```

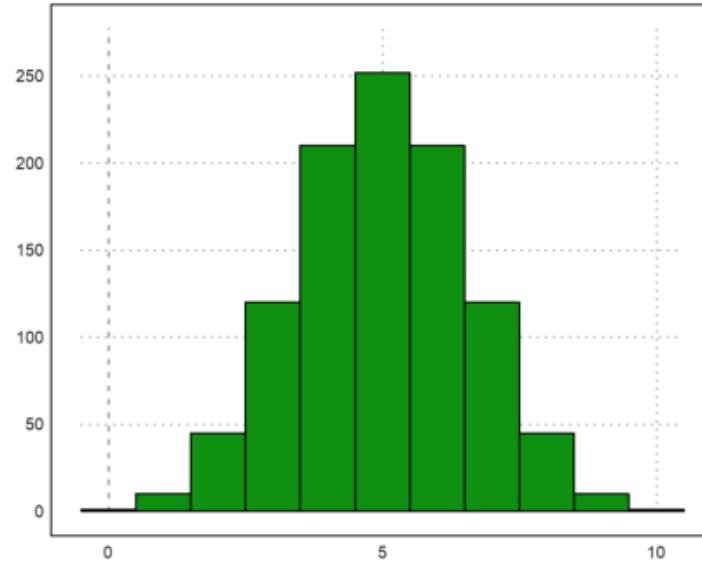


Data untuk plot batang (bar=1) dan histogram (histogram=1) dapat dinyatakan secara eksplisit dalam xv dan yv, atau dapat dihitung dari distribusi empiris dalam xv dengan >distribusi (atau distribusi=n). Histogram nilai xv akan dihitung secara otomatis dengan >histogram. Jika >genap ditentukan, nilai xv akan dihitung dalam interval bilangan bulat.

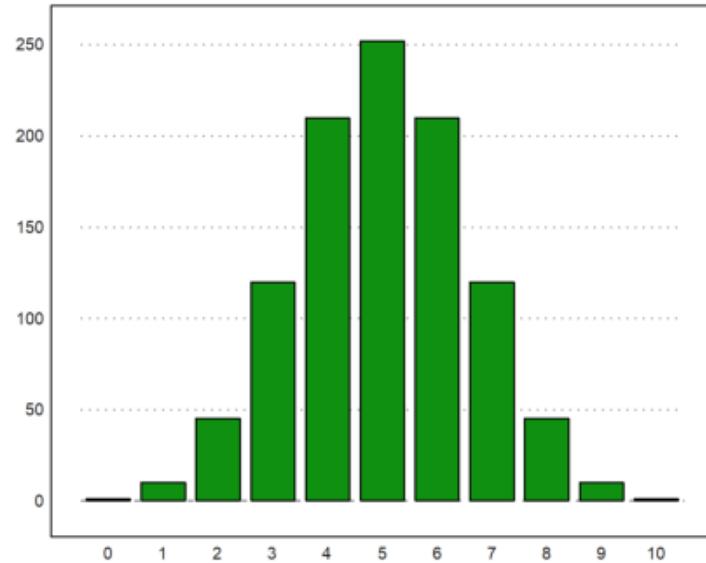
```
>plot2d(normal(10000),distribution=50):
```



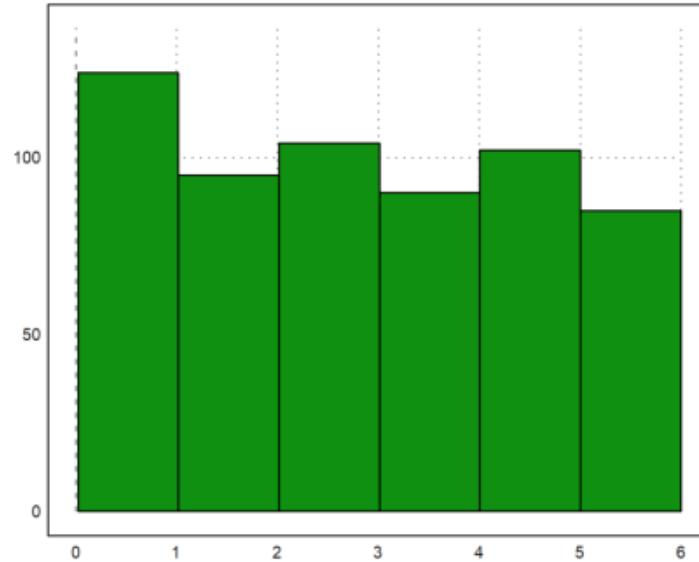
```
>k=0:10; m=bin(10,k); x=(0:11)-0.5; plot2d(x,m,>bar):
```



```
>columnsplot(m,k):
```

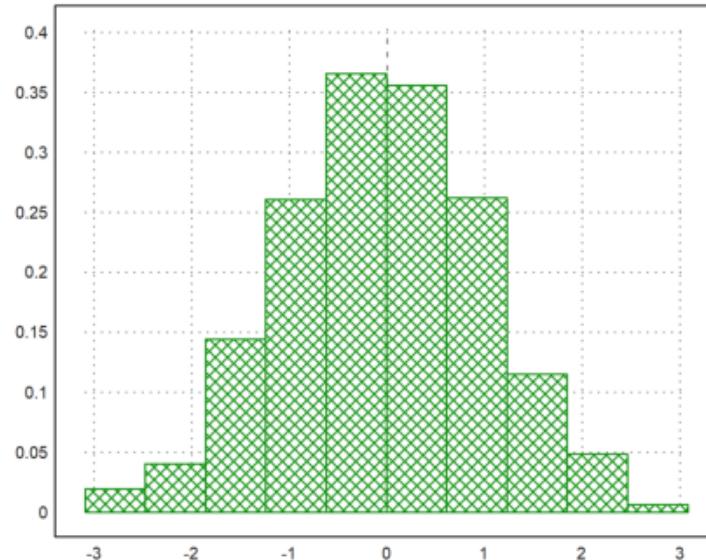


```
>plot2d(random(600)*6,histogram=6):
```



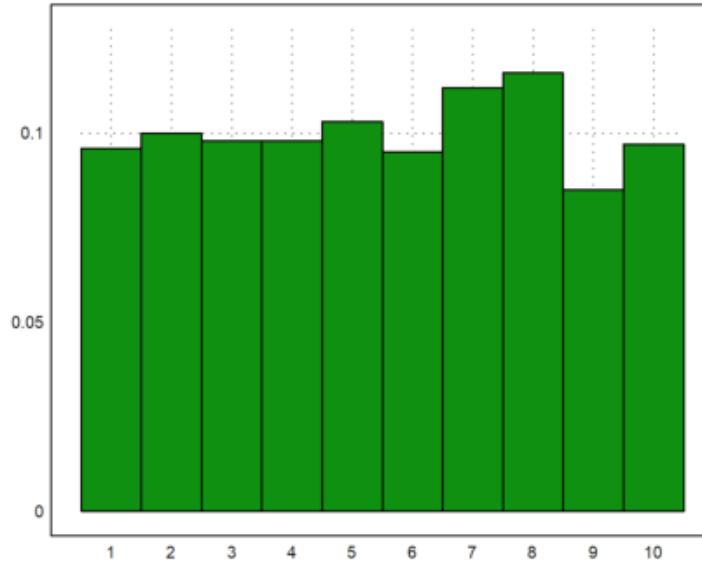
Untuk distribusi, ada parameter `distribution=n`, yang menghitung nilai secara otomatis dan mencetak distribusi relatif dengan n sub-interval.

```
>plot2d(normal(1,1000),distribution=10,style="/"):
```



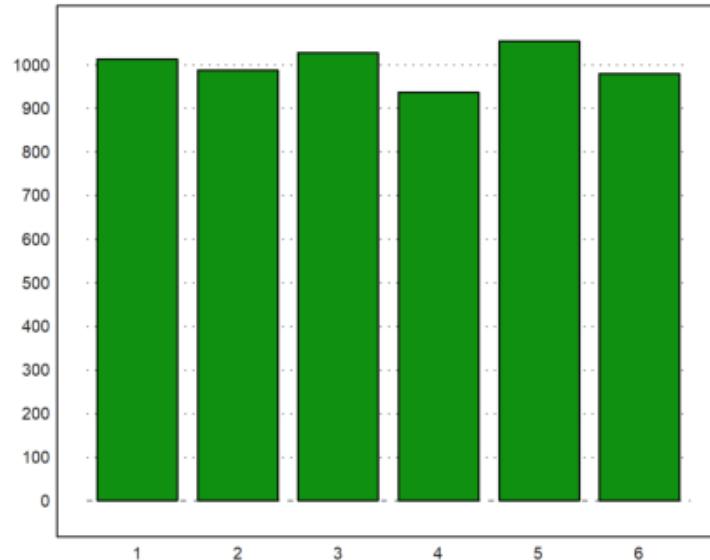
Dengan parameter even=true, ini akan menggunakan interval integer.

```
>plot2d(intrandom(1,1000,10),distribution=10,even=true):
```

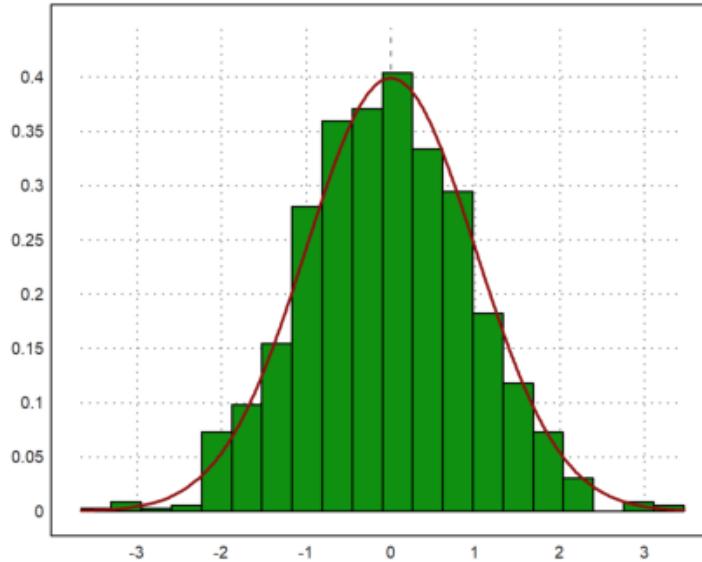


Perhatikan bahwa ada banyak plot statistik, yang mungkin berguna. Silahkan lihat tutorial tentang statistik.

```
>columnsplot(getmultiplicities(1:6,intrandom(1,6000,6))):
```

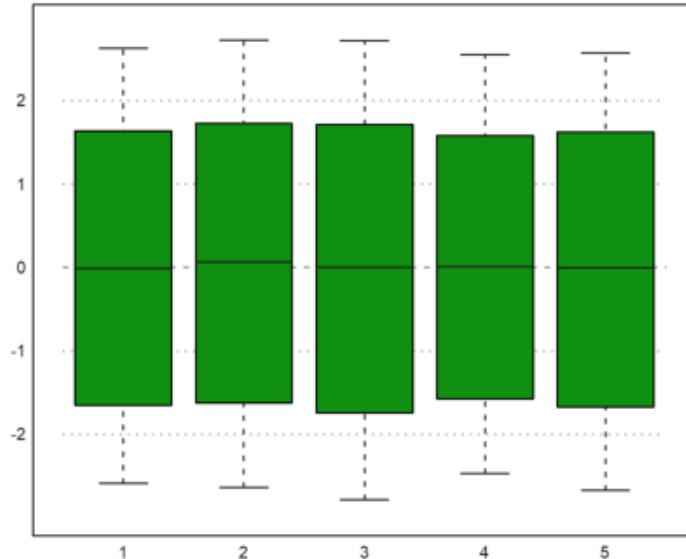


```
>plot2d(normal(1,1000),>distribution); ...
> plot2d("qnormal(x)",color=red,thickness=2,>add):
```



Ada juga banyak plot khusus untuk statistik. Boxplot menunjukkan kuartil dari distribusi ini dan banyak outlier. Menurut definisi, outlier dalam boxplot adalah data yang melebihi 1,5 kali kisaran 50% tengah plot.

```
>M=normal(5,1000); boxplot(quartiles(M));
```



## Fungsi Implisit

---

Plot implisit menunjukkan garis level yang menyelesaikan  $f(x,y)=\text{level}$ , di mana "level" dapat berupa nilai tunggal atau vektor nilai. Jika  $\text{level}=\text{"auto"}$ , akan ada garis level nc, yang akan menyebar antara fungsi minimum dan maksimum secara merata. Warna yang lebih gelap atau lebih terang dapat ditambahkan dengan  $\text{>hue}$  untuk menunjukkan nilai fungsi. Untuk fungsi implisit, xv harus berupa fungsi atau ekspresi dari parameter x dan y, atau, sebagai alternatif, xv dapat berupa matriks nilai.

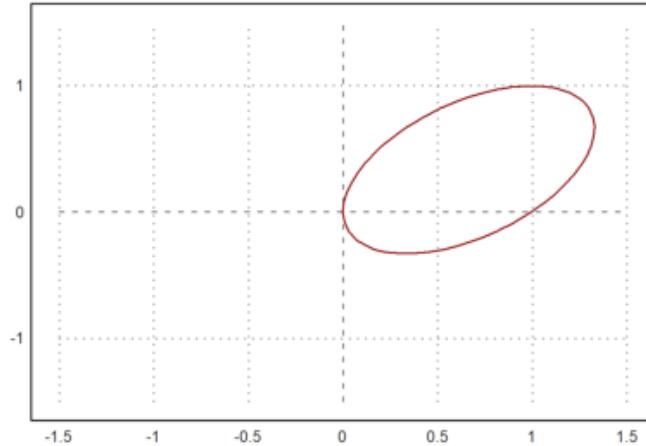
Euler dapat menandai garis level

$$f(x, y) = c$$

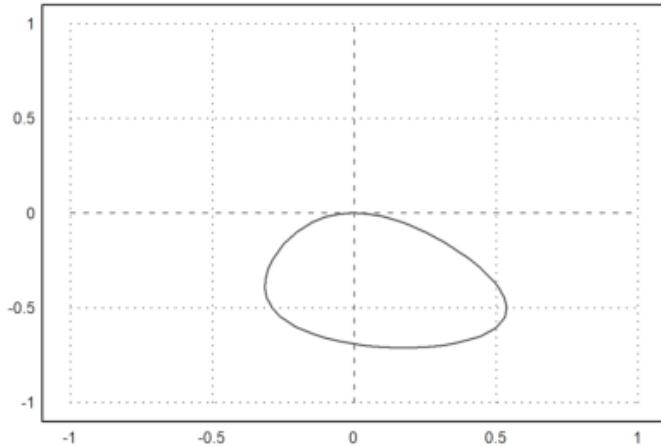
dari fungsi apapun.

Untuk menggambar himpunan  $f(x,y)=c$  untuk satu atau lebih konstanta  $c$ , Anda dapat menggunakan `plot2d()` dengan plot implisitnya di dalam bidang. Parameter untuk  $c$  adalah  $\text{level}=c$ , di mana  $c$  dapat berupa vektor garis level. Selain itu, skema warna dapat digambar di latar belakang untuk menunjukkan nilai fungsi untuk setiap titik dalam plot. Parameter "n" menentukan kehalusan plot.

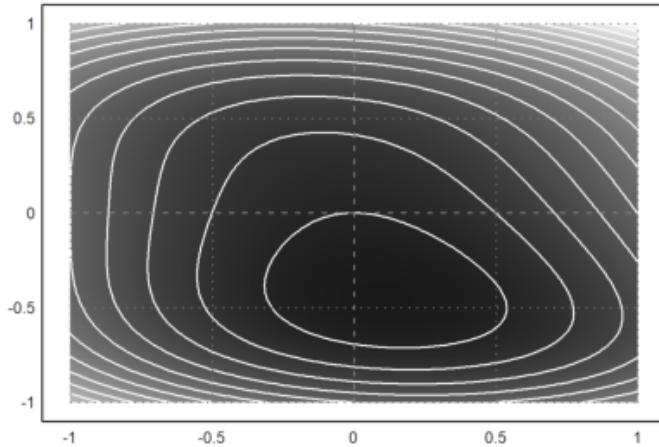
```
>aspect(1.5);
>plot2d("x^2+y^2-x*y-x",r=1.5,level=0,contourcolor=red):
```



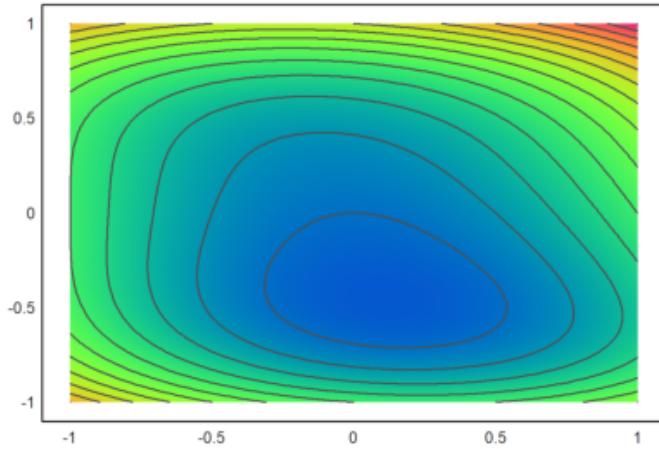
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=0); // Solutions of f(x,y)=0
```



```
>plot2d(expr,level=0:0.5:20,>hue,contourcolor=white,n=200): // nice
```

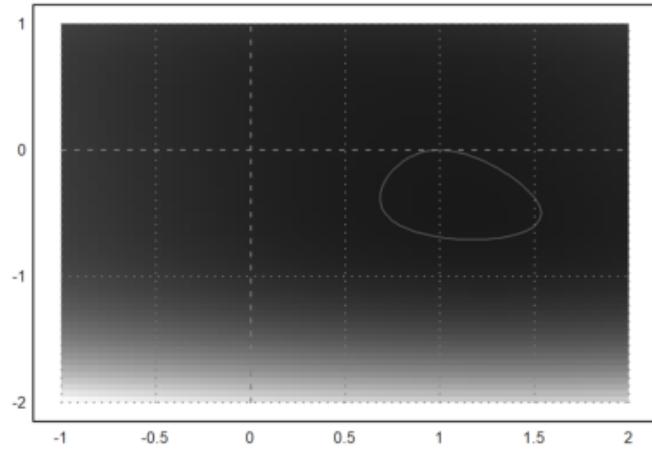


```
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=200,grid=4): // nicer
```

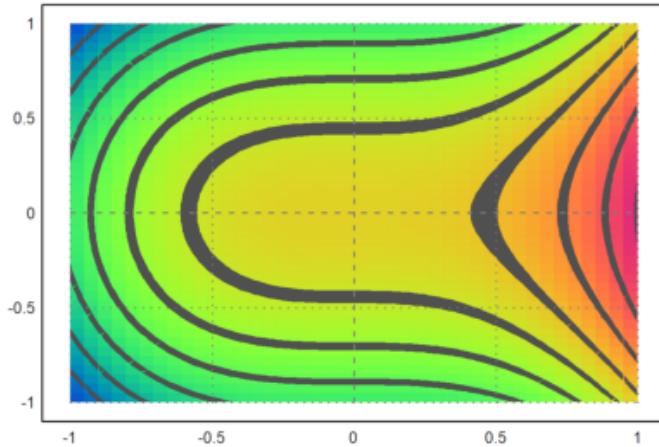


Ini berfungsi untuk plot data juga. Tetapi Anda harus menentukan rentangnya untuk label sumbu.

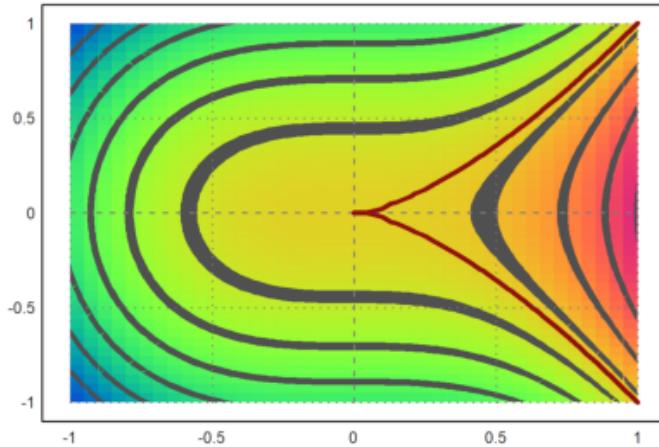
```
>x=-2:0.05:1; y=x'; z=expr(x,y);  
>plot2d(z,level=0,a=-1,b=2,c=-2,d=1,>hue):
```



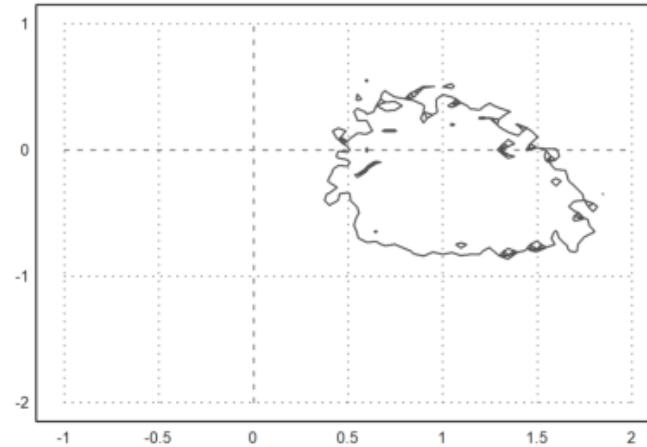
```
>plot2d("x^3-y^2",>contour,>hue,>spectral):
```



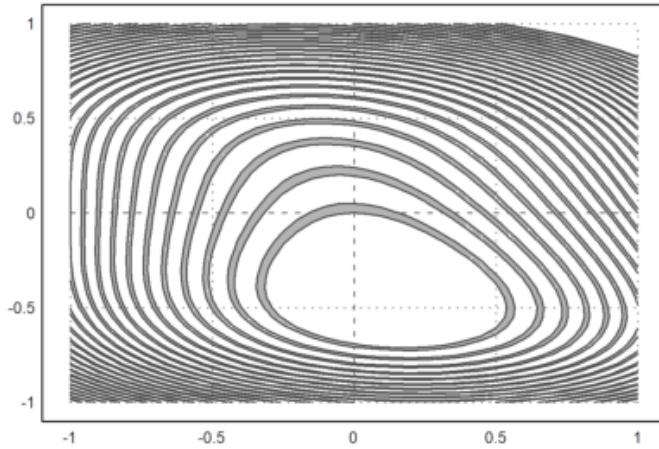
```
>plot2d("x^3-y^2",level=0,contourwidth=3,>add,contourcolor=red):
```



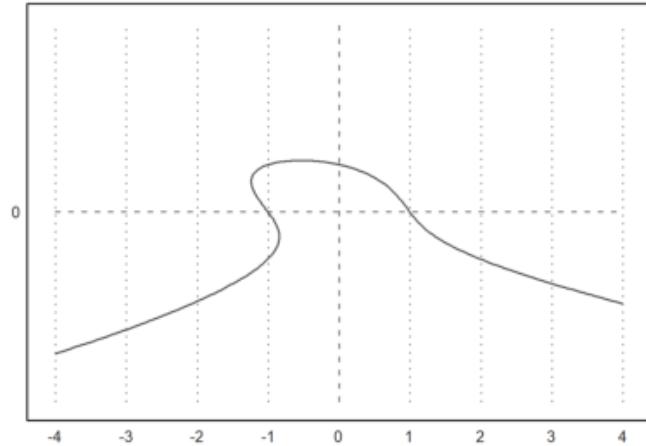
```
>z=z+normal(size(z))*0.2;  
>plot2d(z,level=0.5,a=-1,b=2,c=-2,d=1):
```



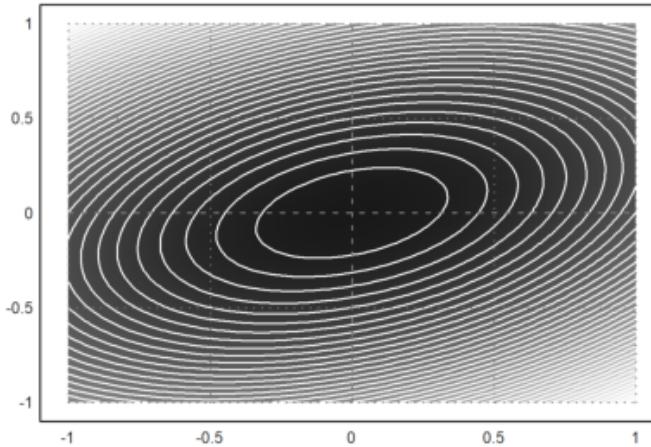
```
>plot2d(expr,level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
```



```
>plot2d("x^2+y^3+x*y",level=1,r=4,n=100):
```



```
>plot2d("x^2+2*y^2-x*y",level=0:0.1:10,n=100,contourcolor=white,>hue):
```



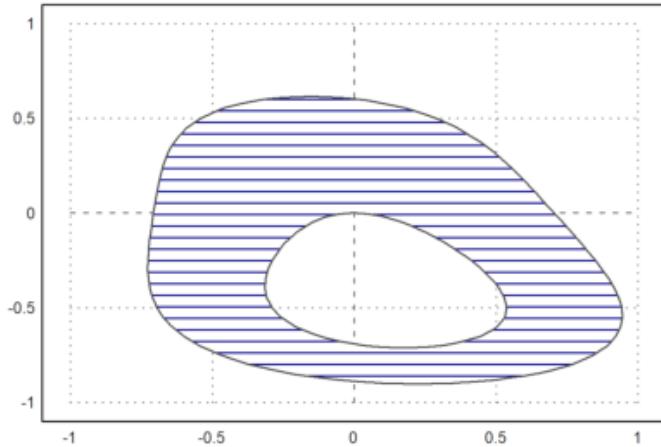
Juga dimungkinkan untuk mengisi set

$$a \leq f(x, y) \leq b$$

dengan rentang tingkat.

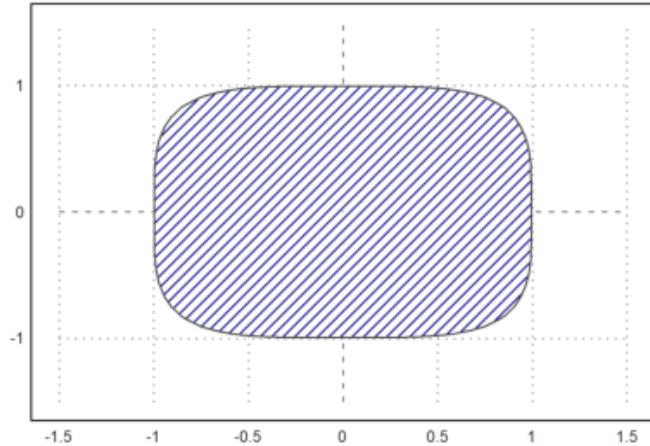
Dimungkinkan untuk mengisi wilayah nilai untuk fungsi tertentu. Untuk ini, level harus berupa matriks 2xn. Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

```
>plot2d(expr,level=[0;1],style="-",color=blue): // 0 <= f(x,y) <= 1
```

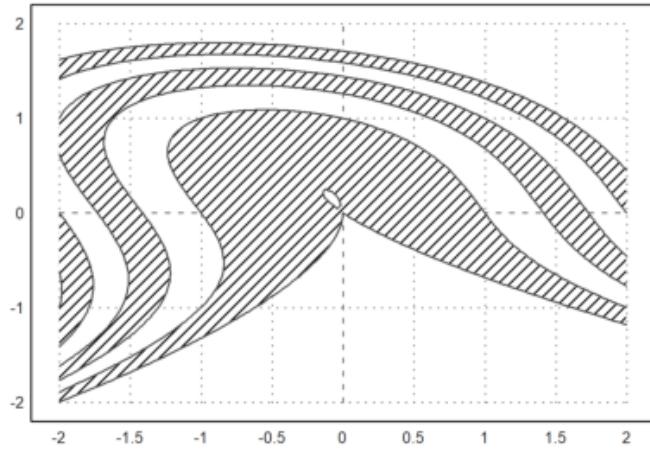


Plot implisit juga dapat menunjukkan rentang level. Kemudian level harus berupa matriks 2xn dari interval level, di mana baris pertama berisi awal dan baris kedua adalah akhir dari setiap interval. Atau, vektor baris sederhana dapat digunakan untuk level, dan parameter dl memperluas nilai level ke interval.

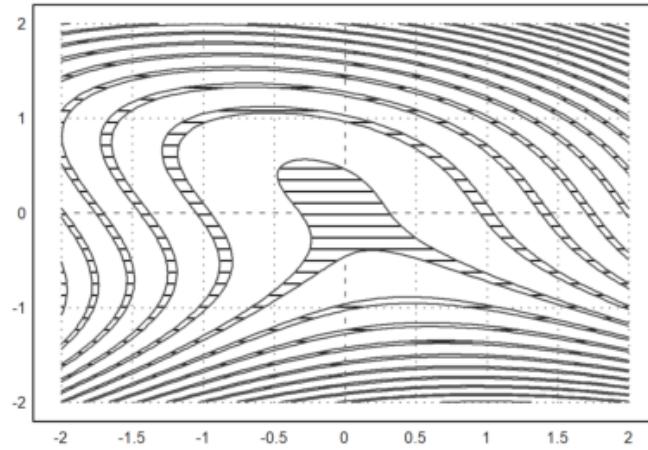
```
>plot2d("x^4+y^4",r=1.5,level=[0;1],color=blue,style="/"):
```



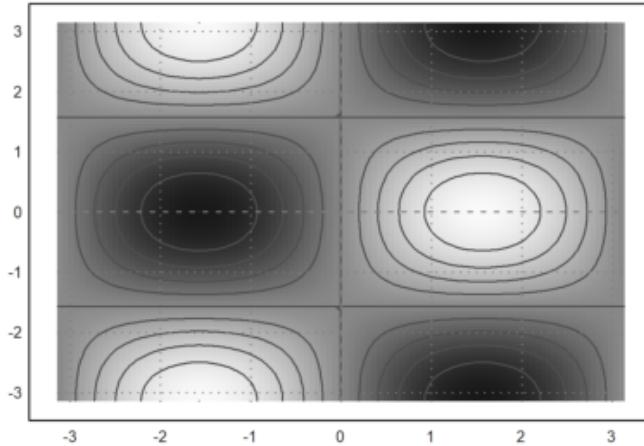
```
>plot2d("x^2+y^3+x*y",level=[0,2,4;1,3,5],style="/",r=2,n=100):
```



```
>plot2d("x^2+y^3+x*y",level=-10:20,r=2,style="-",dl=0.1,n=100):
```



```
>plot2d("sin(x)*cos(y)",r=pi,>hue,>levels,n=100):
```

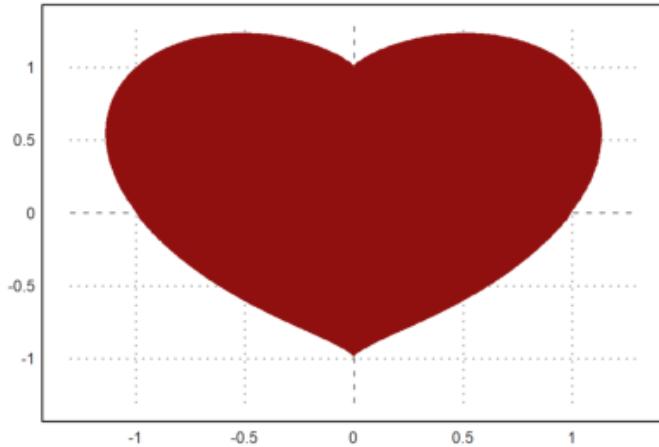


Dimungkinkan juga untuk menandai suatu wilayah

$$a \leq f(x, y) \leq b.$$

Ini dilakukan dengan menambahkan level dengan dua baris.

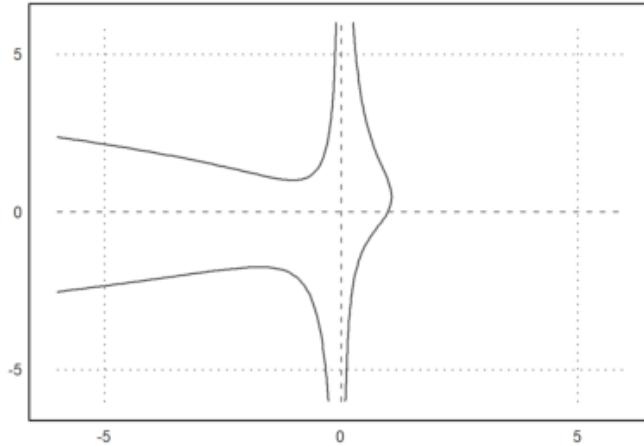
```
>plot2d("(x^2+y^2-1)^3-x^2*y^3",r=1.3, ...
> style="#",color=red,<outline, ...
> level=[-2;0],n=100):
```



Dimungkinkan untuk menentukan level tertentu. Misalnya, kita dapat memplot solusi persamaan seperti

$$x^3 - xy + x^2y^2 = 6$$

```
>plot2d("x^3-x*y+x^2*y^2",r=6,level=1,n=100):
```



```
>function starplot1 (v, style="/", color=green, lab=none) ...
```

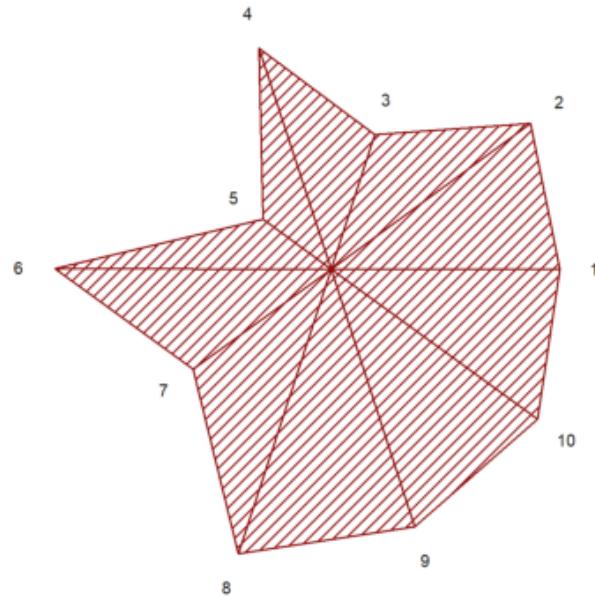
```
    if !holding() then clg; endif;
    w=window(); window(0,0,1024,1024);
    h=holding(1);
    r=max(abs(v))*1.2;
    setplot(-r,r,-r,r);
    n=cols(v); t=linspace(0,2pi,n);
    v=v|v[1]; c=v*cos(t); s=v*sin(t);
    cl=barcolor(color); st=barstyle(style);
    loop 1 to n
        polygon([0,c[#,c[#+1]], [0,s[#,s[#+1]],1];
        if lab!=none then
            rlab=v[#]+r*0.1;
            {col,row}=toscreen(cos(t[#])*rlab,sin(t[#])*rlab);
            ctext(""+lab#[#],col,row-textheight()/2);
```

```
    endif;
end;
barcolor(cl); barstyle(st);
holding(h);
window(w);
endfunction
```

Tidak ada kotak atau sumbu kutu di sini. Selain itu, kami menggunakan jendela penuh untuk plot.

Kami memanggil reset sebelum kami menguji plot ini untuk mengembalikan default grafis. Ini tidak perlu, jika Anda yakin plot Anda berhasil.

```
>reset; starplot1(normal(1,10)+5,color=red,lab=1:10):
```



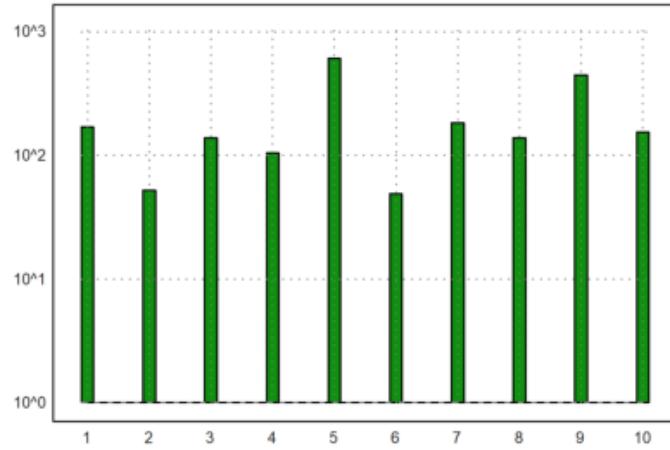
Terkadang, Anda mungkin ingin merencanakan sesuatu yang tidak dapat dilakukan plot2d, tetapi hampir. Dalam fungsi berikut, kami melakukan plot impuls logaritmik. plot2d dapat melakukan plot logaritmik, tetapi tidak untuk batang impuls.

```
>function logimpulseplot1 (x,y) ...
```

```
{x0,y0}=makeimpulse(x,log(y)/log(10));
plot2d(x0,y0,>bar,grid=0);
h=holding(1);
frame();
xgrid(ticks(x));
p=plot();
for i=-10 to 10;
    if i<=p[4] and i>=p[3] then
        ygrid(i,yt="10^"+i);
    endif;
end;
holding(h);
endfunction
```

Mari kita uji dengan nilai yang terdistribusi secara eksponensial.

```
>aspect(1.5); x=1:10; y=-log(random(size(x)))*200; ...
>logimpulseplot1(x,y):
```



Mari kita menganimasikan kurva 2D menggunakan plot langsung. Perintah `plot(x,y)` hanya memplot kurva ke jendela plot. `setplot(a,b,c,d)` mengatur jendela ini.

Fungsi `wait(0)` memaksa plot untuk muncul di jendela grafik. Jika tidak, menggambar ulang terjadi dalam interval waktu yang jarang.

```
>function animliss (n,m) ...
```

```
t=linspace(0,2pi,500);
f=0;
c=framecolor(0);
l=linewidth(2);
setplot(-1,1,-1,1);
repeat
  clg;
```

```
plot(sin(n*t),cos(m*t+f));
wait(0);
if testkey() then break; endif;
f=f+0.02;
end;
framecolor(c);
linewidth(l);
endfunction
```

Tekan sembarang tombol untuk menghentikan animasi ini.

```
>animliss(2,3); // lihat hasilnya, jika sudah puas, tekan ENTER
```

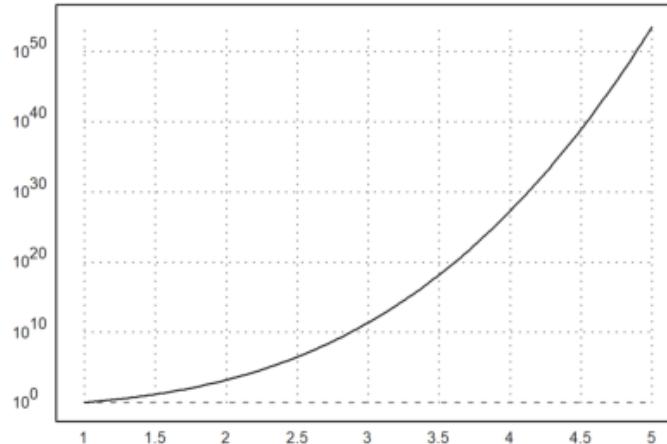
## Plot Logaritmik

EMT menggunakan parameter "logplot" untuk skala logaritmik.

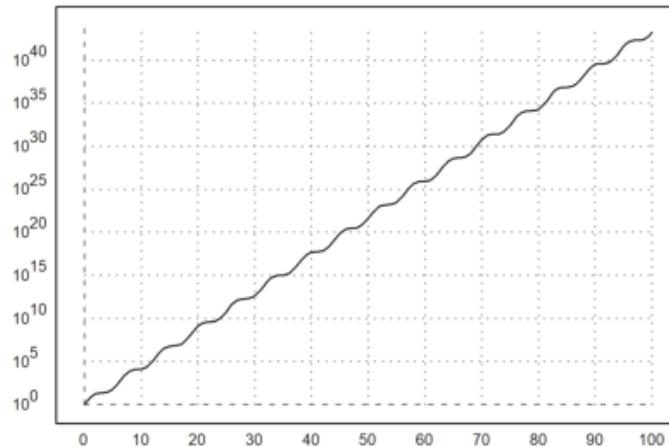
Plot logaritma dapat diplot baik menggunakan skala logaritma dalam y dengan logplot=1, atau menggunakan skala logaritma dalam x dan y dengan logplot=2, atau dalam x dengan logplot=3.

- logplot=1: y-logaritma
- logplot=2: x-y-logaritma
- logplot=3: x-logaritma

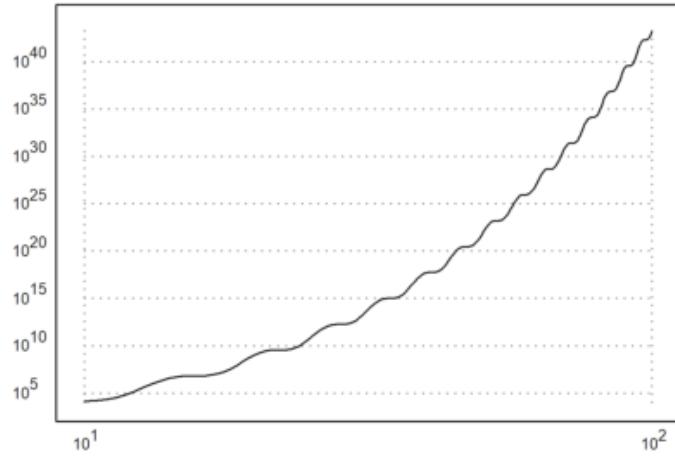
```
>plot2d("exp(x^3-x)*x^2",1,5,logplot=1):
```



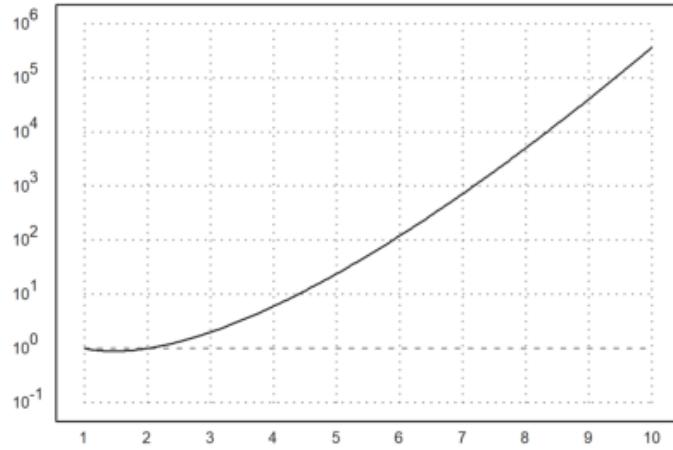
```
>plot2d("exp(x+sin(x))",0,100,logplot=1):
```



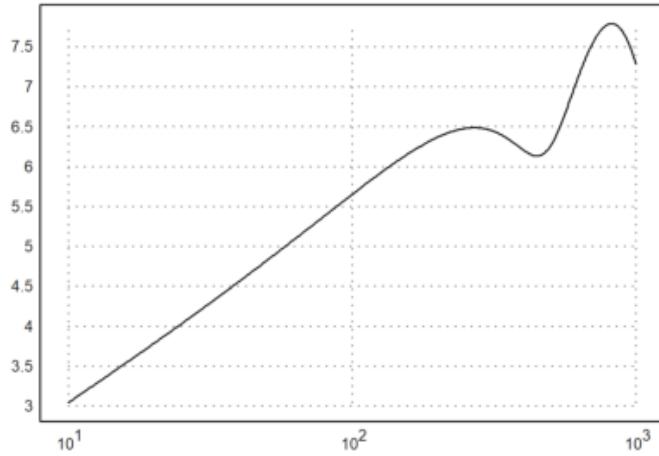
```
>plot2d("exp(x+sin(x))",10,100,logplot=2):
```



```
>plot2d("gamma(x)",1,10,logplot=1):
```

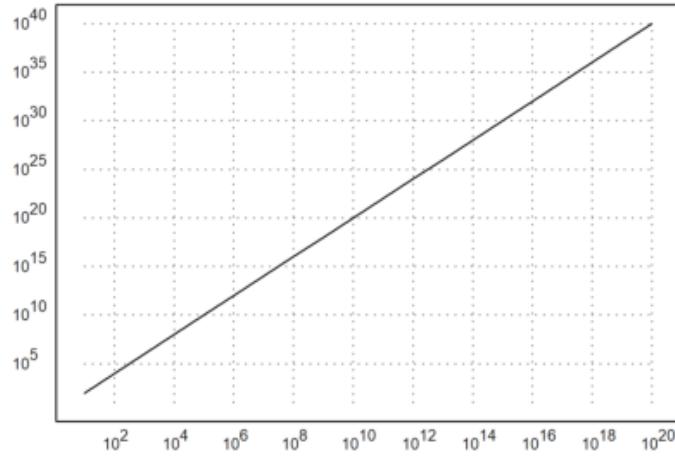


```
>plot2d("log(x*(2+sin(x/100)))",10,1000,logplot=3):
```



Ini juga berfungsi dengan plot data.

```
>x=10^(1:20); y=x^2-x;  
>plot2d(x,y,logplot=2):
```



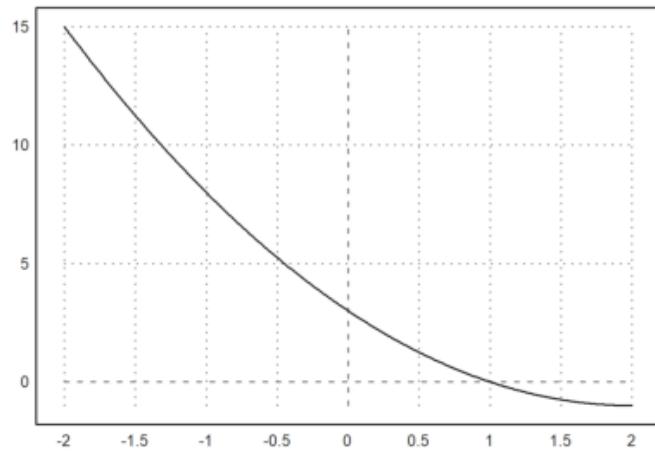
## Contoh Soal

---

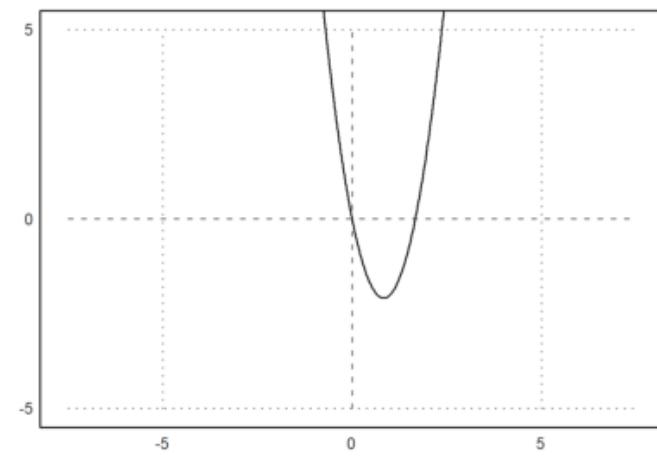
Plot grafik fungsi kuadrat berikut:

$$f(x) = x^2 - 4x + 3$$

```
>plot2d("x^2-4x+3"):
```



```
>function f(x) := 3x^2-5x;  
>plot2d("f",r=5):
```



## Rujukan Lengkap Fungsi plot2d()

---

```
function plot2d (xv, yv, btest, a, b, c, d, xmin, xmax, r, n, ..
logplot, grid, frame, framecolor, square, color, thickness, style,
..  
auto, add, user, delta, points, addpoints, pointstyle, bar,  
histogram, ..  
  
distribution, even, steps, own, adaptive, hue, level, contour, ..
nc, filled, fillcolor, outline, title, xl, yl, maps, contourcolor,  
..  
contourwidth, ticks, margin, clipping, cx, cy, insimg, spectral, ..
cgrid, vertical, smaller, dl, niveau, levels)
```

Multipurpose plot function for plots in the plane (2D plots). This function can do plots of functions of one variables, data plots, curves in the plane, bar plots, grids of complex numbers, and implicit plots of functions of two variables.

#### Parameters

x,y : equations, functions or data vectors  
a,b,c,d : Plot area (default a=-2,b=2)  
r : if r is set, then a=cx-r, b=cx+r, c=cy-r, d=cy+r

r can be a vector [rx,ry] or a vector [rx1,rx2,ry1,ry2].

xmin,xmax : range of the parameter for curves  
auto : Determine y-range automatically (default)  
square : if true, try to keep square x-y-ranges  
n : number of intervals (default is adaptive)  
grid : 0 = no grid and labels,

1 = axis only,  
2 = normal grid (see below for the number of grid lines)  
3 = inside axis  
4 = no grid  
5 = full grid including margin  
6 = ticks at the frame  
7 = axis only  
8 = axis only, sub-ticks

frame : 0 = no frame  
framecolor: color of the frame and the grid  
margin : number between 0 and 0.4 for the margin around the plot  
color : Color of curves. If this is a vector of colors,

it will be used for each row of a matrix of plots. In the

case of

point plots, it should be a column vector. If a row vector

or a

full matrix of colors is used for point plots, it will be

used for

each data point.

thickness : line thickness for curves

This value can be smaller than 1 for very thin lines.

style : Plot style for lines, markers, and fills.

```
For points use
"[]", "<>", ".", "..", "...",
"*", "+", "|", "-", "o"
"[]#", "<>#", "o#" (filled shapes)
"[]w", "<>w", "ow" (non-transparent)
For lines use
"--", "---", "-.", ".-", "-.-", "->"
For filled polygons or bar plots use
"#", "#0", "0", "/", "\", "/\",
"+", "|", "-", "t"
```

points : plot single points instead of line segments

addpoints : if true, plots line segments and points

add : add the plot to the existing plot

user : enable user interaction for functions

delta : step size for user interaction

bar : bar plot (x are the interval bounds, y the interval values)

histogram : plots the frequencies of x in n subintervals

distribution=n : plots the distribution of x with n subintervals

even : use inter values for automatic histograms.

steps : plots the function as a step function (steps=1,2)

adaptive : use adaptive plots (n is the minimal number of steps)

level : plot level lines of an implicit function of two variables

outline : draws boundary of level ranges.

If the level value is a 2xn matrix, ranges of levels will be drawn in the color using the given fill style. If outline is true, it

will be drawn in the contour color. Using this feature, regions of f(x,y) between limits can be marked.

hue : add hue color to the level plot to indicate the function

value

contour : Use level plot with automatic levels  
nc : number of automatic level lines  
title : plot title (default "")  
xl, yl : labels for the x- and y-axis  
smaller : if >0, there will be more space to the left for labels.  
vertical :

Turns vertical labels on or off. This changes the global variable  
verticallabels locally for one plot. The value 1 sets only vertical  
text, the value 2 uses vertical numerical labels on the y axis.

filled : fill the plot of a curve  
fillcolor : fill color for bar and filled curves  
outline : boundary for filled polygons  
logplot : set logarithmic plots

1 = logplot in y,  
2 = logplot in xy,  
3 = logplot in x

own :

A string, which points to an own plot routine. With >user, you get  
the same user interaction as in plot2d. The range will be set  
before each call to your function.

maps : map expressions (0 is faster), functions are always mapped.

contourcolor : color of contour lines

contourwidth : width of contour lines

clipping : toggles the clipping (default is true)

title :

This can be used to describe the plot. The title will appear above the plot. Moreover, a label for the x and y axis can be added with xl="string" or yl="string". Other labels can be added with the functions label() or labelbox(). The title can be a unicode string or an image of a Latex formula.

cgrid :

Determines the number of grid lines for plots of complex grids.

Should be a divisor of the the matrix size minus 1 (number of subintervals). cgrid can be a vector [cx,cy].

## Overview

The function can plot

- expressions, call collections or functions of one variable,
- parametric curves,
- x data against y data,
- implicit functions,
- bar plots,
- complex grids,
- polygons.

If a function or expression for xv is given, plot2d() will compute values in the given range using the function or expression. The

expression must be an expression in the variable x. The range must be defined in the parameters a and b unless the default range should be used. The y-range will be computed automatically, unless c and d are specified, or a radius r, which yields the range  $r,r$

for x and y. For plots of functions, plot2d will use an adaptive evaluation of the function by default. To speed up the plot for complicated functions, switch this off with <adaptive, and optionally decrease the number of intervals n. Moreover, plot2d() will by default use mapping. I.e., it will compute the plot element for element. If your expression or your functions can handle a vector x, you can switch that off with <maps for faster evaluation.

Note that adaptive plots are always computed element for element. If functions or expressions for both xv and for yv are specified, plot2d() will compute a curve with the xv values as x-coordinates and the yv values as y-coordinates. In this case, a range should be defined for the parameter using xmin, xmax. Expressions contained in strings must always be expressions in the parameter variable x.

## Menggambar Plot 3D dengan EMT

---

Ini adalah pengenalan plot 3D di Euler. Kita membutuhkan plot 3D untuk memvisualisasikan fungsi dari dua variabel.

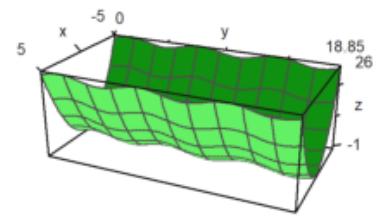
Euler menggambar fungsi tersebut menggunakan algoritma pengurutan untuk menyembunyikan bagian di latar belakang. Secara umum, Euler menggunakan proyeksi pusat. Standarnya adalah dari kuadran x-y positif menuju titik asal  $x=y=z=0$ , tetapi sudut=0° terlihat dari arah sumbu y. Sudut pandang dan tinggi dapat diubah.

Euler dapat merencanakan

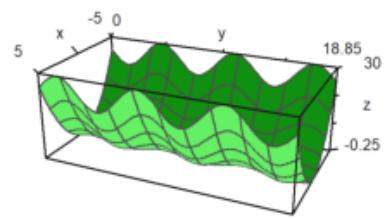
- permukaan dengan bayangan dan garis level atau rentang level,
- awan poin,
- kurva parametrik,
- permukaan implisit.

Plot 3D dari suatu fungsi menggunakan plot3d. Cara termudah adalah dengan memplot ekspresi dalam x dan y. Parameter r mengatur kisaran plot di sekitar (0,0).

```
>aspect(1.5); plot3d("x^2+sin(y)",-5,5,0,6*pi):
```



```
>plot3d("x^2+x*sin(y)", -5,5,0,6*pi):
```



Untuk grafik fungsi, gunakan

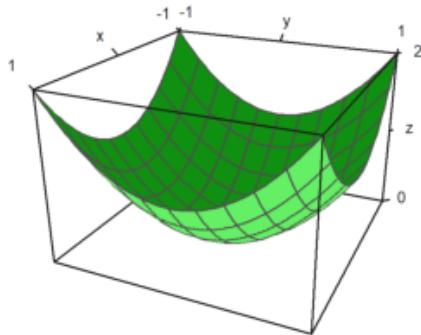
- ekspresi sederhana dalam x dan y,
- nama fungsi dari dua variabel
- atau matriks data.

Standarnya adalah kotak kawat yang diisi dengan warna berbeda di kedua sisi. Perhatikan bahwa jumlah default interval grid adalah 10, tetapi plot menggunakan jumlah default 40x40 persegi panjang untuk membangun permukaan. Ini bisa diubah.

- n=40, n=[40,40]: jumlah garis grid di setiap arah
- grid=10, grid=[10,10]: jumlah garis grid di setiap arah.

Kami menggunakan default n=40 dan grid=10.

```
>plot3d("x^2+y^2"):
```

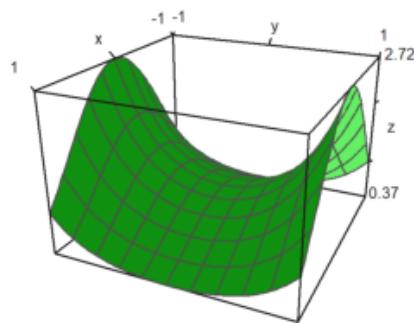


Interaksi pengguna dimungkinkan dengan >parameter pengguna. Pengguna dapat menekan tombol berikut.

- kiri, kanan, atas, bawah: putar sudut pandang
- +,-: memperbesar atau memperkecil
- a: menghasilkan anaglyph (lihat di bawah)
- l: beralih memutar sumber cahaya (lihat di bawah)
- spasi: reset ke default
- kembali: akhiri interaksi

```
>plot3d("exp(-x^2+y^2)",>user, ...
> title="Turn with the vector keys (press return to finish)":
```

Turn with the vector keys (press return to finish)



Rentang plot untuk fungsi dapat ditentukan dengan

- a,b: rentang-x
- c,d: rentang-y
- r: persegi simetris di sekitar (0,0).
- n: jumlah subinterval untuk plot.

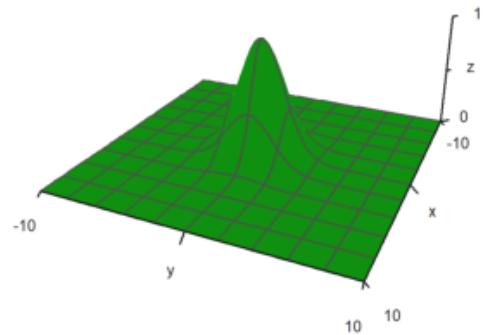
Ada beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.

fscale: skala ke nilai fungsi (defaultnya adalah <fscale>).

skala: angka atau vektor 1x2 untuk skala ke arah x dan y.

bingkai: jenis bingkai (default 1).

```
>plot3d("exp(-(x^2+y^2)/5)",r=10,n=80,fscale=4,scale=1.2,frame=3):
```



Tampilan dapat diubah dengan berbagai cara.

- jarak: jarak pandang ke plot.
- zoom: nilai zoom.
- sudut: sudut terhadap sumbu y negatif dalam radian.
- tinggi: ketinggian tampilan dalam radian.

Nilai default dapat diperiksa atau diubah dengan fungsi `view()`. Ini mengembalikan parameter dalam urutan di atas.

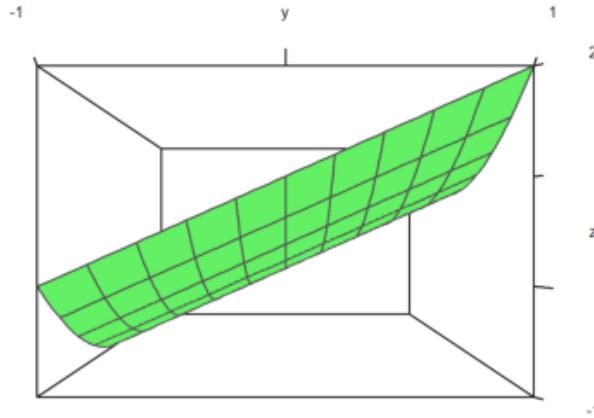
```
>view
```

```
[5, 2.6, 2, 0.4]
```

Jarak yang lebih dekat membutuhkan lebih sedikit zoom. Efeknya lebih seperti lensa sudut lebar.

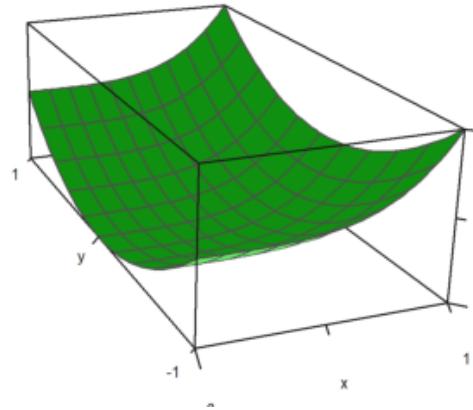
Dalam contoh berikut, sudut=0 dan tinggi=0 terlihat dari sumbu y negatif. Label sumbu untuk y disembunyikan dalam kasus ini.

```
>plot3d("x^2+y",distance=3,zoom=2,angle=pi/2,height=0):
```



Plot terlihat selalu ke pusat kubus plot. Anda dapat memindahkan pusat dengan parameter tengah.

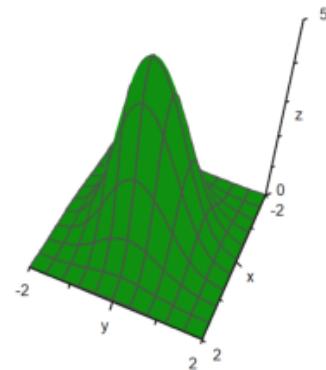
```
>plot3d("x^4+y^2",a=0,b=1,c=-1,d=1,angle=-20°,height=20°, ...
> center=[0.4,0,0],zoom=5):
```



Plot diskalakan agar sesuai dengan kubus satuan untuk dilihat. Jadi tidak perlu mengubah jarak atau zoom tergantung pada ukuran plot. Namun, label mengacu pada ukuran sebenarnya.

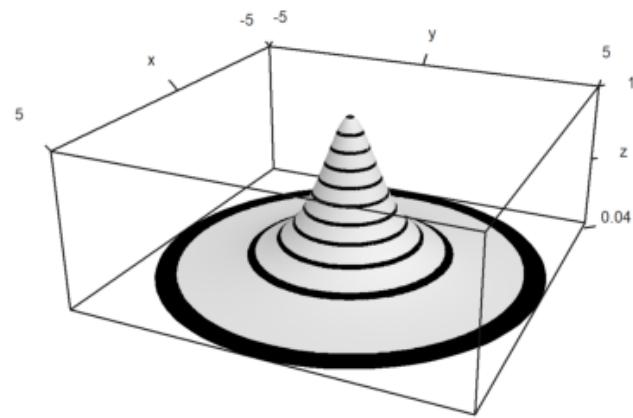
Jika Anda mematikannya dengan `scale=false`, Anda perlu berhati-hati, bahwa plot masih cocok dengan jendela plot, dengan mengubah jarak pandang atau zoom, dan memindahkan pusat.

```
>plot3d("5*exp(-x^2-y^2)",r=2,<fscale,<scale,distance=13,height=50°, ...
> center=[0,0,-2],frame=3):
```

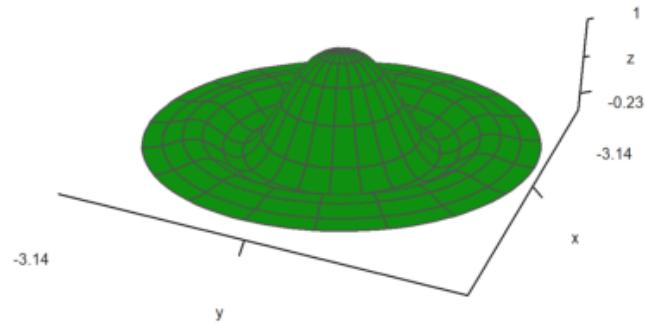


Sebuah plot kutub juga tersedia. Parameter `polar=true` menggambar plot polar. Fungsi tersebut harus tetap merupakan fungsi dari  $x$  dan  $y$ . Parameter `"fscale"` menskalakan fungsi dengan skala sendiri. Jika tidak, fungsi diskalakan agar sesuai dengan kubus.

```
>plot3d("1/(x^2+y^2+1)",r=5,>polar, ...
>fscale=2,>hue,n=100,zoom=4,>contour,color=gray):
```



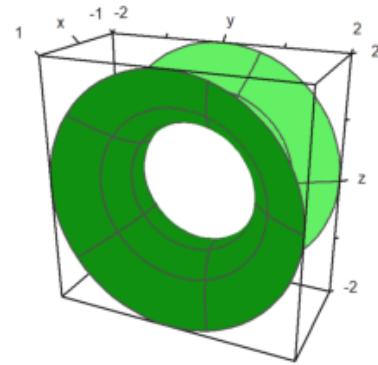
```
>function f(r) := exp(-r/2)*cos(r); ...
>plot3d("f(x^2+y^2)",>polar,scale=[1,1,0.4],r=pi,frame=3,zoom=4):
```



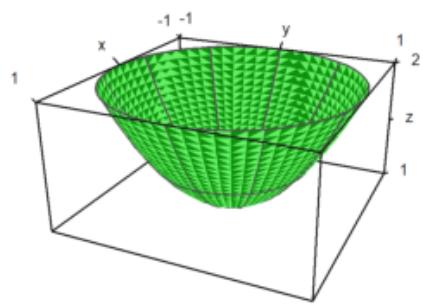
Rotasi parameter memutar fungsi dalam x di sekitar sumbu x.

- rotate=1: Menggunakan sumbu x
- rotate=2: Menggunakan sumbu z

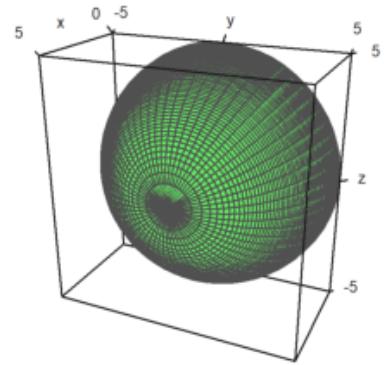
```
>plot3d("x^2+1",a=-1,b=1,rotate=true,grid=5):
```



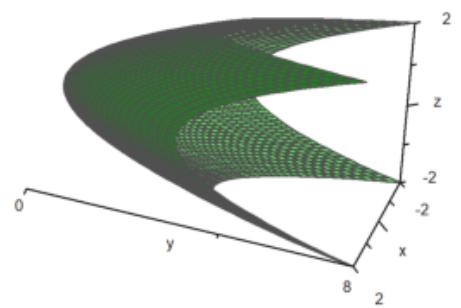
```
>plot3d("x^2+1",a=-1,b=1,rotate=2,grid=5):
```



```
>plot3d("sqrt(25-x^2)",a=0,b=5,rotate=1):
```



```
>plot3d("x", "x^2+y^2", "y", r=2, zoom=3.5, frame=3):
```



## Plot Kontur

---

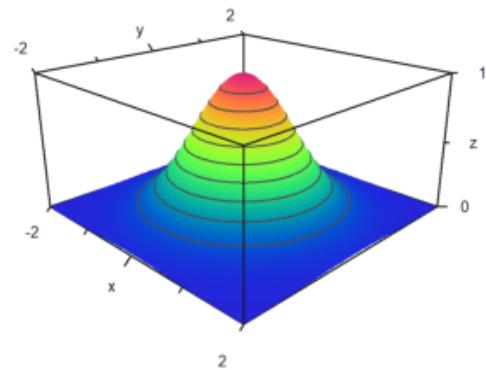
Untuk plot, Euler menambahkan garis grid. Sebagai gantinya dimungkinkan untuk menggunakan garis level dan rona satu warna atau rona berwarna spektral. Euler dapat menggambar tinggi fungsi pada plot dengan bayangan. Di semua plot 3D, Euler dapat menghasilkan anaglyph merah/sian.

- > hue: Menyalakan bayangan cahaya alih-alih kabel.
- > kontur: Memplot garis kontur otomatis pada plot.
- level=... (atau level): Sebuah vektor nilai untuk garis kontur.

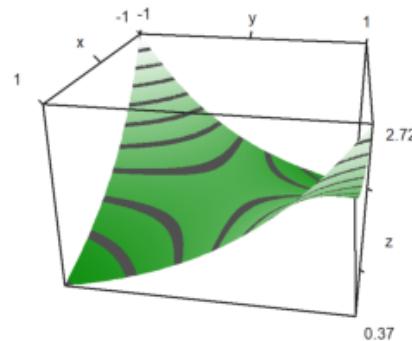
Standarnya adalah level="auto", yang menghitung beberapa garis level secara otomatis. Seperti yang Anda lihat di plot, level sebenarnya adalah rentang level.

Gaya default dapat diubah. Untuk plot kontur berikut, kami menggunakan grid yang lebih halus untuk 100x100 poin, skala fungsi dan plot, dan menggunakan sudut pandang yang berbeda.

```
>plot3d("exp(-x^2-y^2)",r=2,n=100,level="thin", ...
>>contour,>spectral,fscale=1,scale=1.1,angle=45°,height=20°):
```



```
>plot3d("exp(x*y)",angle=100°,>contour,color=green):
```



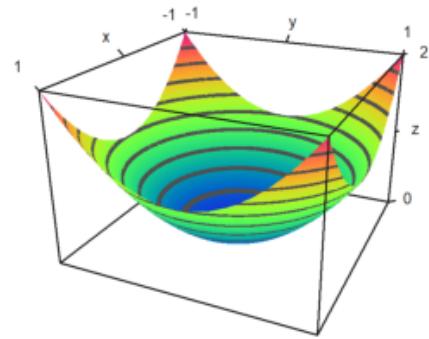
Bayangan default menggunakan warna abu-abu. Tetapi rentang warna spektral juga tersedia.

-> spektral: Menggunakan skema spektral default

- color=...: Menggunakan warna khusus atau skema spektral

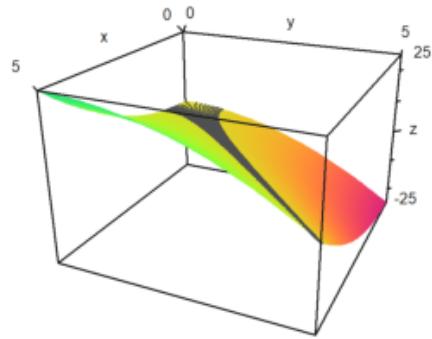
Untuk plot berikut, kami menggunakan skema spektral default dan menambah jumlah titik untuk mendapatkan tampilan yang sangat halus.

```
>plot3d("x^2+y^2",>spectral,>contour,n=100):
```



Alih-alih garis level otomatis, kita juga dapat mengatur nilai garis level. Ini akan menghasilkan garis level tipis alih-alih rentang level.

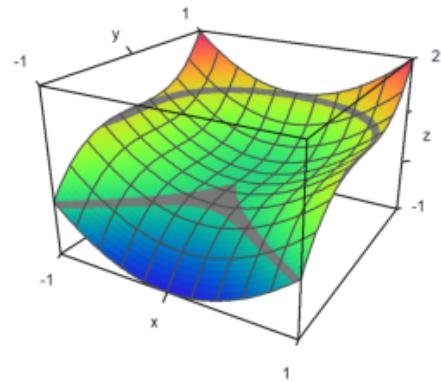
```
>plot3d("x^2-y^2",0,5,0,5,level=-1:0.1:1,color=redgreen):
```



Dalam plot berikut, kami menggunakan dua pita level yang sangat luas dari -0,1 hingga 1, dan dari 0,9 hingga 1. Ini dimasukkan sebagai matriks dengan batas level sebagai kolom.

Selain itu, kami melapisi kisi dengan 10 interval di setiap arah.

```
>plot3d("x^2+y^3",level=[-0.1,0.9;0,1], ...
> >spectral,angle=30°,grid=10,contourcolor=gray):
```

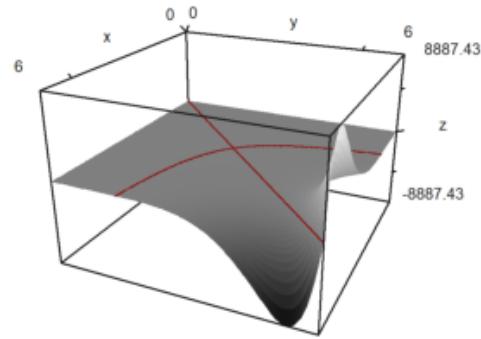


Dalam contoh berikut, kami memplot himpunan, di mana

$$f(x, y) = x^y - y^x = 0$$

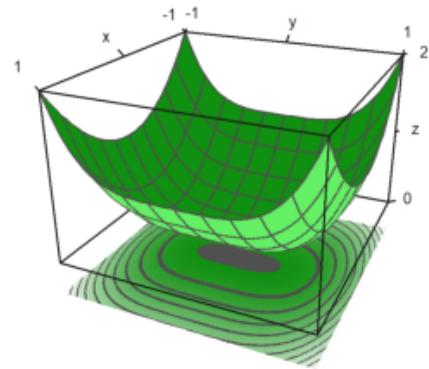
Kami menggunakan satu garis tipis untuk garis level.

```
>plot3d("x^y-y^x",level=0,a=0,b=6,c=0,d=6,contourcolor=red,n=100):
```



Dimungkinkan untuk menunjukkan bidang kontur di bawah plot. Warna dan jarak ke plot dapat ditentukan.

```
>plot3d("x^2+y^4",>cp,cpcolor=green,cpdelta=0.2):
```

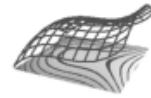
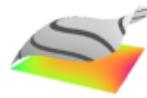
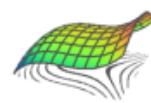
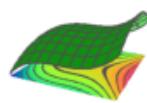


Here are a few more styles. We always turn off the frame, and use various color schemes for the plot and the grid.

```
>figure(2,2); ...
```

Dimungkinkan untuk menunjukkan bidang kontur di bawah plot. Warna dan jarak ke plot dapat ditentukan.

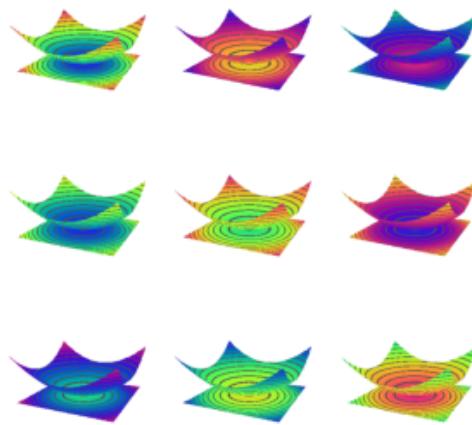
```
>expr="y^3-x^2"; ...
>figure(1); ...
> plot3d(expr,<frame,>cp,cpcolor=spectral); ...
>figure(2); ...
> plot3d(expr,<frame,>spectral,grid=10,cp=2); ...
>figure(3); ...
> plot3d(expr,<frame,>contour,color=gray,nc=5,cp=3,cpcolor=greenred); ...
>figure(4); ...
> plot3d(expr,<frame,>hue,grid=10,>transparent,>cp,cpcolor=gray); ...
>figure(0):
```



Ada beberapa skema spektral lainnya, bermulai dari 1 hingga 9. Tetapi Anda juga dapat menggunakan warna=nilai, di mana nilai

- spektral: untuk rentang dari biru ke merah
- putih: untuk rentang yang lebih redup
- kuningbiru, ungu, hijau, birukuning, hijaumerah
- birukuning, hijau ungu, kuning biru, merah hijau

```
>figure(3,3); ...
>for i=1:9; ...
> figure(i); plot3d("x^2+y^2",spectral=i,>contour,>cp,<frame,zoom=4); ...
>end; ...
>figure(0):
```



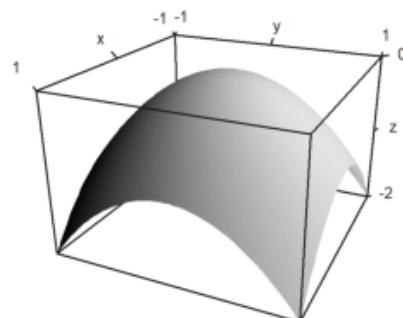
Sumber cahaya dapat diubah dengan 1 dan tombol kursor selama interaksi pengguna. Itu juga dapat diatur dengan parameter.

- cahaya: arah untuk cahaya
- amb: cahaya sekitar antara 0 dan 1

Perhatikan bahwa program tidak membuat perbedaan antara sisi plot. Tidak ada bayangan. Untuk ini, Anda perlu Povray.

```
>plot3d("-x^2-y^2", ...
>  hue=true,light=[0,1,1],amb=0,user=true, ...
>  title="Press l and cursor keys (return to exit)":
```

Press l and cursor keys (return to exit)



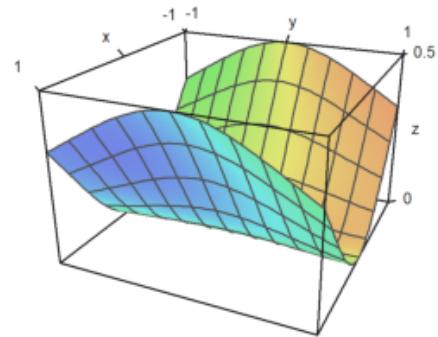
Parameter warna mengubah warna permukaan. Warna garis level juga dapat diubah.

```
>plot3d("-x^2-y^2",color=rgb(0.2,0.2,0),hue=true,frame=false, ...
> zoom=3,contourcolor=red,level=-2:0.1:1,dL=0.01):
```



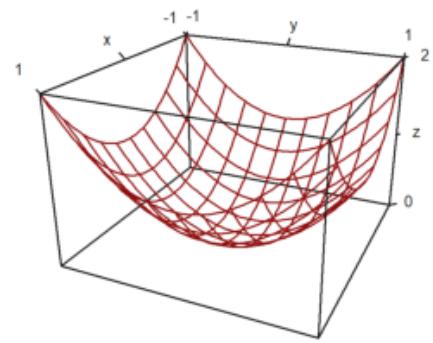
Warna 0 memberikan efek pelangi khusus.

```
>plot3d("x^2/(x^2+y^2+1)",color=0,hue=true,grid=10):
```



Permukaannya juga bisa transparan.

```
>plot3d("x^2+y^2",>transparent,grid=10,wirecolor=red):
```



## Plot Implisit

---

Ada juga plot implisit dalam tiga dimensi. Euler menghasilkan pemotongan melalui objek. Fitur plot3d termasuk plot implisit. Plot-plot ini menunjukkan himpunan nol dari suatu fungsi dalam tiga variabel. Solusi dari

$$f(x, y, z) = 0$$

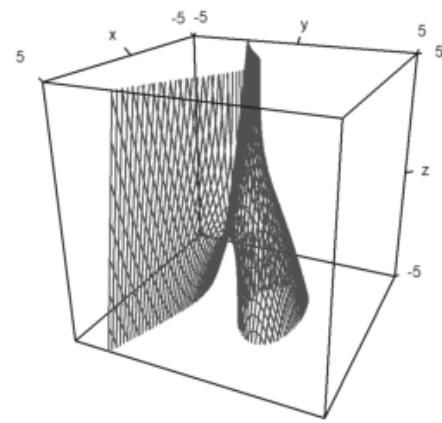
dapat divisualisasikan dalam potongan sejajar dengan bidang x-y-, x-z- dan y-z.

- implicit=1: potong sejajar dengan bidang y-z
- implicit=2: potong sejajar dengan bidang x-z
- implicit=4: potong sejajar dengan bidang x-y

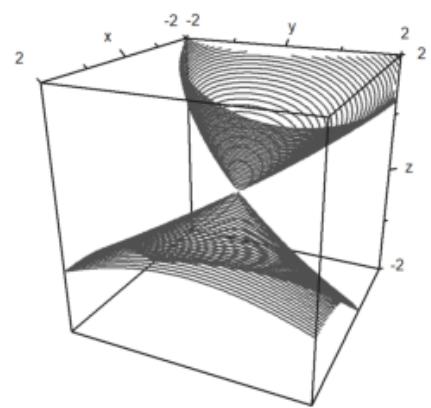
Tambahkan nilai-nilai ini, jika Anda suka. Dalam contoh kita plot

$$M = \{(x, y, z) : x^2 + y^3 + zy = 1\}$$

```
>plot3d("x^2+y^3+z*y-1",r=5,implicit=3):
```



```
>plot3d("x^2+y^2+4*x*z+z^3",>implicit,r=2,zoom=2.5):
```



## Merencanakan Data 3D

---

Sama seperti plot2d, plot3d menerima data. Untuk objek 3D, Anda perlu menyediakan matriks nilai x-, y- dan z, atau tiga fungsi atau ekspresi  $fx(x,y)$ ,  $fy(x,y)$ ,  $fz(x,y)$ .

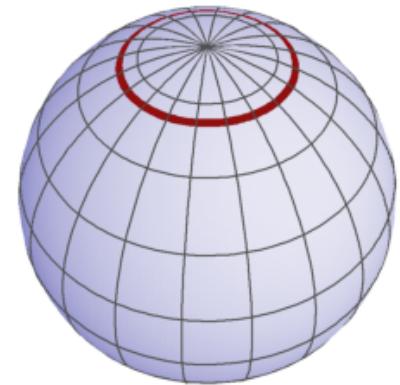
$$\gamma(t,s) = (x(t,s), y(t,s), z(t,s))$$

Karena x,y,z adalah matriks, kita asumsikan bahwa  $(t,s)$  melalui sebuah kotak persegi. Hasilnya, Anda dapat memplot gambar persegi panjang di ruang angkasa.

Anda dapat menggunakan bahasa matriks Euler untuk menghasilkan koordinat secara efektif.

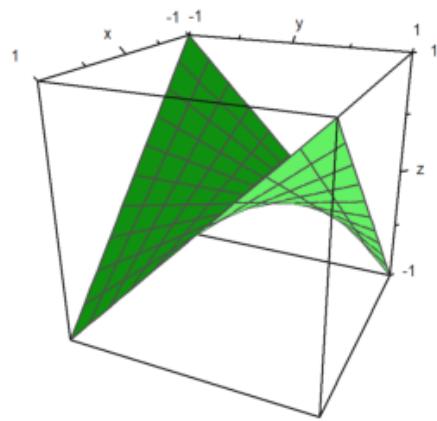
Dalam contoh berikut, kami menggunakan vektor nilai  $t$  dan vektor kolom nilai  $s$  untuk membuat parameter permukaan bola. Dalam gambar kita dapat menandai daerah, dalam kasus kita daerah kutub.

```
>t=linspace(0,2pi,180); s=linspace(-pi/2,pi/2,90)'; ...
>x=cos(s)*cos(t); y=cos(s)*sin(t); z=sin(s); ...
>plot3d(x,y,z,>hue, ...
>color=blue,<frame,grid=[10,20], ...
>values=s,contourcolor=red,level=[90°-24°;90°-22°], ...
>scale=1.4,height=50°):
```



Berikut adalah contoh, yang merupakan grafik fungsi.

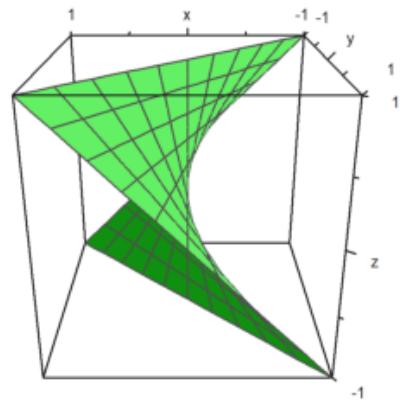
```
>t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,t*s,grid=10):
```



Namun, kita bisa membuat segala macam permukaan. Berikut adalah permukaan yang sama dengan fungsi

$$x = y z$$

```
>plot3d(t*s,t,s,angle=180°,grid=10):
```



Dengan lebih banyak usaha, kami dapat menghasilkan banyak permukaan.

Dalam contoh berikut, kita membuat tampilan bayangan dari bola yang terdistorsi. Koordinat biasa untuk bola adalah

$$\gamma(t, s) = (\cos(t) \cos(s), \sin(t) \sin(s), \cos(s))$$

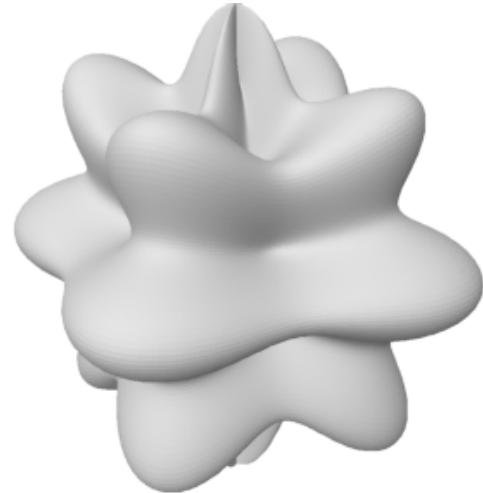
dengan

$$0 \leq t \leq 2\pi, \quad -\frac{\pi}{2} \leq s \leq \frac{\pi}{2}.$$

Kami mendistorsi ini dengan sebuah faktor

$$d(t, s) = \frac{\cos(4t) + \cos(8s)}{4}.$$

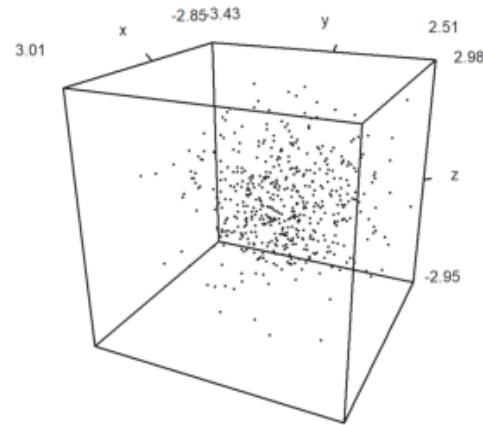
```
>t=linspace(0,2pi,320); s=linspace(-pi/2,pi/2,160)'; ...
>d=1+0.2*(cos(4*t)+cos(8*s)); ...
>plot3d(cos(t)*cos(s)*d,sin(t)*cos(s)*d,sin(s)*d,hue=1, ...
> light=[1,0,1],frame=0,zoom=5):
```



Tentu saja, titik cloud juga dimungkinkan. Untuk memplot data titik dalam ruang, kita membutuhkan tiga vektor untuk koordinat titik-titik tersebut.

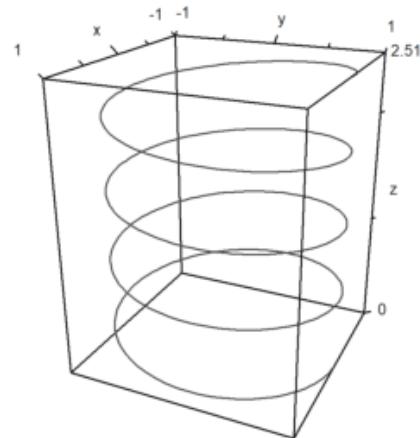
Gayanya sama seperti di plot2d dengan points=true;

```
>n=500; ...
> plot3d(normal(1,n),normal(1,n),normal(1,n),points=true,style="."):
```

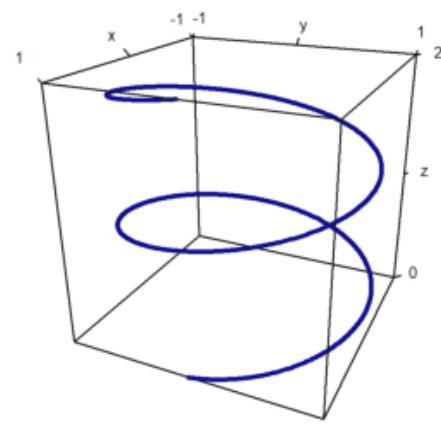


Dimungkinkan juga untuk memplot kurva dalam 3D. Dalam hal ini, lebih mudah untuk menghitung titik-titik kurva. Untuk kurva di pesawat kami menggunakan urutan koordinat dan parameter wire=true.

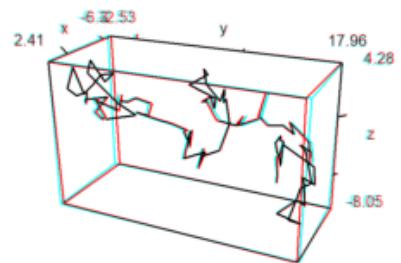
```
>t=linspace(0,8pi,500); ...
>plot3d(sin(t),cos(t),t/10,>wire,zoom=3):
```



```
>t=linspace(0,4pi,1000); plot3d(cos(t),sin(t),t/2pi,>wire, ...
>lineWidth=3,wirecolor=blue):
```

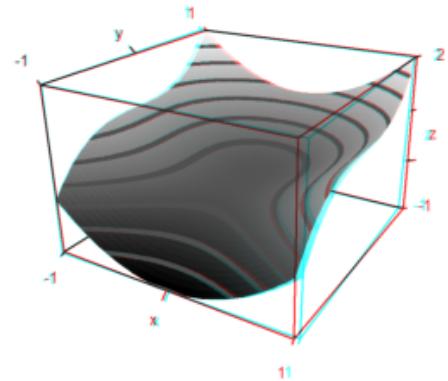


```
>X=cumsum(normal(3,100)); ...
> plot3d(X[1],X[2],X[3],>anaglyph,>wire):
```



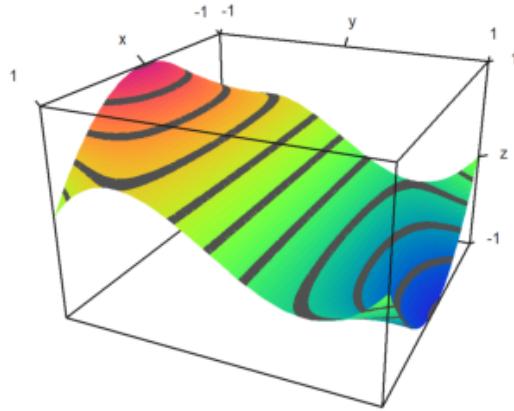
EMT juga dapat memplot dalam mode anaglyph. Untuk melihat plot seperti itu, Anda memerlukan kacamata merah/sian.

```
> plot3d("x^2+y^3",>anaglyph,>contour,angle=30°):
```



Seringkali, skema warna spektral digunakan untuk plot. Ini menekankan ketinggian fungsi.

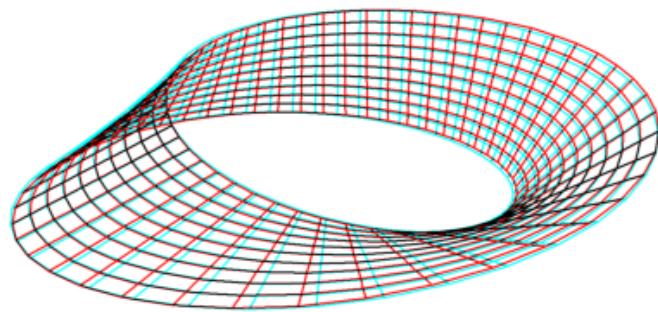
```
>plot3d("x^2*y^3-y",>spectral,>contour,zoom=3.2):
```



Euler juga dapat memplot permukaan berparameter, ketika parameternya adalah nilai x-, y-, dan z dari gambar kotak persegi panjang dalam ruang.

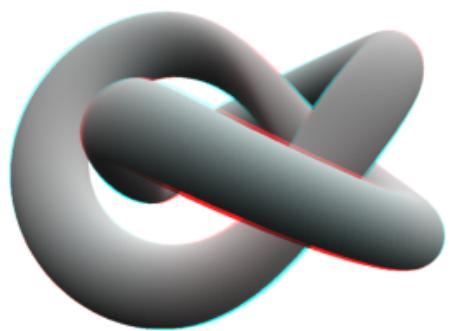
Untuk demo berikut, kami mengatur parameter u- dan v-, dan menghasilkan koordinat ruang dari ini.

```
>u=linspace(-1,1,10); v=linspace(0,2*pi,50)'; ...
>X=(3+u*cos(v/2))*cos(v); Y=(3+u*cos(v/2))*sin(v); Z=u*sin(v/2); ...
>plot3d(X,Y,Z,>anaglyph,<frame,>wire,scale=2.3):
```



Berikut adalah contoh yang lebih rumit, yang megah dengan kacamata merah/sian.

```
>u:=linspace(-pi,pi,160); v:=linspace(-pi,pi,400)'; ...
>x:=(4*(1+.25*sin(3*v))+cos(u))*cos(2*v); ...
>y:=(4*(1+.25*sin(3*v))+cos(u))*sin(2*v); ...
>z=sin(u)+2*cos(3*v); ...
>plot3d(x,y,z,frame=0,scale=1.5,hue=1,light=[1,0,-1],zoom=2.8,>anaglyph):
```



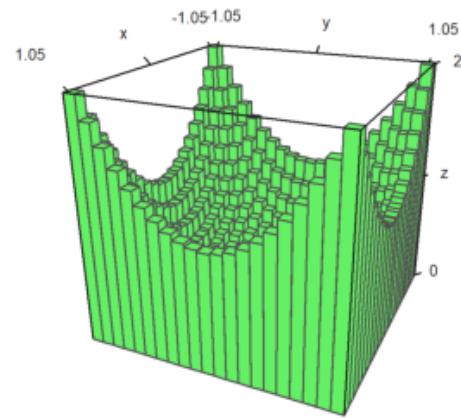
Plot bar juga dimungkinkan. Untuk ini, kita harus menyediakan

- x: vektor baris dengan n+1 elemen
- y: vektor kolom dengan n+1 elemen
- z: matriks nilai nxn.

z bisa lebih besar, tetapi hanya nilai nxn yang akan digunakan.

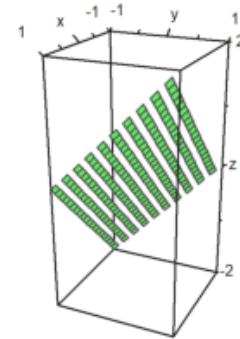
Dalam contoh, pertama-tama kita menghitung nilainya. Kemudian kita sesuaikan x dan y, sehingga vektor berpusat pada nilai yang digunakan.

```
>x=-1:0.1:1; y=x'; z=x^2+y^2; ...
>xa=(x|1.1)-0.05; ya=(y_1.1)-0.05; ...
>plot3d(xa,ya,z,bar=true);
```



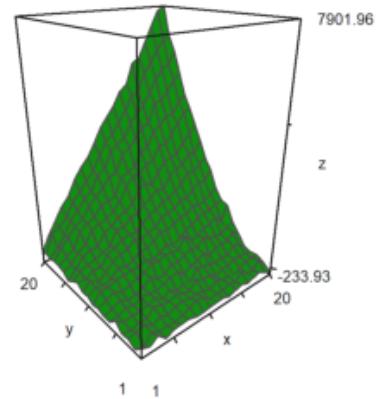
Dimungkinkan untuk membagi plot permukaan menjadi dua atau lebih bagian.

```
>x=-1:0.1:1; y=x'; z=x+y; d=zeros(size(x)); ...
>plot3d(x,y,z,disconnect=2:2:20);
```

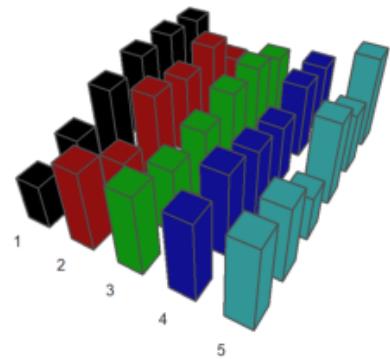


Jika memuat atau menghasilkan matriks data M dari file dan perlu memplotnya dalam 3D, Anda dapat menskalakan matriks ke [-1,1] dengan scale(M), atau menskalakan matriks dengan >zscale. Ini dapat dikombinasikan dengan faktor penskalaan individu yang diterapkan sebagai tambahan.

```
>i=1:20; j=i'; ...
>plot3d(i*j^2+100*normal(20,20),>zscale,scale=[1,1,1.5],angle=-40°,zoom=1.8):
```

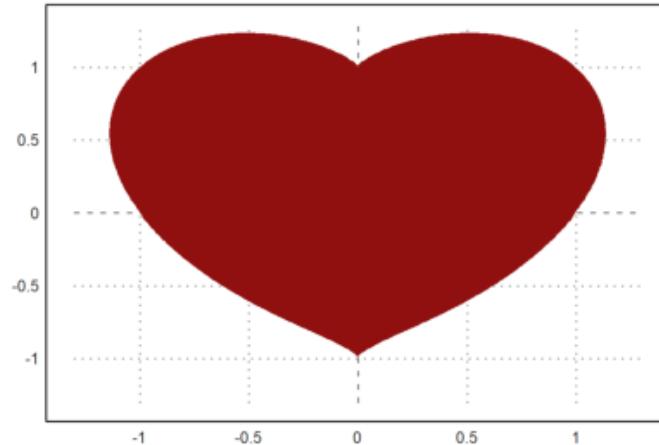


```
>Z=intrandom(5,100,6); v=zeros(5,6); ...
>loop 1 to 5; v[#]=getmultiplicities(1:6,Z[#]); end; ...
>columnsplot3d(v',scols=1:5,ccols=[1:5]):
```



## Permukaan Benda Putar

```
>plot2d("(x^2+y^2-1)^3-x^2*y^3",r=1.3, ...
>style="#",color=red,<outline, ...
>level=[-2;0],n=100):
```



```
>ekspresi &= (x^2+y^2-1)^3-x^2*y^3; $ekspresi
```

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

Kami ingin memutar kurva jantung di sekitar sumbu y. Berikut adalah ungkapan, yang mendefinisikan hati:

$$f(x, y) = (x^2 + y^2 - 1)^3 - x^2 \cdot y^3.$$

Selanjutnya kita atur

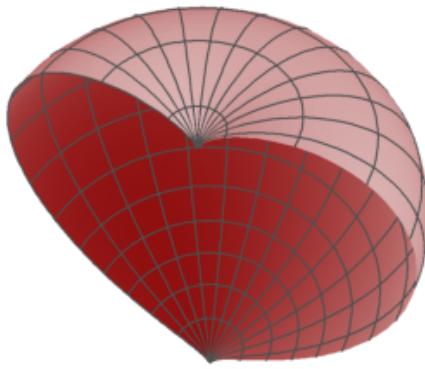
$$x = r \cos(a), \quad y = r \sin(a).$$

```
>function fr(r,a) &= ekspresi with [x=r*cos(a),y=r*sin(a)] | trigreduce; $fr(r,a)
```

$$(r^2 - 1)^3 + \frac{(\sin(5a) - \sin(3a) - 2 \sin a) r^5}{16}$$

Hal ini memungkinkan untuk mendefinisikan fungsi numerik, yang memecahkan r, jika a diberikan. Dengan fungsi itu kita dapat memplot jantung yang diputar sebagai permukaan parametrik.

```
>function map f(a) := bisect("fr",0,2;a); ...
>t=linspace(-pi/2,pi/2,100); r=f(t); ...
>s=linspace(pi,2pi,100)'; ...
>plot3d(r*cos(t)*sin(s),r*cos(t)*cos(s),r*sin(t), ...
>>hue,<frame,color=red,zoom=4,amb=0,max=0.7,grid=12,height=50°):
```

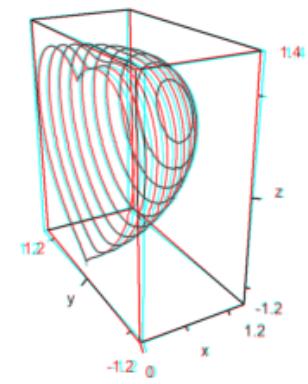


Berikut ini adalah plot 3D dari gambar di atas yang diputar di sekitar sumbu z. Kami mendefinisikan fungsi, yang menggambarkan objek.

```
>function f(x,y,z) ...
```

```
r=x^2+y^2;  
return (r+z^2-1)^3-r*z^3;  
endfunction
```

```
>plot3d("f(x,y,z)", ...  
>xmin=0,xmax=1.2,ymin=-1.2,ymax=1.2,zmin=-1.2,zmax=1.4, ...  
>implicit=1,angle=-30°,zoom=2.5,n=[10,60,60],>anaglyph):
```



## Plot 3D Khusus

---

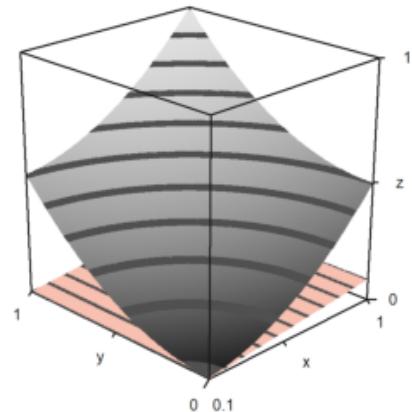
Fungsi plot3d bagus untuk dimiliki, tetapi tidak memenuhi semua kebutuhan. Selain rutinitas yang lebih mendasar, dimungkinkan untuk mendapatkan plot berbingkai dari objek apa pun yang Anda suka.

Meskipun Euler bukan program 3D, ia dapat menggabungkan beberapa objek dasar. Kami mencoba memvisualisasikan paraboloid dan garis singgungnya.

```
>function myplot ...
y=0:0.01:1; x=(0.1:0.01:1)';
plot3d(x,y,0.2*(x-0.1)/2,<scale,<frame,>hue, ...
    hues=0.5,>contour,color=orange);
h=holding(1);
plot3d(x,y,(x^2+y^2)/2,<scale,<frame,>contour,>hue);
holding(h);
endfunction
```

Sekarang framedplot() menyediakan frame, dan mengatur tampilan.

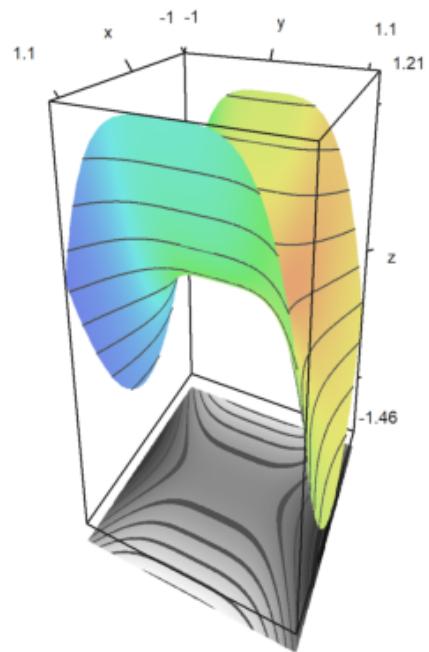
```
>framedplot("myplot",[0.1,1,0,1,0,1],angle=-45°, ...
> center=[0,0,-0.7],zoom=6):
```



Dengan cara yang sama, Anda dapat memplot bidang kontur secara manual. Perhatikan bahwa `plot3d()` menyetel jendela ke `fullwindow()` secara default, tetapi `plotcontourplane()` mengasumsikan itu.

```
>x=-1:0.02:1.1; y=x'; z=x^2-y^4;
>function myplot (x,y,z) ...
    zoom(2);
    wi=fullwindow();
    plotcontourplane(x,y,z,level="auto",<scale);
    plot3d(x,y,z,>hue,<scale,>add,color=white,level="thin");
    window(wi);
    reset();
endfunction
```

```
>myplot(x,y,z):
```

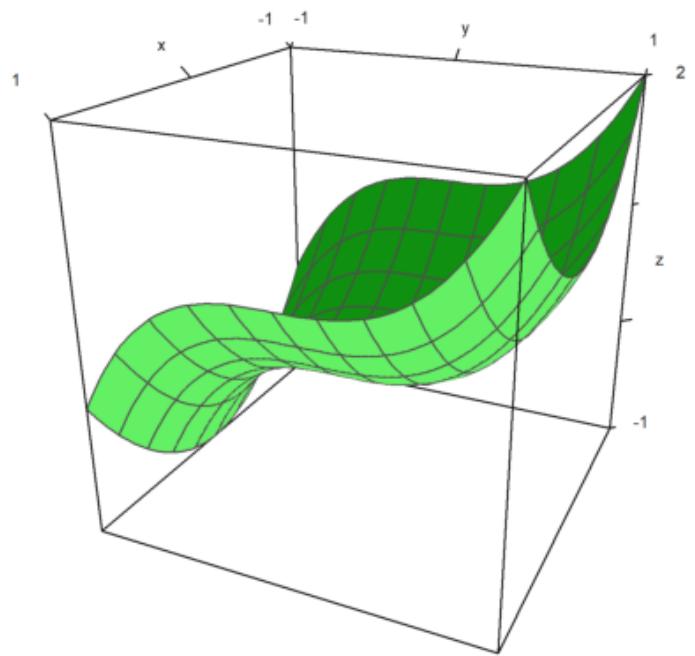


Euler dapat menggunakan frame untuk menghitung animasi terlebih dahulu.

Salah satu fungsi yang memanfaatkan teknik ini adalah rotate. Itu dapat mengubah sudut pandang dan menggambar ulang plot 3D. Fungsi memanggil addpage() untuk setiap plot baru. Akhirnya itu menjiwai plot.

Silakan pelajari sumber rotasi untuk melihat lebih detail.

```
>function testplot () := plot3d("x^2+y^3"); ...
>rotate("testplot"); testplot():
```



Menggambar Povray

---

Dengan bantuan file Euler povray.e, Euler dapat menghasilkan file Povray. Hasilnya sangat bagus untuk dilihat.

Anda perlu menginstal Povray (32bit atau 64bit) dari <http://www.povray.org/>, dan meletakkan sub-direktori "bin" dari Povray ke jalur lingkungan, atau mengatur variabel "defaultpovray" dengan path lengkap yang menunjuk ke "pvengine.exe".

Antarmuka Povray dari Euler menghasilkan file Povray di direktori home pengguna, dan memanggil Povray untuk mengurai file-file ini. Nama file default adalah current.pov, dan direktori default adalah eulerhome(), biasanya c:\Users\Username\Euler. Povray menghasilkan file PNG, yang dapat dimuat oleh Euler ke dalam buku catatan. Untuk membersihkan file-file ini, gunakan povclear().

Fungsi pov3d memiliki semangat yang sama dengan plot3d. Ini dapat menghasilkan grafik fungsi  $f(x,y)$ , atau permukaan dengan koordinat X,Y,Z dalam matriks, termasuk garis level opsional. Fungsi ini memulai raytracer secara otomatis, dan memuat adegan ke dalam notebook Euler.

Selain pov3d(), ada banyak fungsi yang menghasilkan objek Povray. Fungsi-fungsi ini mengembalikan string, yang berisi kode Povray untuk objek. Untuk menggunakan fungsi ini, mulai file Povray dengan povstart(). Kemudian gunakan writeln(...) untuk menulis objek ke file adegan. Terakhir, akhiri file dengan povend(). Secara default, raytracer akan dimulai, dan PNG akan dimasukkan ke dalam notebook Euler.

Fungsi objek memiliki parameter yang disebut "look", yang membutuhkan string dengan kode Povray untuk tekstur dan hasil akhir objek. Fungsi povlook() dapat digunakan untuk menghasilkan string ini. Ini memiliki parameter untuk warna, transparansi, Phong Shading dll.

Perhatikan bahwa alam semesta Povray memiliki sistem koordinat lain. Antarmuka ini menerjemahkan semua koordinat ke sistem Povray. Jadi Anda dapat terus berpikir dalam sistem koordinat Euler dengan z menunjuk vertikal ke atas, dan x,y,z sumbu dalam arti tangan kanan.

Anda perlu memuat file povray.

```
>load povray;
```

Pastikan, direktori bin Povray ada di jalur. Jika tidak, edit variabel berikut sehingga berisi path ke povray yang dapat dieksekusi.

```
>defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe

Untuk kesan pertama, kami memplot fungsi sederhana. Perintah berikut menghasilkan file povray di direktori pengguna Anda, dan menjalankan Povray untuk ray tracing file ini.

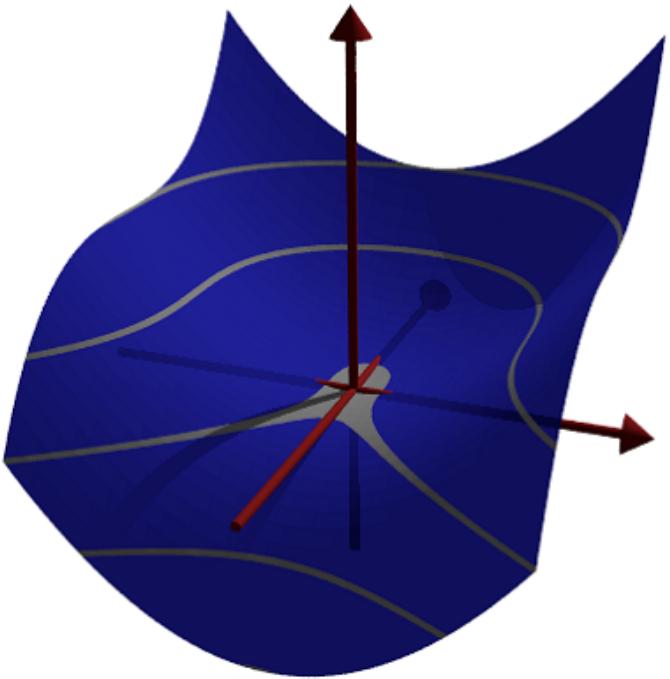
Jika Anda memulai perintah berikut, GUI Povray akan terbuka, menjalankan file, dan menutup secara otomatis. Karena alasan keamanan, Anda akan ditanya, apakah Anda ingin mengizinkan file exe untuk dijalankan. Anda dapat menekan batal untuk menghentikan pertanyaan lebih lanjut. Anda mungkin harus menekan OK di jendela Povray untuk mengakui dialog awal Povray.

```
>pov3d("x^2+y^2",zoom=3);
```



Kita dapat membuat fungsi menjadi transparan dan menambahkan hasil akhir lainnya. Kami juga dapat menambahkan garis level ke plot fungsi.

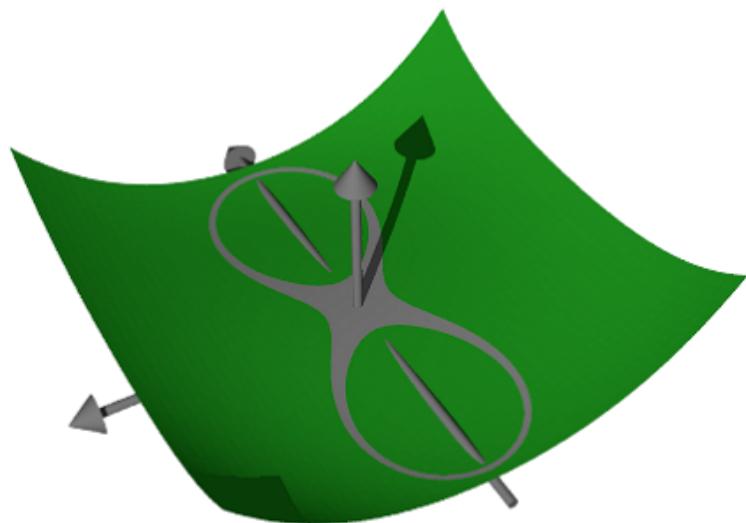
```
>pov3d("x^2+y^3",axiscolor=red,angle=20°, ...
> look=povlook(blue,0.2),level=-1:0.5:1,zoom=3.8);
```



Terkadang perlu untuk mencegah penskalaan fungsi, dan menskalakan fungsi dengan tangan.

Kami memplot himpunan titik di bidang kompleks, di mana produk dari jarak ke 1 dan -1 sama dengan 1.

```
>pov3d("((x-1)^2+y^2)*((x+1)^2+y^2)/40",r=1.5, ...
> angle=-120°,level=1/40,dlevel=0.005,light=[-1,1,1],height=45°,n=50, ...
> <fscale,zoom=3.8);
```



## Merencanakan dengan Koordinat

---

Alih-alih fungsi, kita dapat memplot dengan koordinat. Seperti pada plot3d, kita membutuhkan tiga matriks untuk mendefinisikan objek.

Dalam contoh kita memutar fungsi di sekitar sumbu z.

```
>function f(x) := x^3-x+1; ...
>x=-1:0.01:1; t=linspace(0,2pi,8)'; ...
>Z=x; X=cos(t)*f(x); Y=sin(t)*f(x); ...
>pov3d(X,Y,Z,angle=40°,height=20°,axis=0,zoom=4,light=[10,-5,5]);
```



Dalam contoh berikut, kami memplot gelombang teredam. Kami menghasilkan gelombang dengan bahasa matriks Euler.

Kami juga menunjukkan, bagaimana objek tambahan dapat ditambahkan ke adegan pov3d. Untuk pembuatan objek, lihat contoh berikut. Perhatikan bahwa plot3d menskalakan plot, sehingga cocok dengan kubus satuan.

```
>r=linspace(0,1,80); phi=linspace(0,2pi,80)'; ...
>x=r*cos(phi); y=r*sin(phi); z=exp(-5*r)*cos(8*pi*r)/3; ...
>pov3d(x,y,z,zoom=5,axis=0,add=povsphere([0,0,0.5],0.1,povlook(green)), ...
> w=500,h=300);
```



Dengan metode bayangan canggih dari Povray, sangat sedikit titik yang dapat menghasilkan permukaan yang sangat halus. Hanya di perbatasan dan dalam bayang-bayang triknya mungkin menjadi jelas.

Untuk ini, kita perlu menambahkan vektor normal di setiap titik matriks.

```
>z &= x^2*y^3
```

$$\begin{matrix} 2 & 3 \\ x & y \end{matrix}$$

Persamaan permukaannya adalah  $[x,y,Z]$ . Kami menghitung dua turunan ke  $x$  dan  $y$  ini dan mengambil produk silang sebagai normal.

```
>dx &= diff([x,y,Z],x); dy &= diff([x,y,Z],y);
```

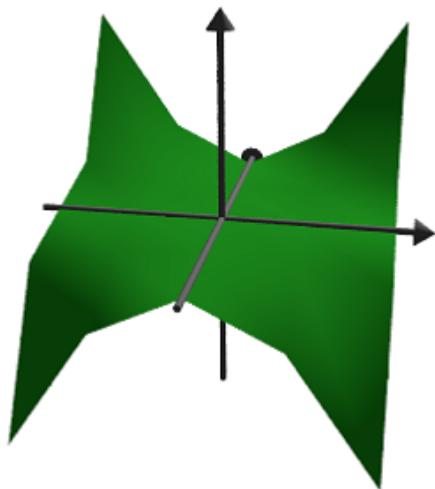
Kami mendefinisikan normal sebagai produk silang dari turunan ini, dan mendefinisikan fungsi koordinat.

```
>N &= crossproduct(dx,dy); NX &= N[1]; NY &= N[2]; NZ &= N[3]; N,
```

$$\begin{bmatrix} 3 & 2 & 2 \\ -2x^3y & -3x^2y & 1 \end{bmatrix}$$

Kami hanya menggunakan 25 poin.

```
>x=-1:0.5:1; y=x';  
>pov3d(x,y,Z(x,y),angle=10°, ...  
> xv=NX(x,y),yv=NY(x,y),zv=NZ(x,y),<shadow>;
```



Berikut ini adalah simpul Trefoil yang dilakukan oleh A. Busser di Povray. Ada versi yang ditingkatkan dari ini dalam contoh.

Lihat: Contoh\Trefoil Simpul | Simpul trefoil

Untuk tampilan yang bagus dengan tidak terlalu banyak titik, kami menambahkan vektor normal di sini. Kami menggunakan Maxima untuk menghitung normal bagi kami. Pertama, ketiga fungsi koordinat sebagai ekspresi simbolik.

```
>X &= ((4+sin(3*y))+cos(x))*cos(2*y); ...
>Y &= ((4+sin(3*y))+cos(x))*sin(2*y); ...
>Z &= sin(x)+2*cos(3*y);
```

Kemudian kedua vektor turunan ke x dan y.

```
>dx &= diff([X,Y,Z],x); dy &= diff([X,Y,Z],y);
```

Sekarang normal, yang merupakan produk silang dari dua turunan.

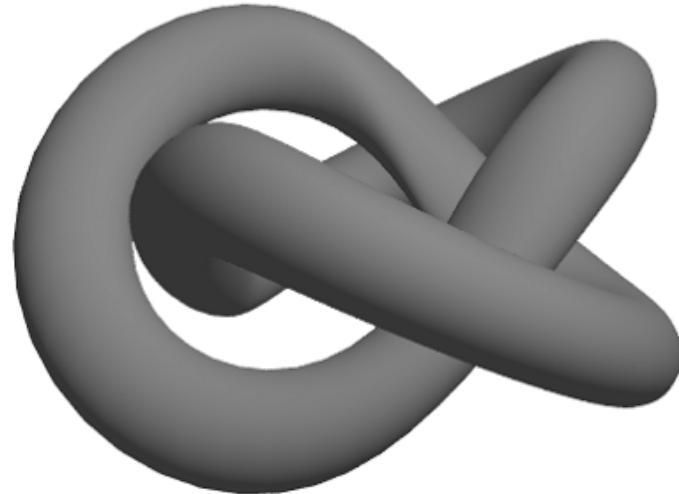
```
>dn &= crossproduct(dx,dy);
```

Kami sekarang mengevaluasi semua ini secara numerik.

```
>x:=linspace(-%pi,%pi,40); y:=linspace(-%pi,%pi,100)';
```

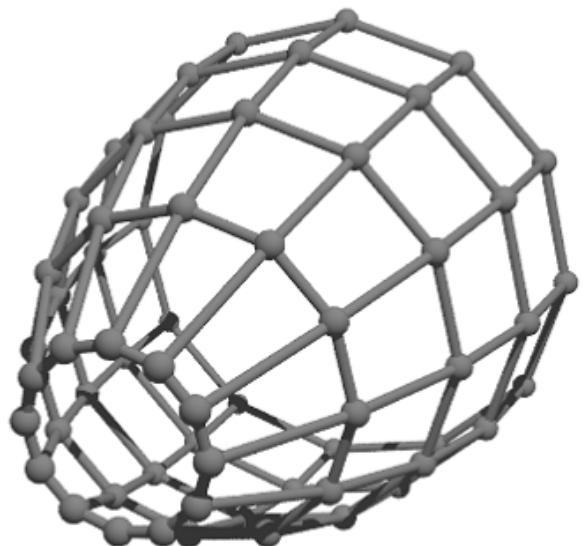
Vektor normal adalah evaluasi dari ekspresi simbolik  $dn[i]$  untuk  $i=1,2,3$ . Sintaks untuk ini adalah &”expression”(parameters). Ini adalah alternatif dari metode pada contoh sebelumnya, di mana kita mendefinisikan ekspresi simbolik NX, NY, NZ terlebih dahulu.

```
>pov3d(X(x,y),Y(x,y),Z(x,y),axis=0,zoom=5,w=450,h=350, ...
>  <shadow,look=povlook(gray), ...
>  xv=&"dn[1] "(x,y), yv=&"dn[2] "(x,y), zv=&"dn[3] "(x,y));
```



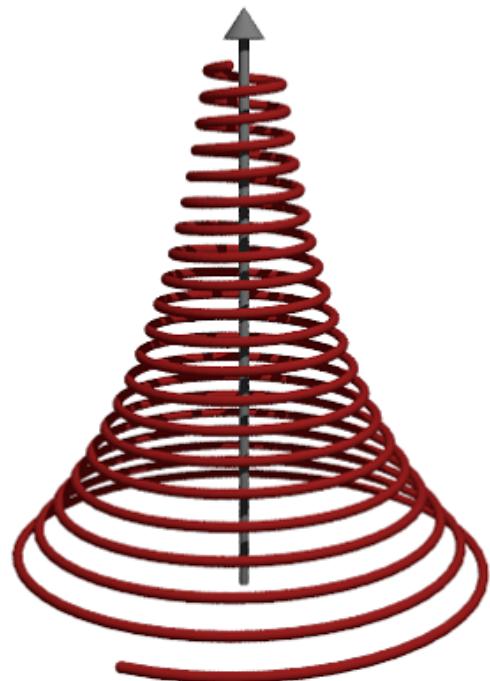
Kami juga dapat menghasilkan grid dalam 3D.

```
>povstart(zoom=4); ...
>x=-1:0.5:1; r=1-(x+1)^2/6; ...
>t=(0°:30°:360°)'; y=r*cos(t); z=r*sin(t); ...
>writeln(povgrid(x,y,z,d=0.02,dballs=0.05)); ...
>povend();
```



Dengan povgrid(), kurva dimungkinkan.

```
>povstart(center=[0,0,1],zoom=3.6); ...
>t=linspace(0,2,1000); r=exp(-t); ...
>x=cos(2*pi*10*t)*r; y=sin(2*pi*10*t)*r; z=t; ...
>writeln(povgrid(x,y,z,povlook(red))); ...
>writeAxis(0,2,axis=3); ...
>povend();
```



## Objek Povray

---

Di atas, kami menggunakan pov3d untuk memplot permukaan. Antarmuka povray di Euler juga dapat menghasilkan objek Povray. Objek-objek ini disimpan sebagai string di Euler, dan perlu ditulis ke file Povray.

Kami memulai output dengan povstart().

```
>povstart(zoom=4);
```

Pertama kita mendefinisikan tiga silinder, dan menyimpannya dalam string di Euler.

Fungsi povx() dll. hanya mengembalikan vektor [1,0,0], yang dapat digunakan sebagai gantinya.

```
>c1=povcylinder(-povx,povx,1,povlook(red)); ...
>c2=povcylinder(-povy,povy,1,povlook(green)); ...
>c3=povcylinder(-povz,povz,1,povlook(blue)); ...
```

Pertama kita mendefinisikan tiga silinder, dan menyimpannya dalam string di Euler.

Fungsi povx() dll. hanya mengembalikan vektor [1,0,0], yang dapat digunakan sebagai pengingat.

```
>c1
```

```
cylinder { <-1,0,0>, <1,0,0>, 1
    texture { pigment { color rgb <0.564706,0.0627451,0.0627451> } }
    finish { ambient 0.2 }
}
```

Seperti yang Anda lihat, kami menambahkan tekstur ke objek dalam tiga warna berbeda.

Itu dilakukan oleh povlook(), yang mengembalikan string dengan kode Povray yang relevan. Kita dapat menggunakan warna Euler default, atau menentukan warna kita sendiri. Kami juga dapat menambahkan transparansi, atau mengubah cahaya sekitar.

```
>povlook(rgb(0.1,0.2,0.3),0.1,0.5)
```

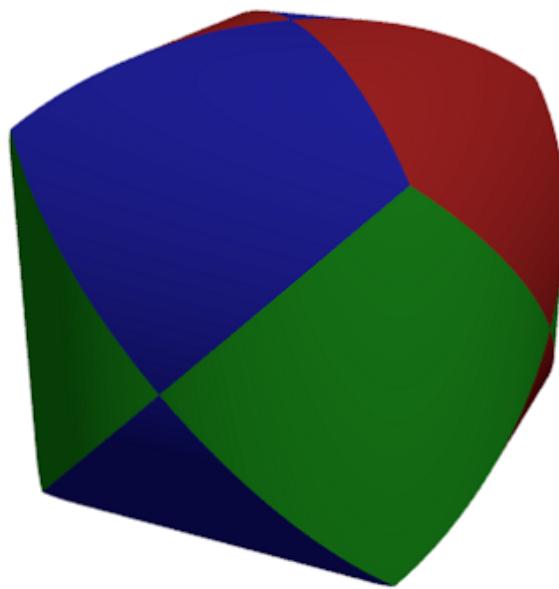
```
texture { pigment { color rgbf <0.101961,0.2,0.301961,0.1> } }
finish { ambient 0.5 }
```

Sekarang kita mendefinisikan objek persimpangan, dan menulis hasilnya ke file.

```
>writeln(povintersection([c1,c2,c3]));
```

Persimpangan tiga silinder sulit untuk divisualisasikan, jika Anda belum pernah melihatnya sebelumnya.

```
>povend;
```



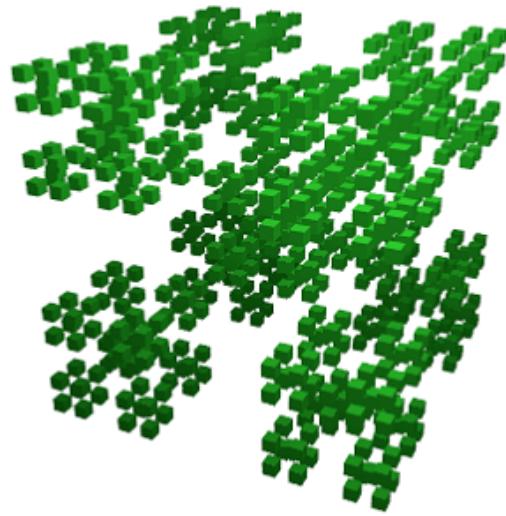
Fungsi berikut menghasilkan fraktal secara rekursif.

Fungsi pertama menunjukkan, bagaimana Euler menangani objek Povray sederhana. Fungsi povbox() mengembalikan string, yang berisi koordinat kotak, tekstur, dan hasil akhir.

```
>function onebox(x,y,z,d) := povbox([x,y,z],[x+d,y+d,z+d],povlook());  
>function fractal (x,y,z,h,n) ...
```

```
if n==1 then writeln(onebox(x,y,z,h));  
else  
    h=h/3;  
    fractal(x,y,z,h,n-1);  
    fractal(x+2*h,y,z,h,n-1);  
    fractal(x,y+2*h,z,h,n-1);  
    fractal(x,y,z+2*h,h,n-1);  
    fractal(x+2*h,y+2*h,z,h,n-1);  
    fractal(x+2*h,y,z+2*h,h,n-1);  
    fractal(x,y+2*h,z+2*h,h,n-1);  
    fractal(x+2*h,y+2*h,z+2*h,h,n-1);  
    fractal(x+h,y+h,z+h,h,n-1);  
endif;  
endfunction
```

```
>povstart(fade=10,<shadow);  
>fractal(-1,-1,-1,2,4);  
>povend();
```



Perbedaan memungkinkan memotong satu objek dari yang lain. Seperti persimpangan, ada bagian dari objek CSG Povray.

```
>povstart(light=[5,-5,5],fade=10);
```

Untuk demonstrasi ini, kami mendefinisikan objek di Povray, alih-alih menggunakan string di Euler. Definisi ditulis ke file segera.

Koordinat kotak -1 berarti [-1,-1,-1].

```
>povdefine("mycube",povbox(-1,1));
```

Kita dapat menggunakan objek ini di povobject(), yang mengembalikan string seperti biasa.

```
>c1=povobject("mycube",povlook(red));
```

Kami menghasilkan kubus kedua, dan memutar dan menskalakannya sedikit.

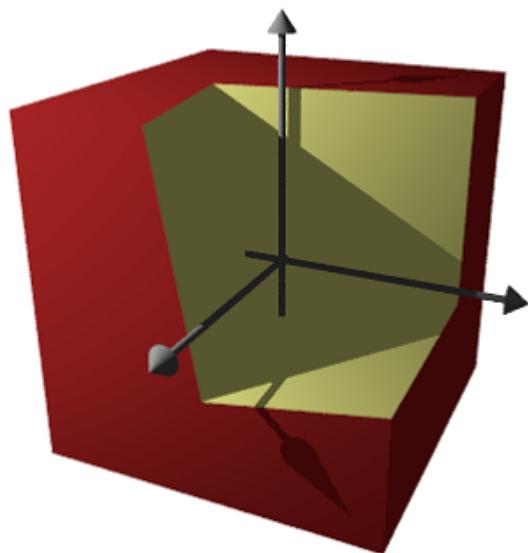
```
>c2=povobject("mycube",povlook(yellow),translate=[1,1,1], ...
>   rotate=xrotate(10°)+yrotate(10°), scale=1.2);
```

Kemudian kita ambil selisih kedua benda tersebut.

```
>writeln(povdifference(c1,c2));
```

Sekarang tambahkan tiga sumbu.

```
>writeAxis(-1.2,1.2,axis=1); ...
>writeAxis(-1.2,1.2,axis=2); ...
>writeAxis(-1.2,1.2,axis=4); ...
>povend();
```



## Fungsi Implisit

---

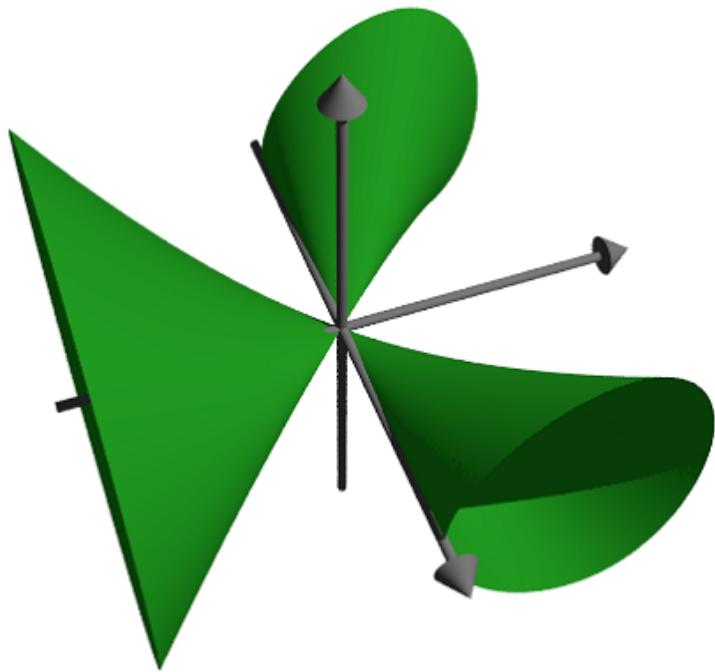
Povray dapat memplot himpunan di mana  $f(x,y,z)=0$ , seperti parameter implisit di plot3d. Namun, hasilnya terlihat jauh lebih baik.

Sintaks untuk fungsinya sedikit berbeda. Anda tidak dapat menggunakan output dari ekspresi Maxima atau Euler.

```
>povstart(angle=70°,height=50°,zoom=4);
```

Buat permukaan implisit. Perhatikan sintaks yang berbeda dalam ekspresi.

```
>writeln(povsurface("pow(x,2)*y-pow(y,3)-pow(z,2)",povlook(green))); ...
>writeAxes(); ...
>povend();
```



Dalam contoh ini, kami menunjukkan cara membuat objek mesh, dan menggambarnya dengan informasi tambahan.

Kami ingin memaksimalkan xy di bawah kondisi  $x+y=1$  dan menunjukkan sentuhan tangensial dari garis level.

```
>povstart(angle=-10°,center=[0.5,0.5,0.5],zoom=7);
```

Kami tidak dapat menyimpan objek dalam string seperti sebelumnya, karena terlalu besar. Jadi kita mendefinisikan objek dalam file Povray menggunakan declare. Fungsi povtriangle() melakukan ini secara otomatis. Itu dapat menerima vektor normal seperti pov3d().

Berikut ini mendefinisikan objek mesh, dan langsung menulisnya ke dalam file.

```
>x=0:0.02:1; y=x'; z=x*y; vx=-y; vy=-x; vz=1;  
>mesh=povtriangles(x,y,z,"",vx,vy,vz);
```

Sekarang kita mendefinisikan dua buah cakram, yang akan berpotongan dengan permukaan.

```
>cl=povdisc([0.5,0.5,0],[1,1,0],2); ...  
>ll=povdisc([0,0,1/4],[0,0,1],2);
```

Tuliskan permukaan dikurangi dengan dua cakram.

```
>writeln(povdifference(mesh,povunion([cl,ll]),povlook(green)));
```

Tuliskan dua persimpangan

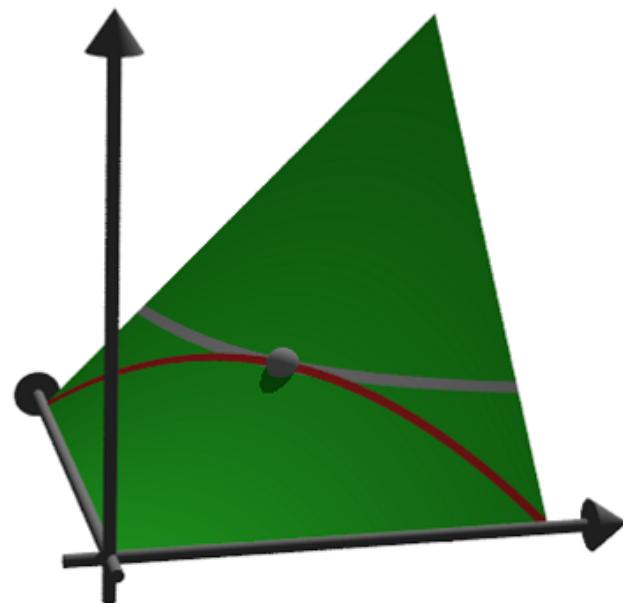
```
>writeln(povintersection([mesh,cl],povlook(red))); ...
>writeln(povintersection([mesh,ll],povlook(gray)));
```

Menuliskan sebuah titik secara maksimal.

```
>writeln(povpoint([1/2,1/2,1/4],povlook(gray),size=2*defaultpointsize));
```

Tambahkan sumbu dan selesaikan.

```
>writeAxes(0,1,0,1,0,1,d=0.015); ...
>povend();
```



## Anaglyphs di Povray

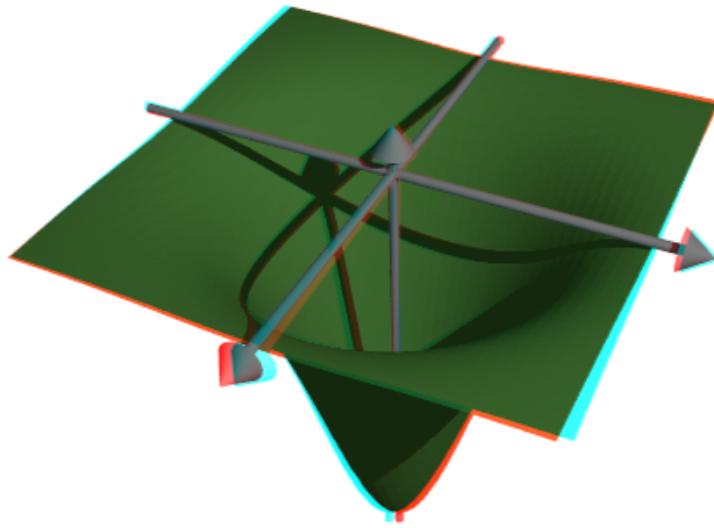
---

Untuk menghasilkan anaglyph untuk kacamata merah/cyan, Povray harus dijalankan dua kali dari posisi kamera yang berbeda. Ini menghasilkan dua file Povray dan dua file PNG, yang dimuat dengan fungsi loadanaglyph().

Tentu saja, Anda membutuhkan kacamata merah/cyan untuk melihat contoh berikut dengan benar.

Fungsi pov3d() memiliki sebuah saklar sederhana untuk menghasilkan anaglyph.

```
>pov3d("-exp(-x^2-y^2)/2",r=2,height=45°,>anaglyph, ...
> center=[0,0,0.5],zoom=3.5);
```



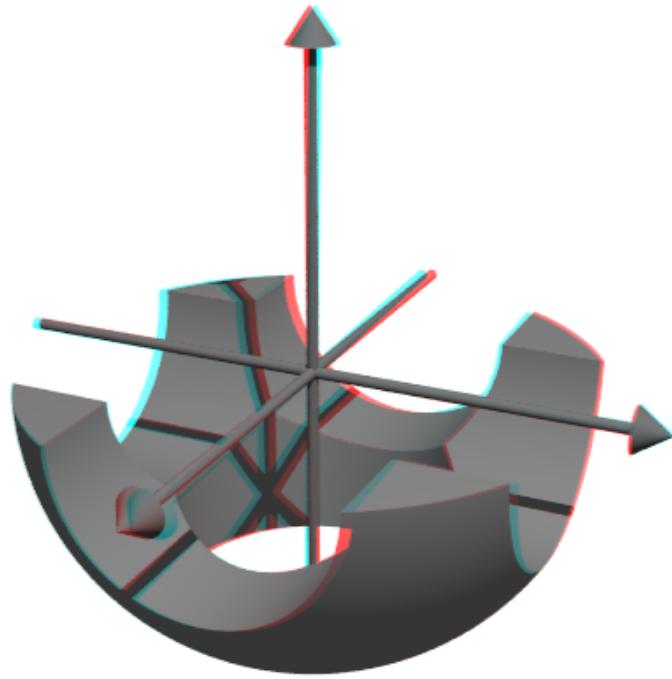
Jika Anda membuat adegan dengan objek, Anda perlu memasukkan pembuatan adegan ke dalam sebuah fungsi, dan jalankan dua kali dengan nilai yang berbeda untuk parameter anaglyph.

```
>function myscene ...
```

```
s=povsphere(povc,1);
cl=povcylinder(-povz,povz,0.5);
clk=povobject(cl,rotate=xrotate(90°));
clv=povobject(cl,rotate=yrotate(90°));
c=povbox([-1,-1,0],1);
un=povunion([cl,clk,clv,c]);
obj=povdifference(s,un,povlook(red));
writeln(obj);
writeAxes();
endfunction
```

Fungsi povanaglyph() melakukan semua ini. Parameter-parameternya adalah seperti di dalam povstart() dan povend() digabungkan.

```
>povanaglyph("myscene",zoom=4.5);
```



## Mendefinisikan Objek Sendiri

---

Antarmuka povray Euler berisi banyak objek. Tapi Anda tidak terbatas pada objek-objek tersebut. Anda dapat membuat objek sendiri, yang menggabungkan objek lain, atau objek yang benar-benar baru.

Kami mendemonstrasikan sebuah torus. Perintah Povray untuk ini adalah “torus”. Jadi kami mengembalikan sebuah string dengan perintah ini dan parameternya. Perhatikan bahwa torus selalu berpusat pada titik asal.

```
>function povdonat (r1,r2,look="") ...  
  
    return "torus {" + r1 + "," + r2 + look + "}";  
endfunction
```

Ini adalah torus pertama kita.

```
>t1=povdonat(0.8,0.2)  
  
torus {0.8,0.2}
```

Mari kita gunakan objek ini untuk membuat torus kedua, ditranslasikan dan diputar.

```
>t2=povobject(t1,rotate=xrotate(90°),translate=[0.8,0,0])
```

```
object { torus {0.8,0.2}
    rotate 90 *x
    translate <0.8,0,0>
}
```

Sekarang kita tempatkan objek-objek ini ke dalam sebuah scene. Untuk tampilan, kita menggunakan Phong

```
>povstart(center=[0.4,0,0],angle=0°,zoom=3.8,aspect=1.5); ...
>writeln(povobject(t1,povlook(green,phong=1))); ...
>writeln(povobject(t2,povlook(green,phong=1))); ...
```

```
>povend();
```

memanggil program Povray. Namun, jika terjadi kesalahan program ini tidak menampilkan kesalahan. Oleh karena itu, Anda harus menggunakan

```
>povend(<exit>);
```

jika ada yang tidak berhasil. Ini akan membiarkan jendela Povray terbuka.

```
>povend(h=320,w=480);
```



Berikut adalah contoh yang lebih rumit. Kami menyelesaikan

$$Ax \leq b, \quad x \geq 0, \quad c.x \rightarrow \text{Max.}$$

dan menunjukkan titik-titik yang layak dan optimal dalam plot 3D.

```
>A=[10,8,4;5,6,8;6,3,2;9,5,6];
>b=[10,10,10,10]';
>c=[1,1,1];
```

Pertama, mari kita periksa, apakah contoh ini memiliki solusi atau tidak.

```
>x=simplex(A,b,c,>max,>check)'
```

```
[0, 1, 0.5]
```

Ya, ada.

Selanjutnya kita mendefinisikan dua objek. Yang pertama adalah pesawat

$$a \cdot x \leq b$$

```
>function oneplane (a,b,look=""') ...
```

```
    return povplane(a,b,look)
endfunction
```

Kemudian kita mendefinisikan perpotongan dari semua setengah ruang dan sebuah kubus.

```
>function adm (A, b, r, look="") ...
ol=[];
loop 1 to rows(A); ol=ol|oneplane(A[#,b[#]); end;
ol=ol|povbox([0,0,0],[r,r,r]);
return povintersection(ol,look);
endfunction
```

Kita sekarang dapat memplot adegan.

```
>povstart(angle=120°,center=[0.5,0.5,0.5],zoom=3.5); ...
>writeln(adm(A,b,2,povlook(green,0.4))); ...
>writeAxes(0,1.3,0,1.6,0,1.5); ...
```

Berikut ini adalah sebuah lingkaran di sekitar titik optimal.

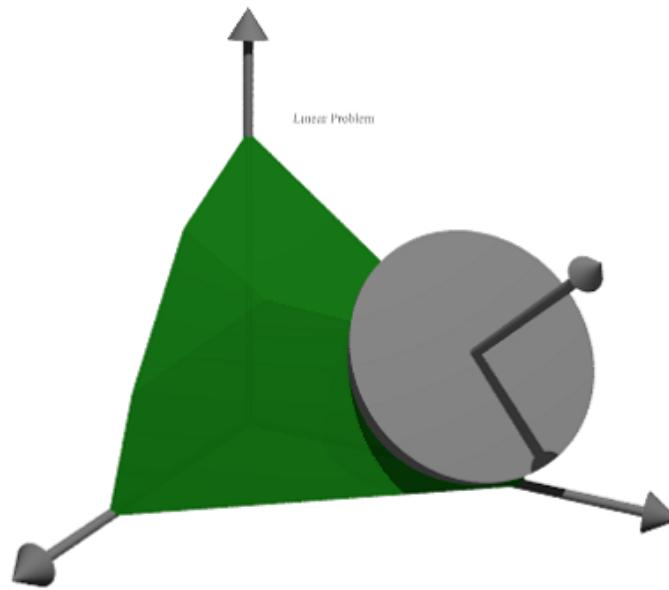
```
>writeln(povintersection([povsphere(x,0.5),povplane(c,c.x')], ...
> povlook(red,0.9)));
```

Dan kesalahan ke arah optimum

```
>writeln(povarrow(x,c*0.5,povlook(red)));
```

Kami menambahkan teks ke layar. Teks hanyalah sebuah objek 3D. Kita perlu menempatkan dan mengubahnya sesuai dengan pandangan kita.

```
>writeln(povtext("Linear Problem",[0,0.2,1.3],size=0.05,rotate=125°)); ...
>povend();
```



Contoh Lainnya

Anda dapat menemukan beberapa contoh lain untuk Povray di Euler dalam file-file berikut.

Lihat: Examples/Dandelin Spheres

Lihat: Contoh/Contoh/Donat Matematika

Lihat: Contoh/Simpul Trefoil

Lihat: Contoh/Optimalisasi dengan Penskalaan Affine

## Contoh Soal

---

Grafik dari fungsi  $f$  dengan dua variabel yang dimaksud adalah grafik dari persamaan  $z = f(x,y)$ . Biasanya grafik ini berupa permukaan dan karena setiap  $(x,y)$  di daerah asal hanya berpadanan dengan satu nilai  $z$ , maka setiap garis tegak lurus bidang-xy memotong permukaan pada paling banyak satu titik.

Soal :

1. Grafik merupakan sebuah paraboloida

$$f(x,y) = y^2 - x^2$$

2.

$$z = -4x^3y^2$$

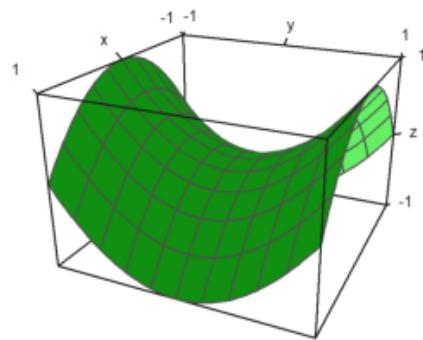
3.

$$z = xy\exp(-x^2 - y^2)$$

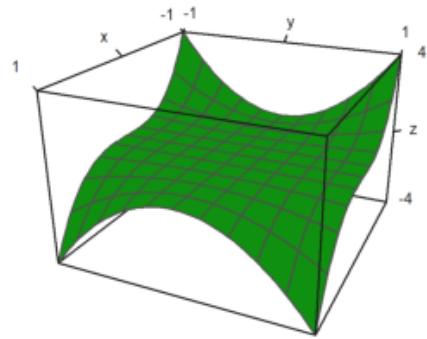
4.

$$z = x - 1/8x^3 - 1/3y^2$$

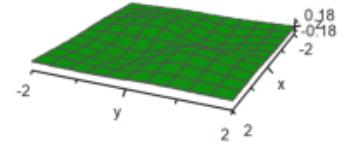
```
> aspect(1.5); plot3d("y^2-x^2"):
```



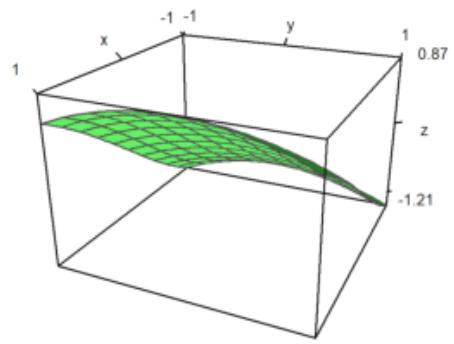
```
> aspect(1.5); plot3d("-4x^3*y^2"):
```



```
> plot3d("x*y*exp(-x^2-y^2)",r=2,<fscale,<scale,distance=13,height=20°, ...
> center=[0,0,-0.2],frame=3):
```



```
> aspect(1.5); plot3d("x-1/8*x^3-1/3*y^2"):
```



# Kalkulus dengan EMT

---

Materi Kalkulus mencakup di antaranya:

- Fungsi (fungsi aljabar, trigonometri, eksponensial, logaritma, komposisi fungsi)
- Limit Fungsi,
- Turunan Fungsi,
- Integral Tak Tentu,
- Integral Tentu dan Aplikasinya,
- Barisan dan Deret (kekonvergenan barisan dan deret).

EMT (bersama Maxima) dapat digunakan untuk melakukan semua perhitungan di dalam kalkulus, baik secara numerik maupun analitik (eksak).

## Mendefinisikan Fungsi

---

Terdapat beberapa cara mendefinisikan fungsi pada EMT, yakni:

- Menggunakan format `nama_fungsi := rumus fungsi` (untuk fungsi numerik),
- Menggunakan format `nama_fungsi &= rumus fungsi` (untuk fungsi simbolik, namun dapat dihitung secara numerik),
- Menggunakan format `nama_fungsi &&= rumus fungsi` (untuk fungsi simbolik murni, tidak dapat dihitung langsung),
- Fungsi sebagai program EMT.

Setiap format harus diawali dengan perintah `function` (bukan sebagai ekspresi).

Berikut adalah beberapa contoh cara mendefinisikan fungsi.

```
>function f(x) := 2*x^2+exp(sin(x)) // fungsi numerik  
>f(0), f(1), f(pi)
```

1  
4.31977682472  
20.7392088022

```
>function g(x) := sqrt(x^2-3*x)/(x+1)  
>g(3)
```

0

```
>g(0)
```

0

```
>g(f(1))
```

0.448835801122

```
>f(g(5)) // komposisi fungsi
```

2.20920171961

```
>g(f(5))
```

0.950898070639

```
>function h(x) := f(g(x)) // definisi komposisi fungsi  
>h(5) // sama dengan f(g(5))
```

2.20920171961

```
>f(0:10) // nilai-nilai f(1), f(2), ..., f(10)
```

[1, 4.31978, 10.4826, 19.1516, 32.4692, 50.3833, 72.7562,  
99.929, 130.69, 163.51, 200.58]

```
>fmap(0:10) // sama dengan f(0:10), berlaku untuk semua fungsi
```

```
[1, 4.31978, 10.4826, 19.1516, 32.4692, 50.3833, 72.7562,  
99.929, 130.69, 163.51, 200.58]
```

Misalkan kita akan mendefinisikan fungsi

$$f(x) = \begin{cases} x^3 & x > 0 \\ x^2 & x \leq 0. \end{cases}$$

Fungsi tersebut tidak dapat didefinisikan sebagai fungsi numerik secara "inline" menggunakan format `:=`, melainkan didefinisikan sebagai program. Perhatikan, kata "map" digunakan agar fungsi dapat menerima vektor sebagai input, dan hasilnya berupa vektor. Jika tanpa kata "map" fungsinya hanya dapat menerima input satu nilai.

```
>function map f(x) ...
```

```
    if x>0 then return x^3  
    else return x^2  
    endif;  
endfunction
```

```
>f(1)
```

1

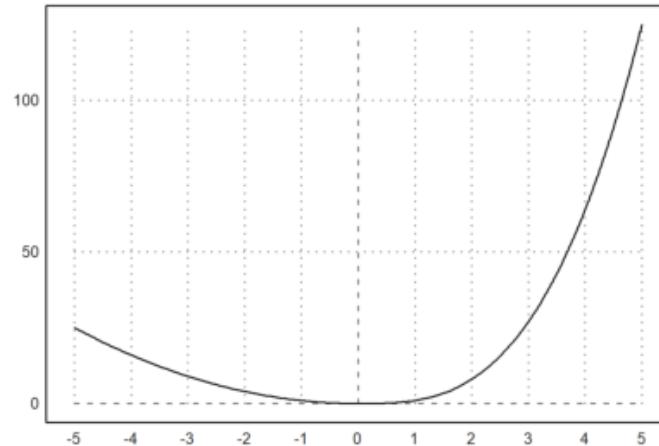
```
>f(-2)
```

4

```
>f(-5:5)
```

```
[25, 16, 9, 4, 1, 0, 1, 8, 27, 64, 125]
```

```
>aspect(1.5); plot2d("f(x)", -5, 5):
```



```
>function f(x) &= 2*E^x // fungsi simbolik
```

$$\frac{x}{2} E$$

```
>function g(x) &= 3*x+1
```

$$3x + 1$$

```
>function h(x) &= f(g(x)) // komposisi fungsi
```

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

Bukalah buku Kalkulus. Cari dan pilih beberapa (paling sedikit 5 fungsi berbeda tipe/bentuk/jenis) fungsi dari buku tersebut, kemudian definisikan di EMT pada baris-baris perintah berikut (jika perlu tambahkan lagi). Untuk setiap fungsi, hitung beberapa nilainya, baik untuk satu nilai maupun vektor. Gambar grafik tersebut.

Juga, carilah fungsi beberapa (dua) variabel. Lakukan hal sama seperti di atas.

Jawab:

### A). FUNGSI 1 VARIABEL

#### 1. Fungsi 1

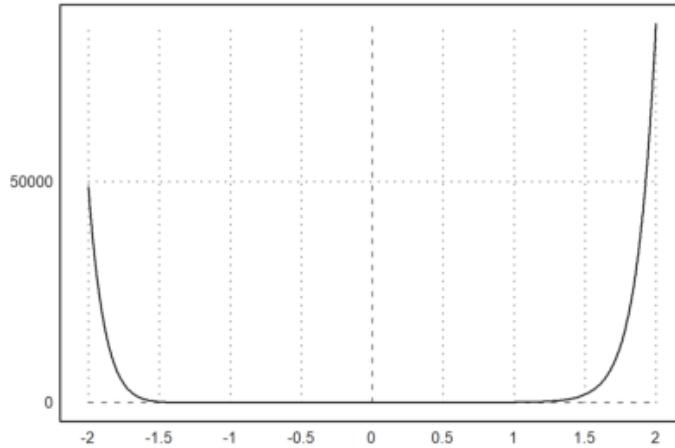
```
>function k(x) := x*(x^5+3)^3  
>k(3), k(5), k(7)
```

```
44660808  
153027765760  
3.3250729687e+13
```

```
>kmap(-3:3)
```

```
[4.1472e+07, 48778, -8, 0, 64, 85750, 4.46608e+07]
```

```
>plot2d("k(x)":
```



## 2. Fungsi 2

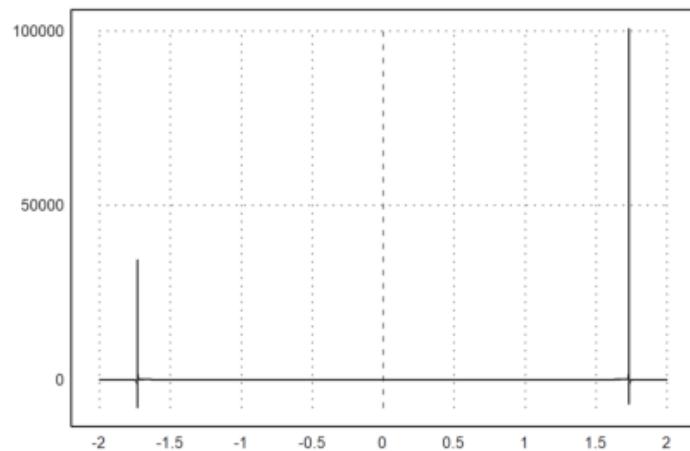
```
>function m(x) := (x)^4/(3-x^2)
>m(2), m(-2), m(1)
```

```
-16
-16
0.5
```

```
>mmap(-5:-5)
```

-28.4090909091

```
>plot2d("m(x)":
```



### 3. Fungsi 3

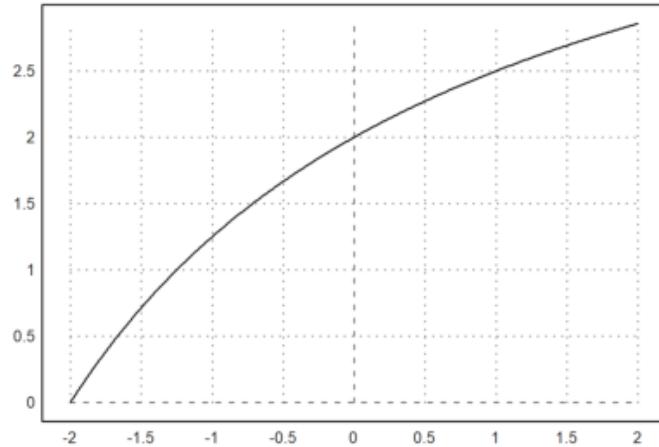
```
>function n(x) := 3*x/(x+5)+2  
>n(2), n(-1), n(-3), n(4)
```

```
2.85714285714  
1.25  
-2.5  
3.33333333333
```

```
>nmap(2:5)
```

```
[2.85714, 3.125, 3.33333, 3.5]
```

```
>plot2d("n(x)":
```



#### 4. Fungsi 4

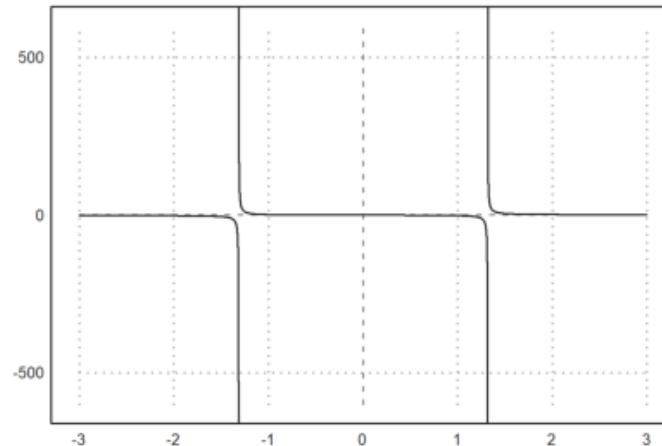
```
>function l(x) := 3*x^3/(x^4-3)
>l(5), l(4), l(3)
```

0.602893890675  
0.758893280632  
1.03846153846

```
>lmap(5:8)
```

```
[0.602894,  0.50116,  0.429108,  0.375275]
```

```
>plot2d("l(x)",-3,3,-600,600):
```



## 5. Fungsi 5

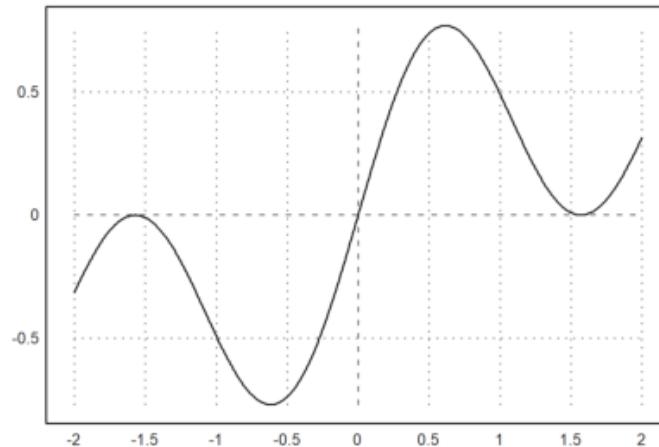
```
>function j(x) := (cos(x))*sin(2*x)
>j(pi), j(0), j(pi/3)
```

```
0  
0  
0.433012701892
```

```
>jmap(0:3pi)
```

```
[0, 0.491295, 0.314941, 0.276619, -0.646688, -0.154318,  
-0.515201, 0.746821, 0.0418899, 0.684247]
```

```
>plot2d("j(x)":
```



## 6. Fungsi 6

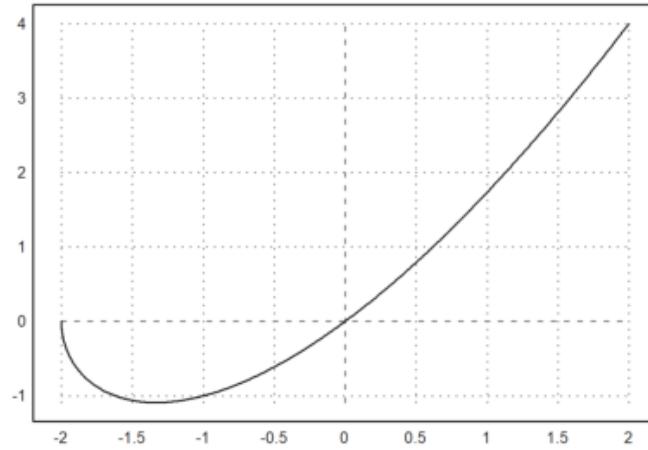
```
>function o(x) := x*sqrt(x+2)
>o(3), o(5), o(7)
```

```
6.7082039325
13.2287565553
21
```

```
>omap(3:12)
```

```
[6.7082, 9.79796, 13.2288, 16.9706, 21, 25.2982, 29.8496,
34.641, 39.6611, 44.8999]
```

```
>plot2d("o(x)":
```



## B). FUNGSI 2 VARIABEL

### 1. Fungsi 1

```
>function a(x,y) ...
```

```
    return x^2+y^2-24  
endfunction
```

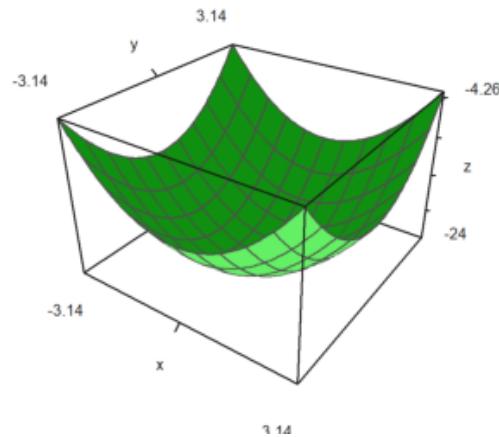
```
>a(2,1), a(5,4), a(2,4)
```

```
-19  
17  
-4
```

```
>amap(-2:2,3:3)
```

```
[-11, -14, -15, -14, -11]
```

```
>aspect=1.5; plot3d("a(x,y)",a=-100,b=100,c=-80,d=80,angle=35°,height=30°,r=pi,n=100):
```



## 2. Fungsi 2

```
>function q(x,y) ...
```

```
    return y^2/(x^2/3)
    endfunction
```

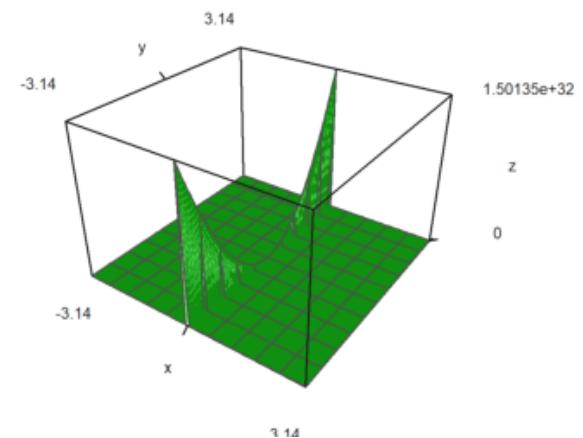
```
>q(4,2), q(2,3), q(4,3)
```

```
0.75
6.75
1.6875
```

```
>qmap(2:2,-2:2)
```

```
[3, 0.75, 0, 0.75, 3]
```

```
>aspect=1.5; plot3d("q(x,y)",a=-100,b=100,c=-80,d=80,angle=35°,height=30°,r=pi,n=100):
```



## Menghitung Limit

---

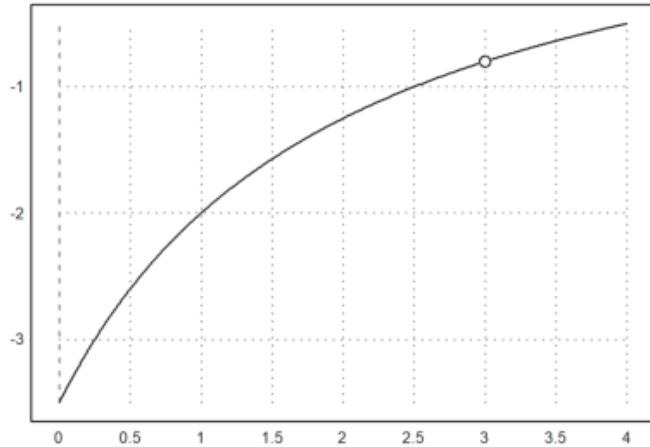
Perhitungan limit pada EMT dapat dilakukan dengan menggunakan fungsi Maxima, yakni "limit". Fungsi "limit" dapat digunakan untuk menghitung limit fungsi dalam bentuk ekspresi maupun fungsi yang sudah didefinisikan sebelumnya. Nilai limit dapat dihitung pada sebarang nilai atau pada tak hingga (-inf, minf, dan inf). Limit kiri dan limit kanan juga dapat dihitung, dengan cara memberi opsi "plus" atau "minus". Hasil limit dapat berupa nilai, "und" (tak definisi), "ind" (tak tentu namun terbatas), "infinity" (kompleks tak hingga).

Perhatikan beberapa contoh berikut. Perhatikan cara menampilkan perhitungan secara lengkap, tidak hanya menampilkan hasilnya saja.

```
>$limit((x^3-13*x^2+51*x-63)/(x^3-4*x^2-3*x+18),x,3)
```

$$-\frac{4}{5}$$

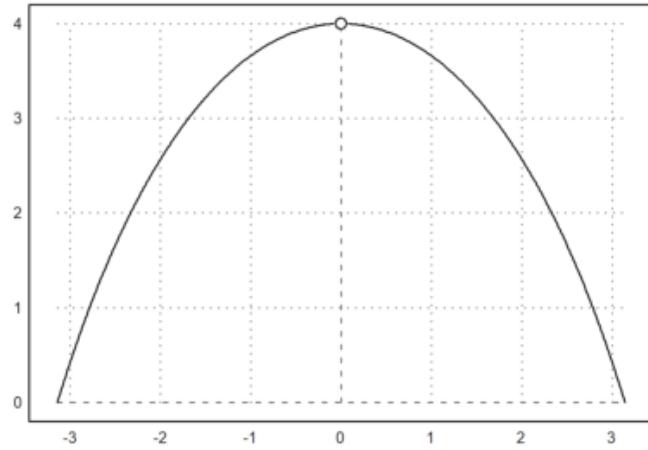
```
>aspect(1.5); plot2d("(x^3-13*x^2+51*x-63)/(x^3-4*x^2-3*x+18)",0,4); plot2d(3,-4/5,>points,style="ow")
```



```
>$limit(2*x*sin(x)/(1-cos(x)),x,0)
```

4

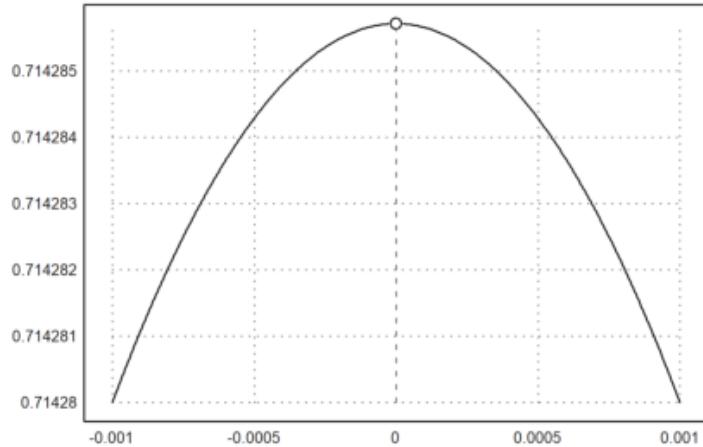
```
>plot2d("2*x*sin(x)/(1-cos(x))",-pi,pi); plot2d(0,4,>points,style="ow",>add):
```



```
>$limit(cot(7*h)/cot(5*h),h,0)
```

$$\frac{5}{7}$$

```
>plot2d("cot(7*x)/cot(5*x)",-0.001,0.001); plot2d(0,5/7,>points,style="ow",>add):
```



```
>$showev('limit(1/(2*x-1),x,0))
```

$$\lim_{x \rightarrow 0} \frac{1}{2x - 1} = -1$$

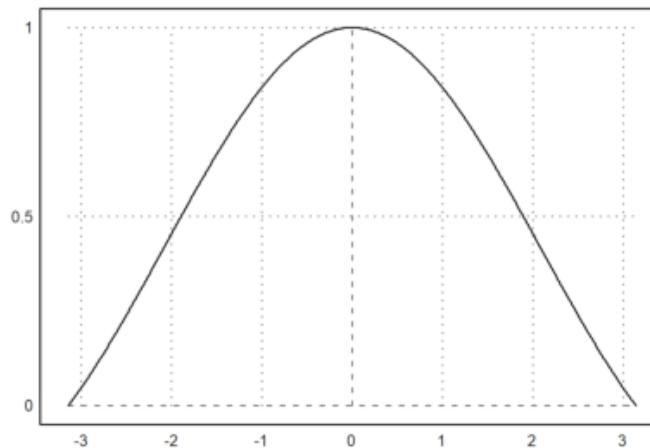
```
>$showev('limit((x^2-3*x-10)/(x-5),x,5))
```

$$\lim_{x \rightarrow 5} \frac{x^2 - 3x - 10}{x - 5} = 7$$

```
>$showev('limit(sin(x)/x,x,0))
```

$$\lim_{x \rightarrow 0} \frac{\sin x}{x} = 1$$

```
>plot2d("sin(x)/x",-pi,pi):
```



```
>$showev('limit(sin(x^3)/x,x,0))
```

$$\lim_{x \rightarrow 0} \frac{\sin x^3}{x} = 0$$

```
>$showev('limit(log(x), x, minf))
```

$$\lim_{x \rightarrow -\infty} \log x = \text{infinity}$$

```
>$showev('limit((-2)^x,x, inf))
```

$$\lim_{x \rightarrow \infty} (-2)^x = \text{infinity}$$

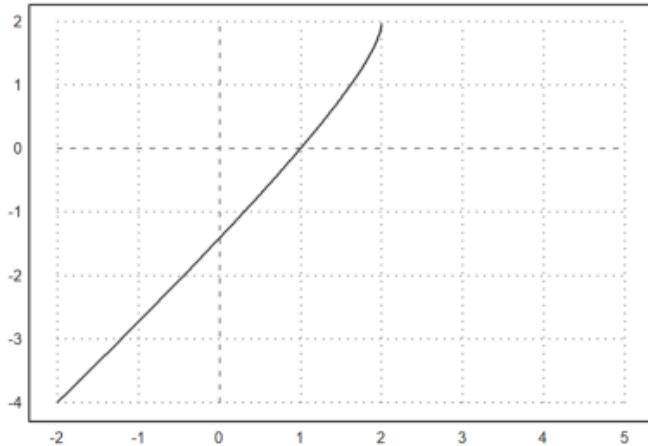
```
>$showev('limit(t-sqrt(2-t),t,2,minus))
```

$$\lim_{t \uparrow 2} t - \sqrt{2-t} = 2$$

```
>$showev('limit(t-sqrt(2-t),t,5,plus)) // Perhatikan hasilnya
```

$$\lim_{t \downarrow 5} t - \sqrt{2-t} = 5 - \sqrt{3} i$$

```
>plot2d("x-sqrt(2-x)",-2,5):
```



```
>$showev('limit((x^2-9)/(2*x^2-5*x-3),x,3))
```

$$\lim_{x \rightarrow 3} \frac{x^2 - 9}{2x^2 - 5x - 3} = \frac{6}{7}$$

```
>$showev('limit((1-cos(x))/x,x,0))
```

$$\lim_{x \rightarrow 0} \frac{1 - \cos x}{x} = 0$$

```
>$showev('limit((x^2+abs(x))/(x^2-abs(x)),x,0))
```

$$\lim_{x \rightarrow 0} \frac{|x| + x^2}{x^2 - |x|} = -1$$

```
>$showev('limit((1+1/x)^x,x,inf))
```

$$\lim_{x \rightarrow \infty} \left( \frac{1}{x} + 1 \right)^x = e$$

```
>$showev('limit((1+k/x)^x,x,inf))
```

$$\lim_{x \rightarrow \infty} \left( \frac{k}{x} + 1 \right)^x = e^k$$

```
>$showev('limit((1+x)^(1/x),x,0))
```

$$\lim_{x \rightarrow 0} (x + 1)^{\frac{1}{x}} = e$$

```
>$showev('limit((x/(x+k))^x,x,inf))
```

$$\lim_{x \rightarrow \infty} \left( \frac{x}{x+k} \right)^x = e^{-k}$$

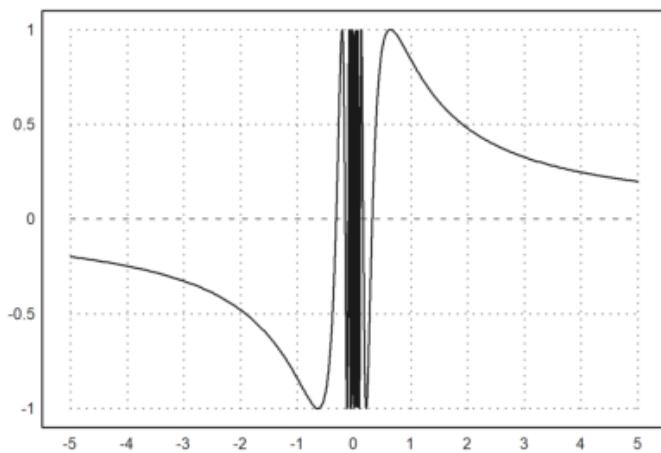
```
>$showev('limit(sin(1/x),x,0))
```

$$\lim_{x \rightarrow 0} \sin\left(\frac{1}{x}\right) = \text{ind}$$

```
>$showev('limit(sin(1/x),x,inf))
```

$$\lim_{x \rightarrow \infty} \sin\left(\frac{1}{x}\right) = 0$$

```
>plot2d("sin(1/x)",-5,5):
```



## Latihan

---

Bukalah buku Kalkulus. Cari dan pilih beberapa (paling sedikit 5 fungsi berbeda tipe/bentuk/jenis) fungsi dari buku tersebut, kemudian definisikan di EMT pada baris-baris perintah berikut (jika perlu tambahkan lagi). Untuk setiap fungsi, hitung nilai limit fungsi tersebut di beberapa nilai dan di tak hingga. Gambar grafik fungsi tersebut untuk mengkonfirmasi nilai-nilai limit tersebut.

Jawab:

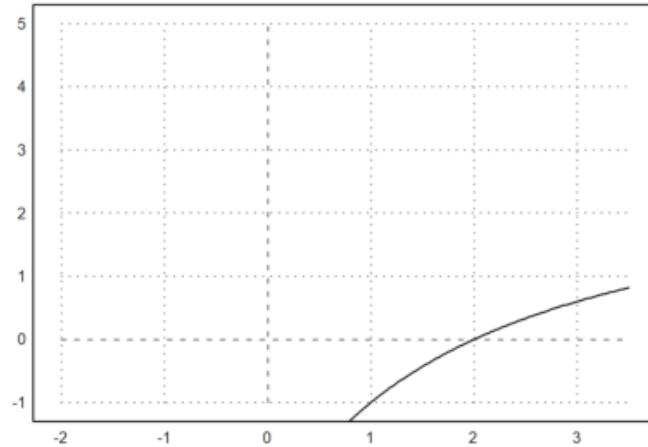
1. Fungsi 1

$$f(x) = \frac{3x - 6}{x + 2}$$

```
>$showev('limit((3*x-6)/(x+2),x,2))
```

$$\lim_{x \rightarrow 2} \frac{3x - 6}{x + 2} = 0$$

```
>plot2d("(3*x-6)/(x+2)",-2,3.5,-1,5):
```



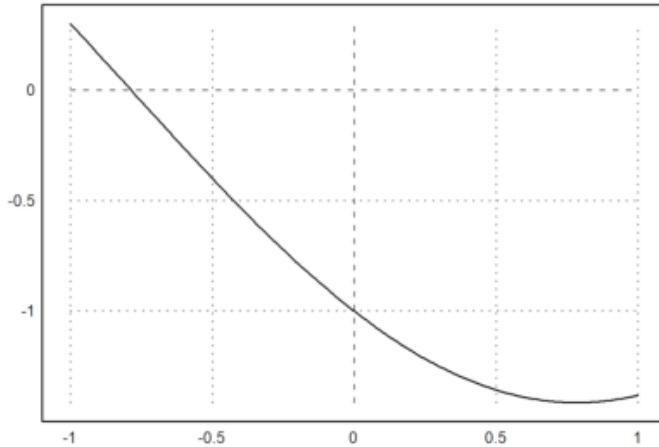
## 2. Fungsi 2

$$f(x) = \frac{\cos 2x}{\sin x - \cos x}$$

```
>$showev('limit(cos(2*x)/(sin(x) - cos (x)),x,0))
```

$$\lim_{x \rightarrow 0} \frac{\cos(2x)}{\sin x - \cos x} = -1$$

```
>plot2d("cos(2*x)/(sin(x) - cos (x))",-1,1):
```



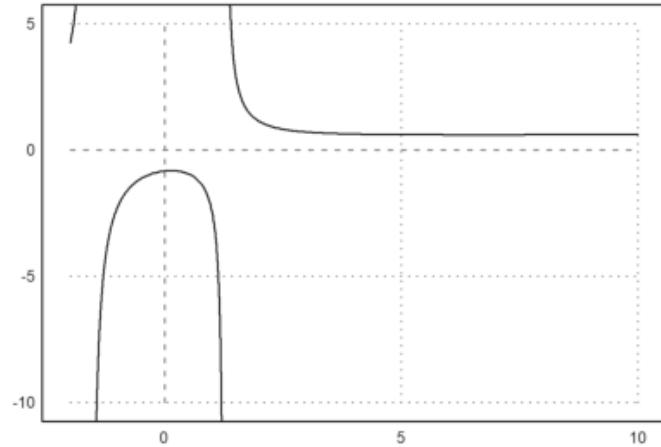
### 3. Fungsi 3

$$f(x) = \frac{2x^2 - 2x + 5}{3x^2 + x - 6}$$

```
>showev('limit(((2*x^2-2*x+5)/(3*x^2+x-6)),x,3))
```

$$\lim_{x \rightarrow 3} \frac{2x^2 - 2x + 5}{3x^2 + x - 6} = \frac{17}{24}$$

```
>plot2d("(2*x^2-2*x+5)/(3*x^2+x-6)", -2, 10, -10, 5):
```



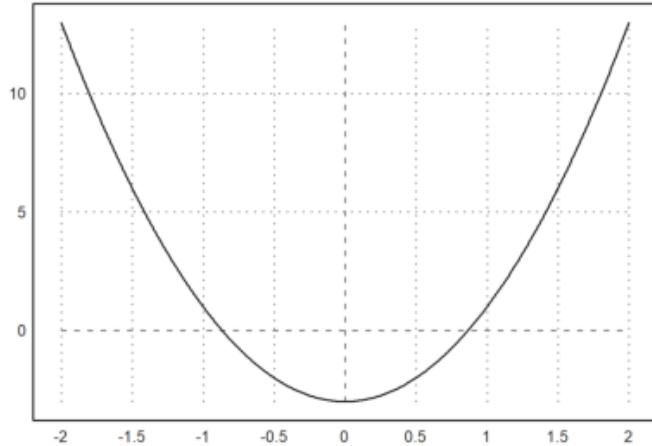
#### 4. Fungsi 4

$$f(x) = 4x^2 - 3$$

```
>$showev('limit((4*x^2-3),x,0))
```

$$\lim_{x \rightarrow 0} 4x^2 - 3 = -3$$

```
>plot2d("(4*x^2-3)":
```



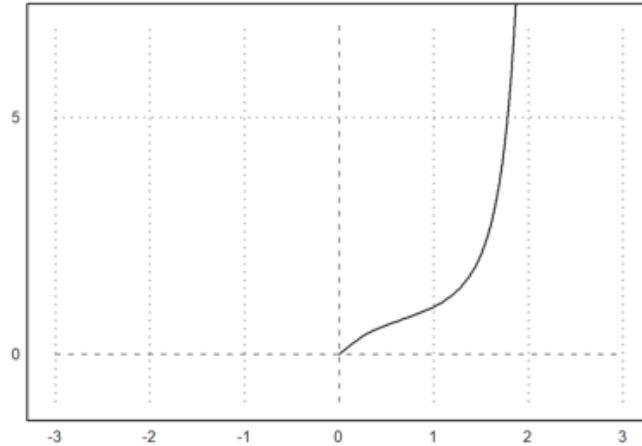
## 5. Fungsi 5

$$f(x) = x^{x^x}$$

```
>$showev('limit((x^(x^x))),x,0,plus))
```

$$\lim_{x \downarrow 0} x^{x^x} = 0$$

```
>plot2d("(x^(x^x))", -3, 3, -1, 7):
```



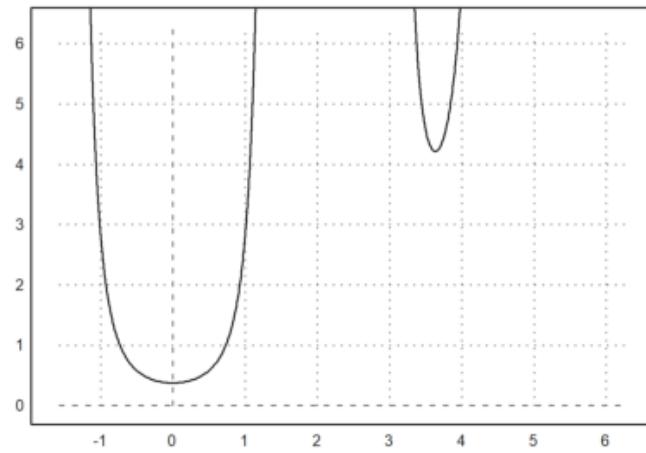
## 6. Fungsi 6

$$f(x) = \frac{3x \tan x}{1 - \cos 4x}$$

```
>$showev('limit((3*x*tan(x))/(1-cos(4*x)),x,0))
```

$$3 \left( \lim_{x \rightarrow 0} \frac{x \tan x}{1 - \cos(4x)} \right) = \frac{3}{8}$$

```
>plot2d("(3*x*tan(x))/(1-cos(4*x))",-pi/2,2pi,0,2pi):
```



## Turunan Fungsi

---

Definisi turunan:

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

Berikut adalah contoh-contoh menentukan turunan fungsi dengan menggunakan definisi turunan (limit).

```
>$showev('limit(((x+h)^n-x^n)/h,h,0)) // turunan x^n
```

$$\lim_{h \rightarrow 0} \frac{(x+h)^n - x^n}{h} = n x^{n-1}$$

Mengapa hasilnya seperti itu? Tuliskan atau tunjukkan bahwa hasil limit tersebut benar, sehingga benar turunan fungsinya benar. Tulis penjelasan Anda di komentar ini.

Sebagai petunjuk, ekspansikan  $(x+h)^n$  dengan menggunakan teorema binomial.

Jawab:

Akan ditunjukkan bahwa  $f'(x) = \lim_{h \rightarrow 0} \frac{(x+h)^n - x^n}{h} = n x^{n-1}$

Pertama, ekspansikan  $(x + h)^n$ , yakni:

$$(x+h)^n = \sum_{k=0}^n \binom{n}{k} x^{n-k} h^k$$

$$\Leftrightarrow (x+h)^n = \binom{n}{0} x^n + \binom{n}{1} x^{n-1} h + \binom{n}{2} x^{n-2} h^2 + \dots + \binom{n}{n} h^n$$

$$\Leftrightarrow (x+h)^n = x^n + nx^{n-1}h + \binom{n}{2} x^{n-2} h^2 + \binom{n}{3} x^{n-3} h^3 + \dots + h^n$$

Sehingga,  $f'(x)$  menjadi:  $f'(x) = \lim_{h \rightarrow 0} \frac{(x+h)^n - x^n}{h}$

$$\Leftrightarrow f'(x) = \lim_{h \rightarrow 0} \frac{x^n + nx^{n-1}h + \binom{n}{2} x^{n-2} h^2 + \binom{n}{3} x^{n-3} h^3 + \dots + h^n - x^n}{h}$$

$$\Leftrightarrow f'(x) = \lim_{h \rightarrow 0} nx^{n-1} + \binom{n}{2} x^{n-2} h + \binom{n}{3} x^{n-3} h^2 + \dots + h^{n-1}$$

$$\Leftrightarrow f'(x) = nx^{n-1}. \text{ Terbukti.}$$

```
>$showev('limit((sin(x+h)-sin(x))/h,h,0)) // turunan sin(x)
```

$$\lim_{h \rightarrow 0} \frac{\sin(x+h) - \sin x}{h} = \cos x$$

Mengapa hasilnya seperti itu? Tuliskan atau tunjukkan bahwa hasil limit tersebut benar, sehingga benar turunan fungsinya benar. Tulis penjelasan Anda di komentar ini.

Sebagai petunjuk, ekspansikan  $\sin(x+h)$  dengan menggunakan rumus jumlah dua sudut.  
Jawab:

Akan ditunjukkan bahwa  $\lim_{h \rightarrow 0} \frac{\sin(x+h) - \sin x}{h} = \cos x$

Diketahui bahwa:

1).  $\sin(x+h) = \sin x \cos h + \cos x \sin h$

2).  $\lim_{h \rightarrow 0} \frac{1 - \cos h}{h} = 0$

3).  $\lim_{h \rightarrow 0} \frac{\sin h}{h} = 1$

$$\lim_{h \rightarrow 0} \frac{\sin(x+h) - \sin x}{h}$$

$$= \lim_{h \rightarrow 0} \frac{\sin x \cos h + \cos x \sin h - \sin x}{h}$$

$$= \lim_{h \rightarrow 0} \left[ -\sin x \cdot \frac{1 - \cos h}{h} + \cos x \cdot \frac{\sin h}{h} \right]$$

$$= (-\sin x) \left[ \lim_{h \rightarrow 0} \frac{1 - \cos h}{h} + (\cos x) \lim_{h \rightarrow 0} \frac{\sin h}{h} \right]$$

$$= (-\sin x)(0) + (\cos x)(1) = \cos x. \text{ Terbukti.}$$

```
>$showev('limit((log(x+h)-log(x))/h,h,0)) // turunan log(x)
```

$$\lim_{h \rightarrow 0} \frac{\log(x+h) - \log x}{h} = \frac{1}{x}$$

Mengapa hasilnya seperti itu? Tuliskan atau tunjukkan bahwa hasil limit tersebut benar, sehingga benar turunan fungsinya benar. Tulis penjelasan Anda di komentar ini.

Sebagai petunjuk, gunakan sifat-sifat logaritma dan hasil limit pada bagian sebelumnya di atas.  
Jawab:

Bukti:

Ambil  $f(x) =^a \log x$ .

$$\lim_{h \rightarrow 0} \frac{^a \log(x+h) - ^a \log x}{h}$$

$$= \lim_{h \rightarrow 0} \frac{^a \log \frac{(x+h)}{x}}{h}$$

$$= \lim_{h \rightarrow 0} \frac{^a \log(1 + \frac{h}{x})}{h}$$

$$= \lim_{h \rightarrow 0} \frac{^a \log(1 + \frac{h}{x})}{\frac{h}{x}x}$$

$$= \lim_{h \rightarrow 0} \frac{\frac{x}{h} \cdot ^a \log(1 + \frac{h}{x})}{x}$$

$$= \lim_{h \rightarrow 0} \frac{^a \log(1 + \frac{h}{x})^{\frac{x}{h}}}{x}$$

$$= \frac{\lim_{h \rightarrow 0} ^a \log(1 + \frac{h}{x})^{\frac{x}{h}}}{\lim_{h \rightarrow 0} x}$$

$$= \frac{1}{x \cdot {}^e \log a}$$

$$= \frac{1}{x \cdot \ln a}$$

Menggunakan hasil di atas, maka:

$$\frac{d \ln x}{dx} = \frac{d^e \log x}{dx} = \frac{1}{x \cdot \ln e} = \frac{1}{x}. \text{ Terbukti.}$$

```
>$showev('limit((1/(x+h)-1/x)/h,h,0)) // turunan 1/x
```

$$\lim_{h \rightarrow 0} \frac{\frac{1}{x+h} - \frac{1}{x}}{h} = -\frac{1}{x^2}$$

```
>$showev('limit((E^(x+h)-E^x)/h,h,0))// turunan f(x)=e^x
```

```
Answering "Is x an integer?" with "integer"
Maxima is asking
Acceptable answers are: yes, y, Y, no, n, N, unknown, uk
Is x an integer?
```

```
Use assume!
Error in:
$showev('limit((E^(x+h)-E^x)/h,h,0))// turunan f(x)=e^x ...
```

Maxima bermasalah dengan limit:

$$\lim_{h \rightarrow 0} \frac{e^{x+h} - e^x}{h}.$$

Oleh karena itu diperlukan trik khusus agar hasilnya benar.

```
>$showev('limit((E^h-1)/h,h,0))
```

$$\lim_{h \rightarrow 0} \frac{e^h - 1}{h} = 1$$

```
>$factor(E^(x+h)-E^x)
```

$$(e^h - 1) e^x$$

```
>$showev('limit(factor((E^(x+h)-E^x)/h),h,0)) // turunan f(x)=e^x
```

$$\left( \lim_{h \rightarrow 0} \frac{e^h - 1}{h} \right) e^x = e^x$$

```
>function f(x) &= x^x
```

$$\begin{matrix} x \\ x \end{matrix}$$

```
>$showev('limit((f(x+h)-f(x))/h,h,0)) // turunan f(x)=x^x
```

$$\lim_{h \rightarrow 0} \frac{(x+h)^{x+h} - x^x}{h} = infinity$$

Di sini Maxima juga bermasalah terkait limit:

$$\lim_{h \rightarrow 0} \frac{(x+h)^{x+h} - x^x}{h}.$$

Dalam hal ini diperlukan asumsi nilai x.

```
>&assume(x>0); $showev('limit((f(x+h)-f(x))/h,h,0)) // turunan f(x)=x^x
```

$$\lim_{h \rightarrow 0} \frac{(x+h)^{x+h} - x^x}{h} = x^x (\log x + 1)$$

```
>&forget(x>0) // jangan lupa, lupakan asumsi untuk kembali ke semula
```

[x > 0]

```
>&forget(x<0)
```

[x < 0]

```
>&facts()
```

```
[kind(sinh, one_to_one), kind(log, one_to_one),
 kind(tanh, one_to_one), kind(log, increasing)]
```

```
>$showev('limit((asin(x+h)-asin(x))/h,h,0)) // turunan arcsin(x)
```

$$\lim_{h \rightarrow 0} \frac{\arcsin(x + h) - \arcsin x}{h} = \frac{1}{\sqrt{1 - x^2}}$$

```
>$showev('limit((tan(x+h)-tan(x))/h,h,0)) // turunan tan(x)
```

$$\lim_{h \rightarrow 0} \frac{\tan(x + h) - \tan x}{h} = \frac{1}{\cos^2 x}$$

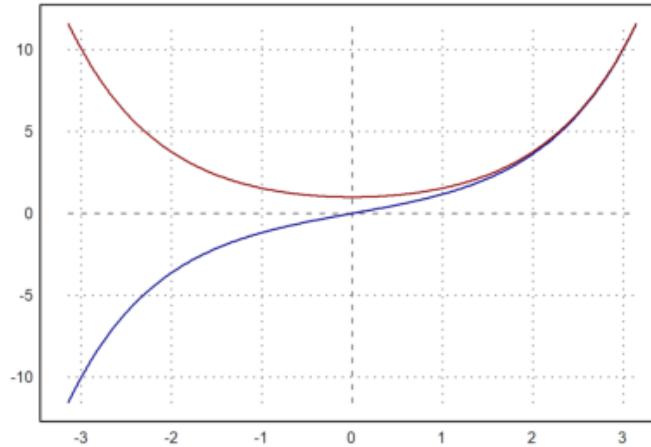
```
>function f(x) &= sinh(x) // definisikan f(x)=sinh(x)
```

$\text{sinh}(x)$

```
>function df(x) &= limit((f(x+h)-f(x))/h,h,0); $df(x) // df(x) = f'(x)
```

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

```
>plot2d(["f(x)","df(x)] ,-pi,pi,color=[blue,red]):
```



```
>function f(x) &= sin(3*x^5+7)^2
```

$$\sin^2(3x^5 + 7)$$

```
>diff(f,3), diffc(f,3)
```

1198.32948904

1198.72863721

diff(f,3) : Ini biasanya menunjukkan F mengenai X dievaluasi X=3.  
Metode diff memberikan turunan yang tepat.

diffc(f,3) : Ini biasanya mengacu pada X=3.  
Metode diffc biasanya memperkirakan.

```
>$showev('diff(f(x),x))
```

$$\frac{d}{dx} \sin^2(3x^5 + 7) = 30x^4 \cos(3x^5 + 7) \sin(3x^5 + 7)$$

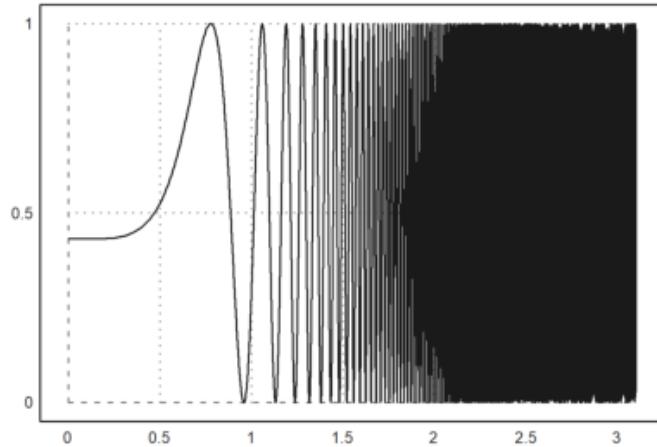
```
>$% with x=3
```

$$\%at\left(\frac{d}{dx} \sin^2(3x^5 + 7), x = 3\right) = 2430 \cos 736 \sin 736$$

```
>$float(%)
```

$$\%at\left(\frac{d^{1.0}}{dx^{1.0}} \sin^2(3.0x^5 + 7.0), x = 3.0\right) = 1198.728637211748$$

```
>plot2d(f,0,3.1):
```



```
>function f(x) &=5*cos(2*x)-2*x*sin(2*x) // mendefinisikan fungsi f
```

$$5 \cos(2x) - 2x \sin(2x)$$

```
>function df(x) &=diff(f(x),x) // fd(x) = f'(x)
```

$$- 12 \sin(2x) - 4x \cos(2x)$$

```
>${'f(1)=f(1), $float(f(1)), '$f(2)=f(2), $float(f(2)) // nilai f(1) dan f(2)}
```

-0.2410081230863468

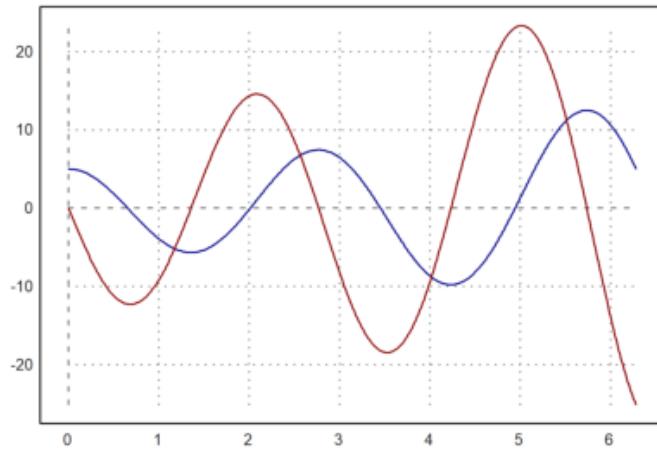
```
>xp=solve("df(x)",1,2,0) // solusi  $f'(x)=0$  pada interval [1, 2]
```

1.35822987384

```
>df(xp), f(xp) // cek bahwa  $f'(xp)=0$  dan nilai ekstrim di titik tersebut
```

0  
-5.67530133759

```
>plot2d(["f(x)","df(x)"],0,2*pi,color=[blue,red]): //grafik fungsi dan turunannya
```



## Latihan

---

Bukalah buku Kalkulus. Cari dan pilih beberapa (paling sedikit 5 fungsi berbeda tipe/bentuk/jenis) fungsi dari buku tersebut, kemudian definisikan di EMT pada baris-baris perintah berikut (jika perlu tambahkan lagi). Untuk setiap fungsi, tentukan turunannya dengan menggunakan definisi turunan (limit), seperti contoh-contoh tersebut. Gambar grafik fungsi asli dan fungsi turunannya pada sumbu koordinat yang sama.

Jawab:

1. Fungsi 1

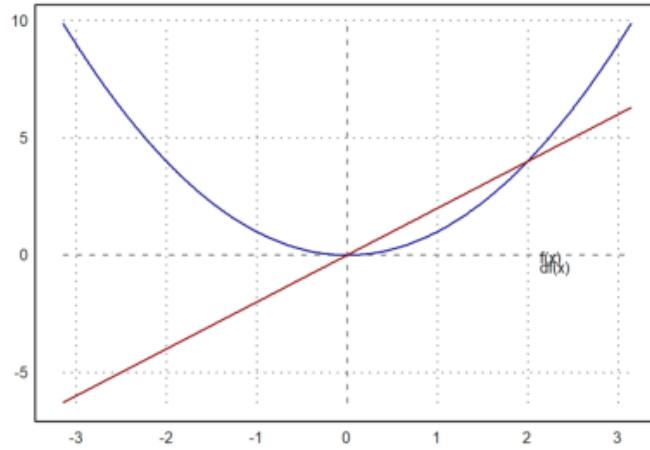
```
>function f(x) := x^2  
>$showev('limit(((x+h)^2-x^2)/h),h,0)) // turunan x^2
```

$$\lim_{h \rightarrow 0} \frac{(x+h)^2 - x^2}{h} = 2x$$

```
>function df(x) &= limit(((x+h)^2-x^2)/h),h,0); $df(x)// df(x) = f'(x)
```

$$2x$$

```
>plot2d(["f(x)","df(x)"],-pi,pi,color=[blue,red]), label("f(x)",2,0.6), label("df(x)",2,0.17):
```



## 2. Fungsi 2

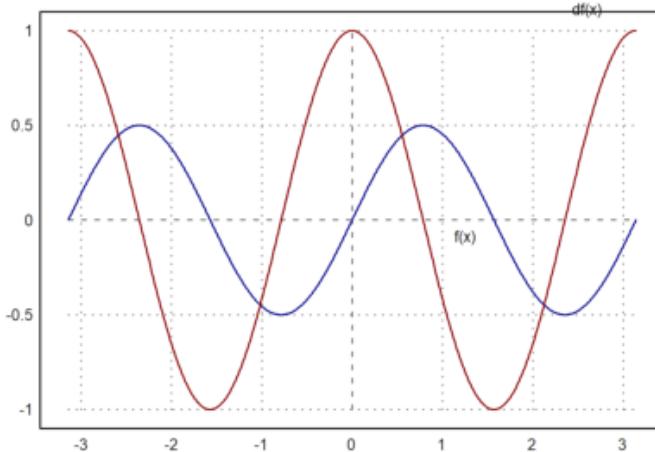
```
>function f(x) := sin(x)*cos(x)
>$showev('limit(((sin(x+h)*cos(x+h))-sin(x)*cos(x))/h,h,0)) // turunan sin(x)*cos(x)
```

$$\lim_{h \rightarrow 0} \frac{\cos(x+h) \sin(x+h) - \cos x \sin x}{h} = \cos^2 x - \sin^2 x$$

```
>function df(x) &= limit(((sin(x+h)*cos(x+h))-sin(x)*cos(x))/h,h,0); $df(x)// df(x) = f'(x)
```

$$\cos^2 x - \sin^2 x$$

```
>plot2d(["f(x)","df(x)"],-pi,pi,color=[blue,red]), label("f(x)",1,0), label("df(x)",2.3,1.2):
```



### 3. Fungsi 3

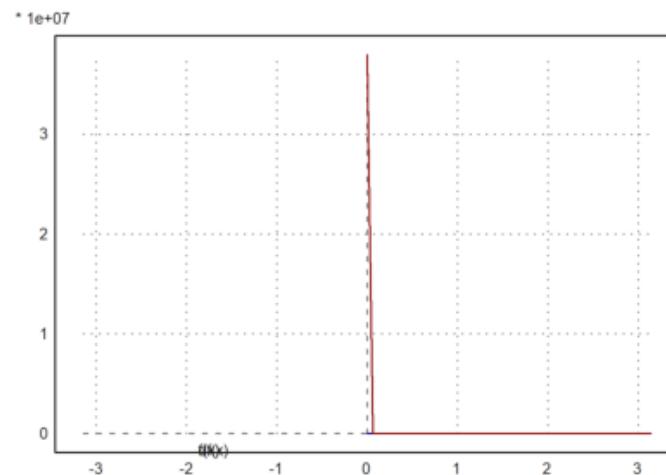
```
>function f(x) := sqrt(x)*4  
>$showev('limit((sqrt(x+h)*4-sqrt(x)*4)/h,h,0)) // turunan sqrt(x)*4
```

$$\lim_{h \rightarrow 0} \frac{4\sqrt{x+h} - 4\sqrt{x}}{h} = \frac{2}{\sqrt{x}}$$

```
>function df(x) &= limit((sqrt(x+h)*4-sqrt(x)*4)/h,h,0); $df(x)// df(x) = f'(x)
```

$$\frac{2}{\sqrt{x}}$$

```
>plot2d(["f(x)","df(x)"],-pi,pi,color=[blue,red]), label("f(x"),-2,11), label("df(x"),-2,-10):
```



#### 4. Fungsi 4

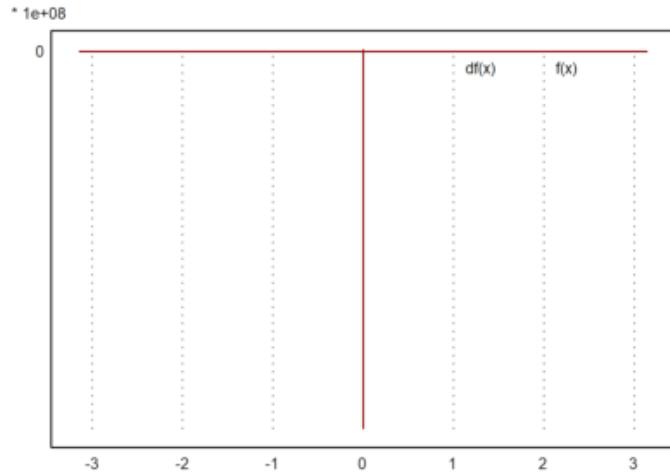
```
>function f(x) := cos(1/x)
>$showev('limit((cos(1/(x+h))-cos(1/x))/h,h,0)) // turunan cos(1/x)
```

$$\lim_{h \rightarrow 0} \frac{\cos\left(\frac{1}{x+h}\right) - \cos\left(\frac{1}{x}\right)}{h} = \frac{\sin\left(\frac{1}{x}\right)}{x^2}$$

```
>function df(x) &= limit((cos(1/(x+h))-cos(1/x))/h,h,0); $df(x)// df(x) = f'(x)
```

$$\frac{\sin\left(\frac{1}{x}\right)}{x^2}$$

```
>plot2d(["f(x)","df(x)"],-pi,pi,color=[blue,red]), label("f(x)",2,0.4), label("df(x)",1,-0.5):
```



## 5. Fungsi 5

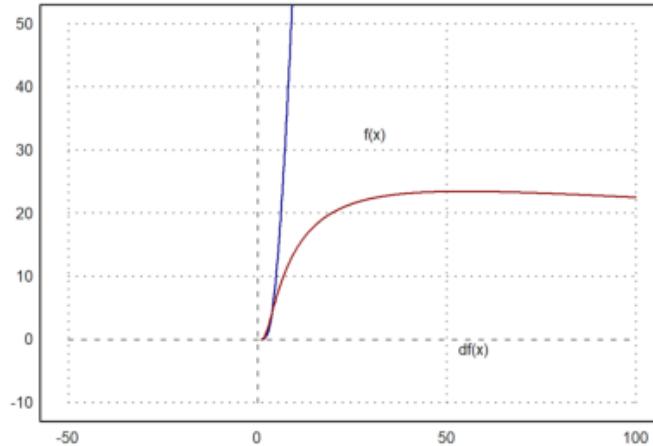
```
>function f(x) := (log(x))^5
>$showev('limit(((log(x+h))^5-(log(x))^5)/h,h,0)) // turunan (log(x))^5
```

$$\lim_{h \rightarrow 0} \frac{\log^5(x+h) - \log^5 x}{h} = \frac{5 \log^4 x}{x}$$

```
>function df(x) &= limit(((log(x+h))^5-(log(x))^5)/h,h,0); $df(x)// df(x) = f'(x)
```

$$\frac{5 \log^4 x}{x}$$

```
>plot2d(["f(x)","df(x)"],-50,100,-10,50,color=[blue,red]), label("f(x)",25,35), label("df(x)",50,1):
```



## 6. Fungsi 6

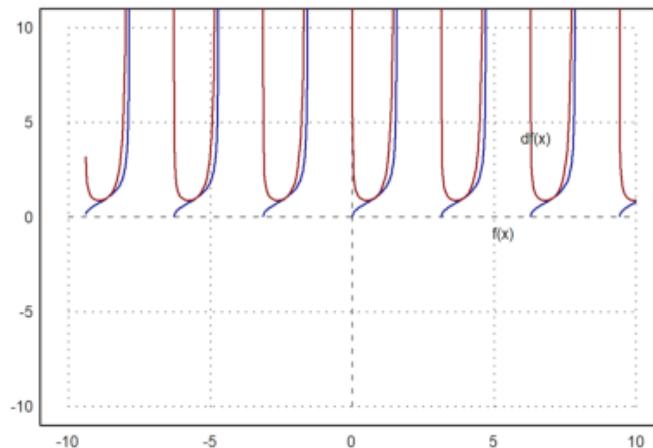
```
>function f(x) := sqrt(tan(x))
>$showev('limit((sqrt(tan(x+h))-sqrt(tan(x)))/h,h,0)) // turunan exp(x)*cos(x)
```

$$\lim_{h \rightarrow 0} \frac{\sqrt{\tan(x+h)} - \sqrt{\tan x}}{h} = \frac{1}{2 \cos^2 x \sqrt{\tan x}}$$

```
>function df(x) &= limit((sqrt(tan(x+h))-sqrt(tan(x)))/h,h,0); $df(x)// df(x) = f'(x)
```

$$\frac{1}{2 \cos^2 x \sqrt{\tan x}}$$

```
>plot2d(["f(x)","df(x)"],-10,10,-10,10,color=[blue,red]), label("f(x"),4.5,0), label("df(x"),5.5,5):
```



EMT dapat digunakan untuk menghitung integral, baik integral tak tentu maupun integral tentu. Untuk integral tak tentu (simbolik) sudah tentu EMT menggunakan Maxima, sedangkan untuk perhitungan integral tentu EMT sudah menyediakan beberapa fungsi yang mengimplementasikan algoritma kuadratur (perhitungan integral tentu menggunakan metode numerik).

Pada notebook ini akan ditunjukkan perhitungan integral tentu dengan menggunakan Teorema Dasar Kalkulus:

$$\int_a^b f(x) \, dx = F(b) - F(a), \quad \text{dengan } F'(x) = f(x).$$

Fungsi untuk menentukan integral adalah `integrate`. Fungsi ini dapat digunakan untuk menentukan, baik integral tentu maupun tak tentu (jika fungsinya memiliki antiderivatif). Untuk perhitungan integral tentu fungsi `integrate` menggunakan metode numerik (kecuali fungsinya tidak integrabel, kita tidak akan menggunakan metode ini).

```
>$showev('integrate(x^n,x))
```

Answering "Is n equal to -1?" with "no"

$$\int x^n \, dx = \frac{x^{n+1}}{n+1}$$

```
>$showev('integrate(1/(1+x),x))
```

$$\int \frac{1}{x+1} dx = \log(x+1)$$

```
>$showev('integrate(1/(1+x^2),x))
```

$$\int \frac{1}{x^2 + 1} dx = \arctan x$$

```
>$showev('integrate(1/sqrt(1-x^2),x))
```

$$\int \frac{1}{\sqrt{1-x^2}} dx = \arcsin x$$

```
>$showev('integrate(sin(x),x,0,pi))
```

$$\int_0^\pi \sin x dx = 2$$

```
>$showev('integrate(sin(x),x,a,b))
```

$$\int_a^b \sin x dx = \cos a - \cos b$$

```
>$showev('integrate(x^n,x,a,b))
```

Answering "Is n positive, negative or zero?" with "positive"

$$\int_a^b x^n \, dx = \frac{b^{n+1}}{n+1} - \frac{a^{n+1}}{n+1}$$

```
>$showev('integrate(x^2*sqrt(2*x+1),x))
```

$$\int x^2 \sqrt{2x+1} \, dx = \frac{(2x+1)^{\frac{7}{2}}}{28} - \frac{(2x+1)^{\frac{5}{2}}}{10} + \frac{(2x+1)^{\frac{3}{2}}}{12}$$

```
>$showev('integrate(x^2*sqrt(2*x+1),x,0,2))
```

$$\int_0^2 x^2 \sqrt{2x+1} \, dx = \frac{25^{\frac{5}{2}}}{21} - \frac{2}{105}$$

```
>$ratsimp(%)
```

$$\int_0^2 x^2 \sqrt{2x+1} \, dx = \frac{25^{\frac{7}{2}} - 2}{105}$$

```
>$showev('integrate((sin(sqrt(x)+a)*E^sqrt(x))/sqrt(x),x,0,pi^2))
```

$$\int_0^{\pi^2} \frac{\sin(\sqrt{x} + a) e^{\sqrt{x}}}{\sqrt{x}} dx = (-e^\pi - 1) \sin a + (e^\pi + 1) \cos a$$

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

$$\int_0^{\pi^2} \frac{\sin(\sqrt{x} + a) e^{\sqrt{x}}}{\sqrt{x}} dx = (-e^\pi - 1) (\sin a - \cos a)$$

```
>function map f(x) &= E^(-x^2); $f(x)
```

$$e^{-x^2}$$

```
>$showev('integrate(f(x),x))
```

$$\int e^{-x^2} dx = \frac{\sqrt{\pi} \operatorname{erf}(x)}{2}$$

Fungsi  $f$  tidak memiliki antiturunan, integralnya masih memuat integral lain.

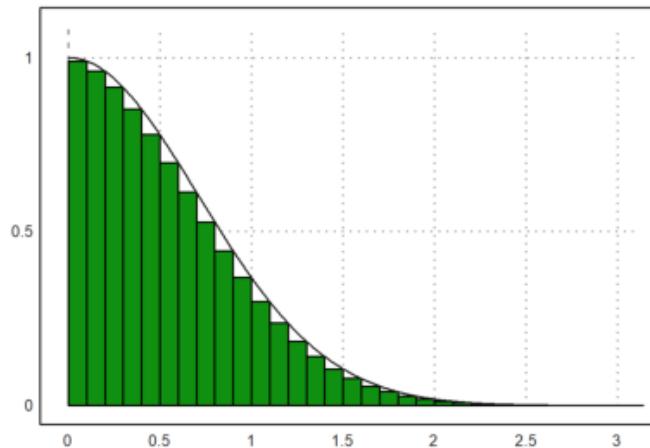
$$\operatorname{erf}(x) = \int \frac{e^{-x^2}}{\sqrt{\pi}} dx.$$

Kita tidak dapat menggunakan teorema Dasar kalkulus untuk menghitung integral tentu fungsi tersebut jika semua batasnya berhingga. Dalam hal ini dapat digunakan metode numerik (rumus kuadratur).

Misalkan kita akan menghitung:

$$\int_0^{\pi} e^{-x^2} dx$$

```
>x=0:0.1:pi-0.1; plot2d(x,f(x+0.1),>bar); plot2d("f(x)",0,pi,>add):
```



Integral tentu

$$\int_0^{\pi} e^{-x^2} dx$$

dapat dihampiri dengan jumlah luas persegi-persegi panjang di bawah kurva  $y=f(x)$  tersebut. Langkah-langkahnya adalah sebagai berikut.

```
>t &= makelist(a,a,0,pi-0.1,0.1); // t sebagai list untuk menyimpan nilai-nilai x  
>fx &= makelist(f(t[i]+0.1),i,1,length(t)); // simpan nilai-nilai f(x)  
>// jangan menggunakan x sebagai list, kecuali Anda pakar Maxima!
```

Hasilnya adalah:

$$\int_0^{\pi} e^{-x^2} dx = 0.8362196102528469$$

Jumlah tersebut diperoleh dari hasil kali lebar sub-subinterval ( $=0.1$ ) dan jumlah nilai-nilai  $f(x)$  untuk  $x = 0.1, 0.2, 0.3, \dots, 3.2$ .

```
>0.1*sum(f(x+0.1)) // cek langsung dengan perhitungan numerik EMT
```

0.836219610253

Untuk mendapatkan nilai integral tentu yang mendekati nilai sebenarnya, lebar sub-intervalnya dapat diperkecil lagi, sehingga daerah di bawah kurva tertutup semuanya, misalnya dapat digunakan lebar subinterval 0.001. (Silakan dicoba!)

Meskipun Maxima tidak dapat menghitung integral tentu fungsi tersebut untuk batas-batas yang berhingga, namun integral tersebut dapat dihitung secara eksak jika batas-batasnya tak hingga. Ini adalah salah satu keajaiban di dalam matematika, yang terbatas tidak dapat dihitung secara eksak, namun yang tak hingga malah dapat dihitung secara eksak.

```
>$showev('integrate(f(x),x,0,inf))
```

$$\int_0^{\infty} e^{-x^2} dx = \frac{\sqrt{\pi}}{2}$$

Berikut adalah contoh lain fungsi yang tidak memiliki antiderivatif, sehingga integral tentunya hanya dapat dihitung dengan metode numerik.

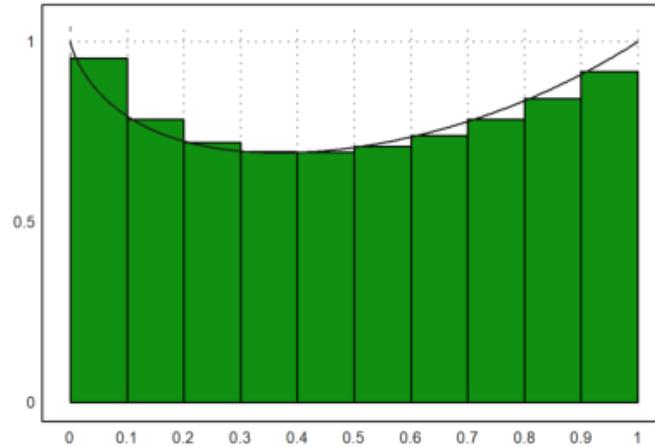
```
>function f(x) &= x^x; $f(x)
```

$$x^x$$

```
>$showev('integrate(f(x),x,0,1))
```

$$\int_0^1 x^x dx = \int_0^1 x^x dx$$

```
>x=0:0.1:1-0.01; plot2d(x,f(x+0.01),>bar); plot2d("f(x)",0,1,>add):
```



Maxima gagal menghitung integral tentu tersebut secara langsung menggunakan perintah integrate. Berikut kita lakukan seperti contoh sebelumnya untuk mendapat hasil atau pendekatan nilai integral tentu tersebut.

```
>t &= makelist(a,a,0,1-0.01,0.01);
>fx &= makelist(f(t[i]+0.01),i,1,length(t));
```

$$\int_0^1 x^x \, dx = 0.7834935879025506$$

Apakah hasil tersebut cukup baik? perhatikan gambarnya.

```
>function f(x) &= sin(3*x^5+7)^2
```

$$\sin^2(3x^5 + 7)$$

```
>integrate(f,0,1)
```

0.542581176074

```
>&showev('integrate(f(x),x,0,1))
```

$$\begin{aligned}&\text{'integrate}(\sin^2(3x^5 + 7), x, 0, 1) = \\&\quad \text{'integrate}(\sin^2(3x^5 + 7), x, 0, 1)\end{aligned}$$

```
>&float(%)
```

```
          2      5  
'integrate(sin (3.0 x  + 7.0), x, 0.0, 1.0) =  
          2      5  
'integrate(sin (3.0 x  + 7.0), x, 0.0, 1.0)
```

```
>$showev('integrate(x*exp(-x),x,0,1)) // Integral tentu (eksak)
```

$$\int_0^1 x e^{-x} dx = 1 - 2 e^{-1}$$

**Latihan**

- 
- Bukalah buku Kalkulus.
  - Cari dan pilih beberapa (paling sedikit 5 fungsi berbeda tipe/bentuk/jenis) fungsi dari buku tersebut, kemudian definisikan di EMT pada baris-baris perintah berikut (jika perlu tambahkan lagi).
  - Untuk setiap fungsi, tentukan anti turunannya (jika ada), hitunglah integral tentu dengan batas-batas yang menarik (Anda tentukan sendiri), seperti contoh-contoh tersebut.
  - Lakukan hal yang sama untuk fungsi-fungsi yang tidak dapat diintegralkan (cari sedikitnya 3 fungsi).
  - Gambar grafik fungsi dan daerah integrasinya pada sumbu koordinat yang sama.
  - Gunakan integral tentu untuk mencari luas daerah yang dibatasi oleh dua kurva yang berpotongan di dua titik. (Cari dan gambar kedua kurva dan arsir (warnai) daerah yang dibatasi oleh keduanya.)
  - Gunakan integral tentu untuk menghitung volume benda putar kurva  $y = f(x)$  yang diputar mengelilingi sumbu x dari  $x=a$  sampai  $x=b$ , yakni

$$V = \int_a^b \pi(f(x))^2 \, dx.$$

- (Pilih fungsinya dan gambar kurva dan benda putar yang dihasilkan. Anda dapat mencari contoh-contoh bagaimana cara menggambar benda hasil perputaran suatu kurva.)
- Gunakan integral tentu untuk menghitung panjang kurva  $y=f(x)$  dari  $x=a$  sampai  $x=b$  dengan menggunakan rumus:

$$S = \int_a^b \sqrt{1 + (f'(x))^2} \, dx.$$

(Pilih fungsi dan gambar kurvanya.)

Jawab:

1. Fungsi 1

```
>function f(x) &= 5*x^2; $f(x)
```

$$5x^2$$

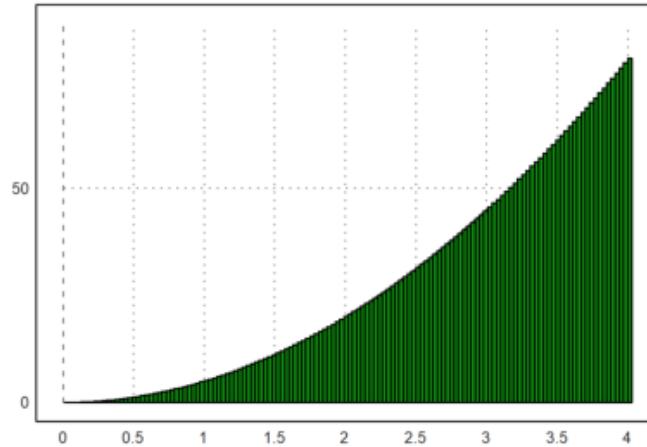
```
>$showev('integrate(f(x),x))
```

$$5 \int x^2 dx = \frac{5x^3}{3}$$

```
>$showev('integrate(f(x),x,2,3))
```

$$5 \int_2^3 x^2 dx = \frac{95}{3}$$

```
>x=0.01:0.03:4; plot2d(x,f(x+0.01),>bar); plot2d("f(x)",2,3,>add):
```



## 2. Fungsi 2

```
>function f(x) &= cos(2*x+5); $f(x)
```

$$\cos(2x + 5)$$

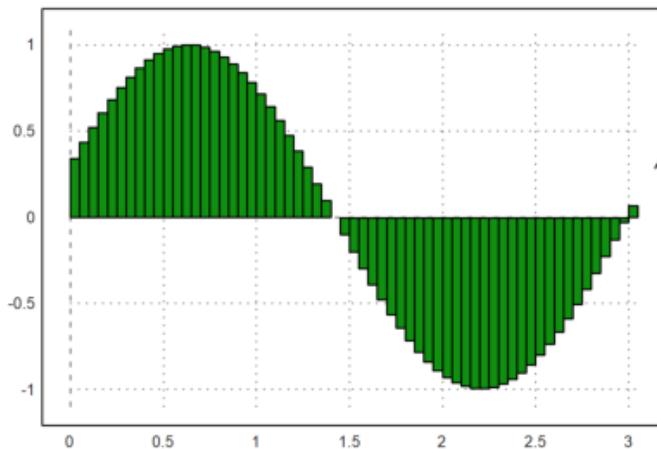
```
>$showev('integrate(f(x),x))
```

$$\int \cos(2x + 5) dx = \frac{\sin(2x + 5)}{2}$$

```
>$showev('integrate(f(x),x,pi,2*pi))
```

$$\int_{\pi}^{2\pi} \cos(2x + 5) dx = 0$$

```
>x=0:0.05:pi-0.1; plot2d(x,f(x+0.03),>bar); plot2d("f(x)",pi,2*pi,>add):
```



### 3. Fungsi 3

```
>function f(x) &= (sin(x))*(cos((x)))^2; $f(x)
```

$$\cos^2 x \sin x$$

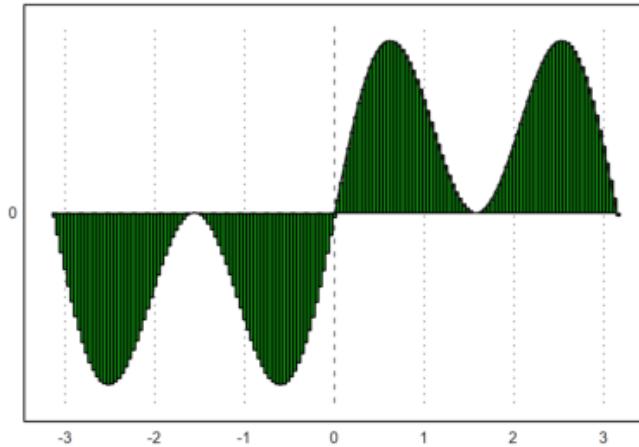
```
>$showev('integrate(f(x),x))
```

$$\int \cos^2 x \sin x \, dx = -\frac{\cos^3 x}{3}$$

```
>$showev('integrate(f(x),x,0,pi))
```

$$\int_0^\pi \cos^2 x \sin x \, dx = \frac{2}{3}$$

```
>x=-pi:0.04:pi; plot2d(x,f(x+0.01),>bar); plot2d("f(x)",0,pi,>add):
```



#### 4. Fungsi 4

```
>function f(x) &= (x^2*(2-x^3)^(1/2)); $f(x)
```

$$x^2 \sqrt{2 - x^3}$$

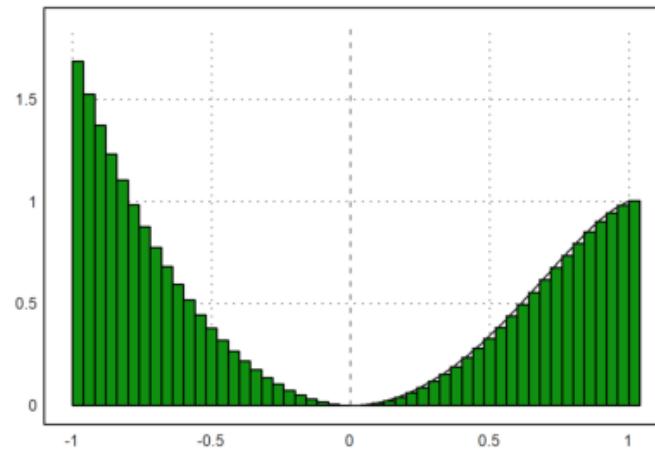
```
>$showev('integrate(f(x),x))
```

$$\int x^2 \sqrt{2 - x^3} dx = -\frac{2 (2 - x^3)^{\frac{3}{2}}}{9}$$

```
>$showev('integrate(f(x),x,0,1))
```

$$\int_0^1 x^2 \sqrt{2 - x^3} dx = \frac{2^{\frac{5}{2}}}{9} - \frac{2}{9}$$

```
>x=-1:0.04:1; plot2d(x,f(x+0.01),>bar); plot2d("f(x)",0,1,>add):
```



5. Fungsi 5

```
>function f(x) &= sqrt(24-x^2); $f(x)
```

$$\sqrt{24 - x^2}$$

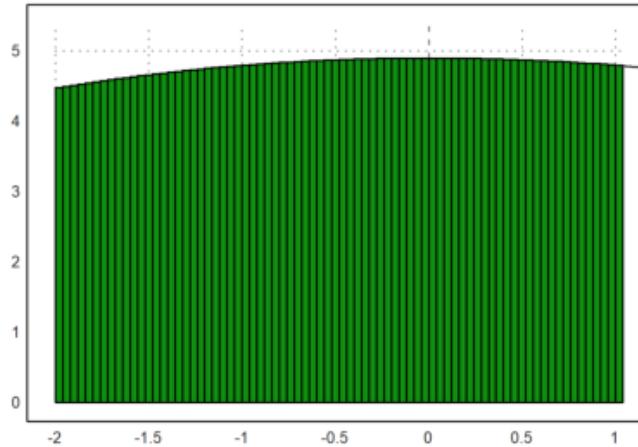
```
>$showev('integrate(f(x),x))
```

$$\int \sqrt{24 - x^2} dx = 12 \arcsin\left(\frac{x}{2\sqrt{6}}\right) + \frac{x\sqrt{24 - x^2}}{2}$$

```
>$showev('integrate(f(x),x,1,2))
```

$$\int_1^2 \sqrt{24 - x^2} dx = 12 \arcsin\left(\frac{1}{\sqrt{6}}\right) - \frac{24 \arcsin\left(\frac{1}{2\sqrt{6}}\right) + \sqrt{23}}{2} + 2\sqrt{5}$$

```
>x=-2:0.04:1; plot2d(x,f(x+0.01),>bar); plot2d("f(x)",1,2,>add):
```

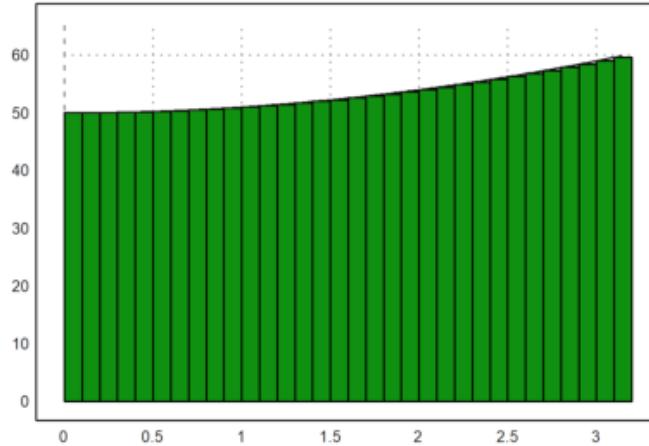


## 6. Fungsi 6

```
>t &= makelist(a,a,0,1-0.01,0.01);
>fx &= makelist(f(t[i]+0.01),i,1,length(t));
>function f(x) &= x^2+50; $f(x)
```

$$x^2 + 50$$

```
>x=0:0.1:pi-0.01; plot2d(x,f(x+0.01),>bar); plot2d("f(x)",0,pi,>add):
```



```
>0.01*sum(f(x+0.01))
```

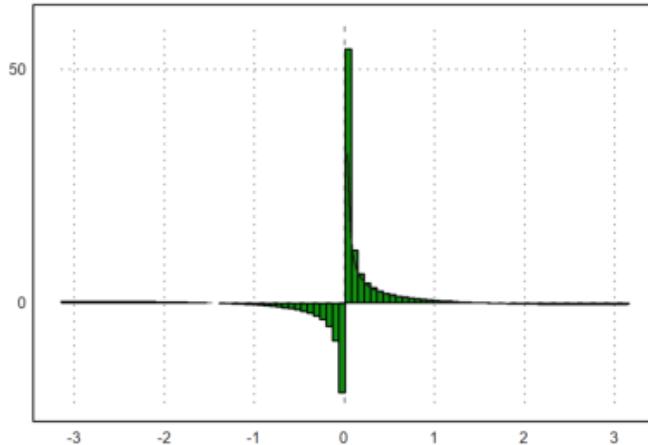
17.051552

## 7. Fungsi 7

```
>t &= makelist(a,a,0,1-0.01,0.01);
>fx &= makelist(f(t[i]+0.01),i,1,length(t));
>function f(x) &= cos(x)/x; $f(x)
```

$$\frac{\cos x}{x}$$

```
>x=-pi:0.07:pi-0.01; plot2d(x,f(x+0.01),>bar); plot2d("f(x)",0,pi,>add):
```



```
>0.01*sum(f(x+0.01))
```

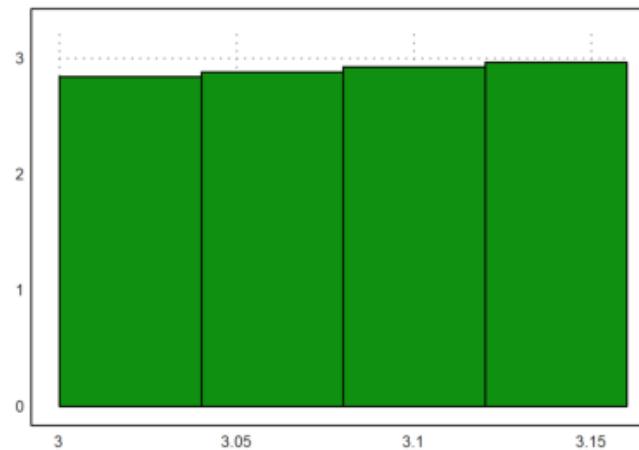
0.415163991256

8. Fungsi 8

```
>t &= makelist(a,a,0,1-0.01,0.01);
>fx &= makelist(f(t[i]+0.01),i,1,length(t));
>function f(x) &= sqrt(x^2-1); $f(x)
```

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

```
>x=3:0.04:pi-0.01; plot2d(x,f(x+0.01),>bar); plot2d("f(x)",0,2,>add):
```



```
>0.01*sum(f(x+0.01))
```

0.11610107668

---

## Luas daerah dibatasi 2 kurva

1). Fungsi 1

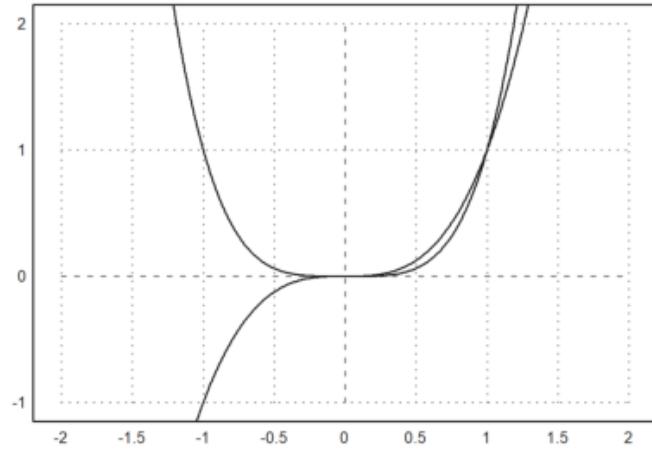
```
>function f(x) &= x^3; $f(x)
```

$$x^3$$

```
>function g(x) &= x; $g(x)
```

$$x$$

```
>plot2d(["x^4","x^3"],-2,2,-1,2):
```



```
>function h(x) &= f(x)-g(x); $h(x)
```

$$x^3 - x$$

```
>$showev('integrate(h(x),x))
```

$$\int x^3 - x \, dx = \frac{x^4}{4} - \frac{x^2}{2}$$

```
>$&solve(f(x)=g(x))
```

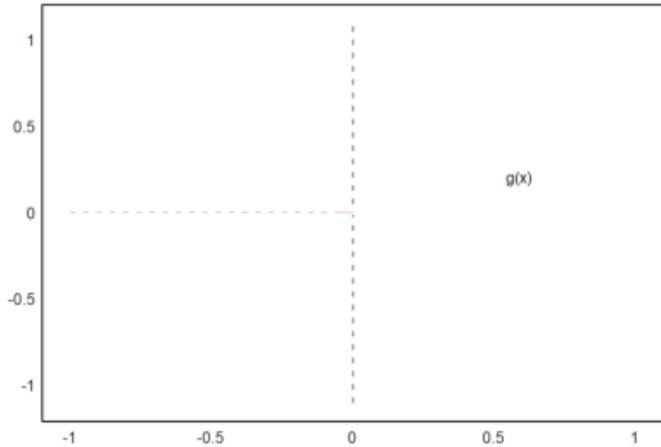
$$[x = -1, x = 1, x = 0]$$

```
>$showev('integrate(h(x),x,0,1)) // menghitung luas daerah yang dibatasi 2 kurva
```

$$\int_0^1 x^3 - x \, dx = -\frac{1}{4}$$

Arsiran daerah yang dibatasi kurva  $f(x)$  dan  $g(x)$  sebagai berikut:

```
>x=-1:0.01:1; plot2d(x,f(x),>bar,>filled,style="-",fillcolor=orange,>grid); plot2d(x,g(x),>bar,>add,
```



2). Fungsi 2

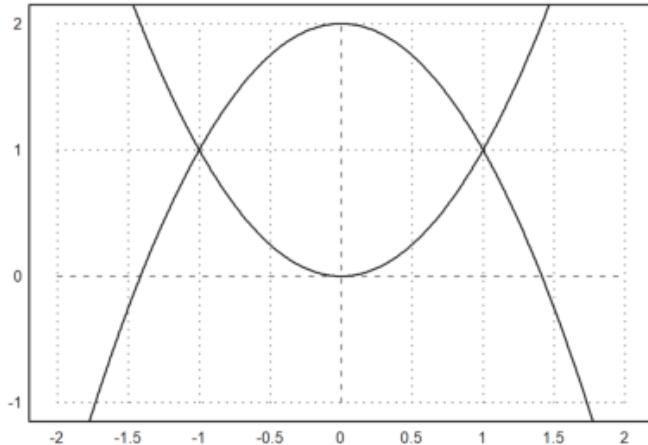
```
>function f(x) &= x^3+1; $f(x)
```

$$x^3 + 1$$

```
>function g(x) &= x^2; $g(x)
```

$$x^2$$

```
>plot2d(["-x^2+2","x^2"],-2,2,-1,2):
```



```
>function h(x) &= f(x)-g(x); $h(x)
```

$$x^3 - x^2 + 1$$

```
>$&solve(f(x)=g(x))
```

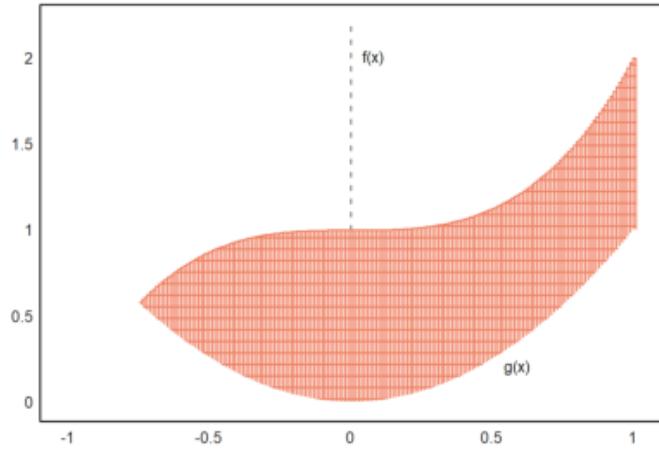
$$\left[ x = \frac{\frac{\sqrt{3}i}{2} - \frac{1}{2}}{9 \left( \frac{\sqrt{23}}{2 \cdot 3^{\frac{3}{2}}} - \frac{25}{54} \right)^{\frac{1}{3}}} + \left( \frac{\sqrt{23}}{2 \cdot 3^{\frac{3}{2}}} - \frac{25}{54} \right)^{\frac{1}{3}} \left( -\frac{\sqrt{3}i}{2} - \frac{1}{2} \right) + \frac{1}{3}, x = \left( \frac{\sqrt{23}}{2 \cdot 3^{\frac{3}{2}}} - \frac{25}{54} \right)^{\frac{1}{3}} \left( \frac{\sqrt{3}i}{2} - \frac{1}{2} \right) + \frac{-\frac{\sqrt{3}i}{2} - \frac{1}{2}}{9 \left( \frac{\sqrt{23}}{2 \cdot 3^{\frac{3}{2}}} - \frac{25}{54} \right)^{\frac{1}{3}}} + \frac{1}{3}, x \right]$$

```
>$showev('integrate(h(x),x,-1,1)) // menghitung luas daerah yang dibatasi 2 kurva
```

$$\int_{-1}^1 x^3 - x^2 + 1 \, dx = \frac{4}{3}$$

Arsiran daerah yang dibatasi kurva  $f(x)$  dan  $g(x)$  sebagai berikut:

```
>x=-1:0.01:1; plot2d(x,f(x),>bar,>filled,style="-",fillcolor=orange,>grid); plot2d(x,g(x),>bar,>add,
```



## Volume benda putar

---

Menghitung volume hasil perputaran kurva

$$m(x) = x^3 + 1$$

dari  $x=-1$  sampai  $x=0$ . Diputar terhadap sumbu-x.

Jawab:

```
>function m(x) &= x^4+3; $m(x)
```

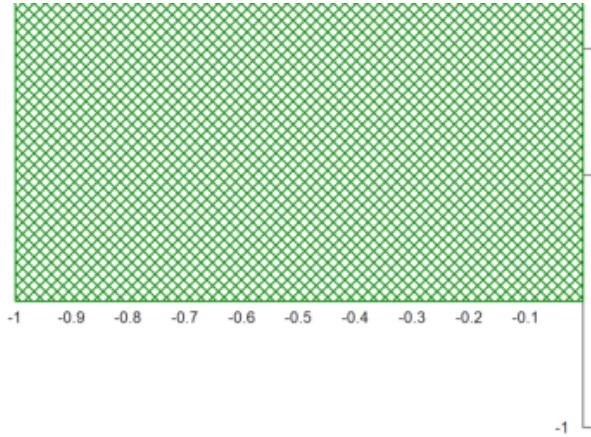
$$x^4 + 3$$

```
>$showev('integrate(pi*(m(x))^2,x,-1,0)) // Menghitung volume hasil perputaran m(x)
```

$$\pi \int_{-1}^0 (x^4 + 3)^2 dx = \frac{464\pi}{45}$$

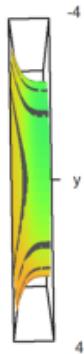
Daerah di bawah kurva yang akan dirotasi terhadap sumbu x sebagai berikut:

```
>plot2d("m(x)",-1,0,-1,2,grid=7,>filled, style="/\"):
```



Hasil perputaran  $m(x)$  terhadap sumbu x sebagai berikut:

```
>plot3d("m(x)",-1,0,-1,1,>rotate,angle=6.3,>hue,>contour,color=redgreen,height=11):
```



## Menghitung panjang kurva

Menghitung panjang kurva

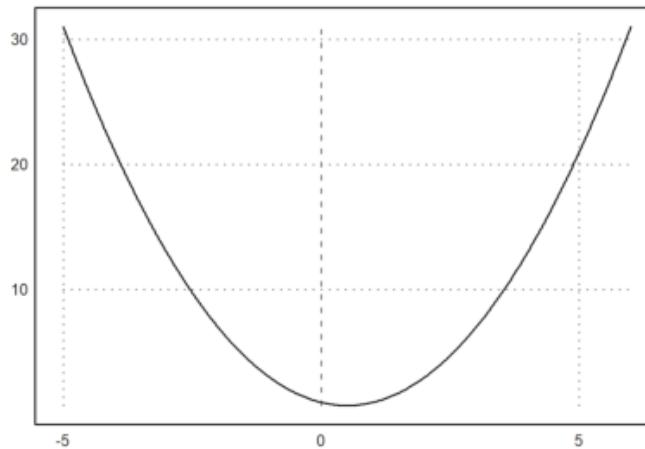
$$y = x^2 - x + 1$$

dari  $x=1$  sampai  $x=3$ .

```
>function d(x) &= x^2-x+1; $d(x)
```

$$x^2 - x + 1$$

```
>plot2d("d(x)",-5,6); // gambar kurva d(x)
```



```
>$showev('limit((d(x+h)-d(x))/h,h,0))
```

$$\lim_{h \rightarrow 0} \frac{(x+h)^2 - x^2 - h}{h} = 2x - 1$$

```
>function dd(x) &= limit((d(x+h)-d(x))/h,h,0); $dd(x)
```

$$2x - 1$$

```
>function q(x) &= ((dd(x))^2); $q(x)
```

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

```
>$showev('integrate(sqrt(1+q(x)),x,1,3)) // menghitung panjang kurva
```

$$\int_1^3 \sqrt{(2x - 1)^2 + 1} \, dx = \frac{\operatorname{asinh} 5 + 5\sqrt{26}}{4} - \frac{\operatorname{asinh} 1 + \sqrt{2}}{4}$$

Jadi, panjang kurva

$$y = x^2 - x + 1$$

dari x=0 sampai x=4 adalah

$$S = \frac{asinh5 + 5sqrt(26)}{4} - \frac{asinh(1) + sqrt(2)}{4}.$$

## Barisan dan Deret

---

(Catatan: bagian ini belum lengkap. Anda dapat membaca contoh-contoh penggunaan EMT dan Maxima untuk menghitung limit barisan, rumus jumlah parsial suatu deret, jumlah tak hingga suatu deret konvergen, dan sebagainya. Anda dapat mengeksplor contoh-contoh di EMT atau perbagai panduan penggunaan Maxima di software Maxima atau dari Internet.)

Barisan dapat didefinisikan dengan beberapa cara di dalam EMT, di antaranya:

- dengan cara yang sama seperti mendefinisikan vektor dengan elemen-elemen beraturan (menggunakan titik dua ":");
- menggunakan perintah "sequence" dan rumus barisan (suku ke -n);
- menggunakan perintah "iterate" atau "niterate";
- menggunakan fungsi Maxima "create\_list" atau "makelist" untuk menghasilkan barisan simbolik;
- menggunakan fungsi biasa yang inputnya vektor atau barisan;
- menggunakan fungsi rekursif.

EMT menyediakan beberapa perintah (fungsi) terkait barisan, yakni:

- sum: menghitung jumlah semua elemen suatu barisan
- cumsum: jumlah kumulatif suatu barisan
- differences: selisih antar elemen-elemen berturutan

EMT juga dapat digunakan untuk menghitung jumlah deret berhingga maupun deret tak hingga, dengan menggunakan perintah (fungsi) "sum". Perhitungan dapat dilakukan secara numerik maupun simbolik dan eksak.

Berikut adalah beberapa contoh perhitungan barisan dan deret menggunakan EMT.

```
>1:10 // barisan sederhana
```

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

```
>1:2:30
```

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29]
```

```
>sum(1:2:30), sum(1/(1:2:30))
```

```
225  
2.33587263431
```

```
>$'sum(k, k, 1, n) = factor(ev(sum(k, k, 1, n),simpsum=true)) // simpsum:menghitung deret secara sim
```

$$\sum_{k=1}^n k = \frac{n(n+1)}{2}$$

```
>$'sum(1/(3^k+k), k, 0, inf) = factor(ev(sum(1/(3^k+k), k, 0, inf),simpsum=true))
```

$$\sum_{k=0}^{\infty} \frac{1}{3^k + k} = \sum_{k=0}^{\infty} \frac{1}{3^k + k}$$

Di sini masih gagal, hasilnya tidak dihitung.

```
>$'sum(1/x^2, x, 1, inf)= ev(sum(1/x^2, x, 1, inf),simpsum=true) // ev: menghitung nilai ekspresi
```

$$\sum_{x=1}^{\infty} \frac{1}{x^2} = \frac{\pi^2}{6}$$

```
>$'sum((-1)^(k-1)/k, k, 1, inf) = factor(ev(sum((-1)^(x-1)/x, x, 1, inf),simpsum=true))
```

$$\sum_{k=1}^{\infty} \frac{(-1)^{k-1}}{k} = - \sum_{x=1}^{\infty} \frac{(-1)^x}{x}$$

Di sini masih gagal, hasilnya tidak dihitung.

```
>$'sum((-1)^k/(2*k-1), k, 1, inf) = factor(ev(sum((-1)^k/(2*k-1), k, 1, inf),simpsum=true))
```

$$\sum_{k=1}^{\infty} \frac{(-1)^k}{2k-1} = \sum_{k=1}^{\infty} \frac{(-1)^k}{2k-1}$$

```
>$ev(sum(1/n!, n, 0, inf),simpsum=true)
```

$$\sum_{n=0}^{\infty} \frac{1}{n!}$$

Di sini masih gagal, hasilnya tidak dihitung, harusnya hasilnya e.

```
>&assume(abs(x)<1); \$'sum(a*x^k, k, 0, inf)=ev(sum(a*x^k, k, 0, inf),simpsum=true), &forget(abs(x)<1)
```

$$a \sum_{k=0}^{\infty} x^k = \frac{a}{1-x}$$

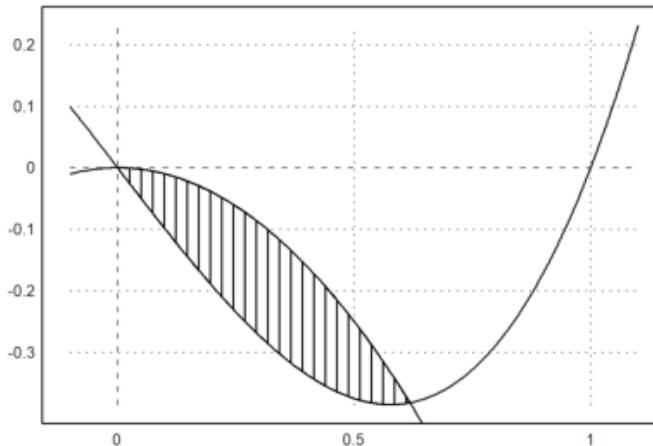
Deret geometri tak hingga, dengan asumsi rasional antara -1 dan 1.

## Aplikasi Integral

## Tentu

---

```
>plot2d("x^3-x",-0.1,1.1); plot2d("-x^2",>add); ...
>b=solve("x^3-x+x^2",0.5); x=linspace(0,b,200); xi=flipx(x); ...
>plot2d(x|xi,x^3-x|-xi^2,>filled,style="|",fillcolor=1,>add); // Plot daerah antara 2 kurva
```



```
>a=solve("x^3-x+x^2",0), b=solve("x^3-x+x^2",1) // absis titik-titik potong kedua kurva
```

0  
0.61803398875

```
>integrate("(-x^2)-(x^3-x)",a,b) // luas daerah yang diarsir
```

0.0758191713542

Hasil tersebut akan kita bandngkan dengan perhitungan secara analitik.

```
>a &= solve((-x^2)-(x^3-x),x); $a // menentukan absis titik potong kedua kurva secara eksak
```

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

```
>$showev('integrate(-x^2-x^3+x,x,0,(sqrt(5)-1)/2)) // Nilai integral secara eksak
```

$$\int_0^{\frac{\sqrt{5}-1}{2}} -x^3 - x^2 + x \, dx = \frac{13 - 5^{\frac{3}{2}}}{24}$$

```
>$float(%)
```

$$\int_{0.0}^{0.6180339887498949} -1.0 x^3 - 1.0 x^2 + x \, dx = 0.07581917135421037$$

Panjang Kurva

---

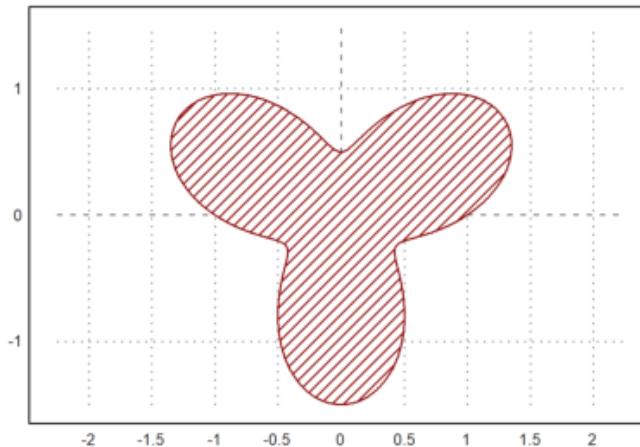
Hitunglah panjang kurva berikut ini dan luas daerah di dalam kurva tersebut.

$$\gamma(t) = (r(t) \cos(t), r(t) \sin(t))$$

dengan

$$r(t) = 1 + \frac{\sin(3t)}{2}, \quad 0 \leq t \leq 2\pi.$$

```
>t=linspace(0,2pi,1000); r=1+sin(3*t)/2; x=r*cos(t); y=r*sin(t); ...
>plot2d(x,y,>filled,fillcolor=red,style="/" ,r=1.5); // Kita gambar kurvanya terlebih dahulu
```



```
>function r(t) &= 1+sin(3*t)/2; $'r(t)=r(t)
```

$r ([0, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17, 0.18, 0.19, 0.2, 0.21, 0.22])$

```
>function fx(t) &= r(t)*cos(t); $'fx(t)=fx(t)
```

$fx ([0, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17, 0.18, 0.19, 0.2, 0.21, 0.22])$

```
>function fy(t) &= r(t)*sin(t); $'fy(t)=fy(t)
```

$fy ([0, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17, 0.18, 0.19, 0.2, 0.21, 0.22])$

```
>function ds(t) &= trigreduce(radcan(sqrt(diff(fx(t),t)^2+diff(fy(t),t)^2))); $'ds(t)=ds(t)
```

Maxima said:

```
diff: second argument must be a variable; found errexp1
-- an error. To debug this try: debugmode(true);
```

Error in:

```
... e(radcan(sqrt(diff(fx(t),t)^2+diff(fy(t),t)^2))); $'ds(t)=ds(t ...
```

```
>$integrate(ds(x),x,0,2*pi) //panjang (keliling) kurva
```

$$\int_0^{2\pi} ds(x) \, dx$$

Maxima gagal melakukan perhitungan eksak integral tersebut.

Berikut kita hitung integralnya secara numerik dengan perintah EMT.

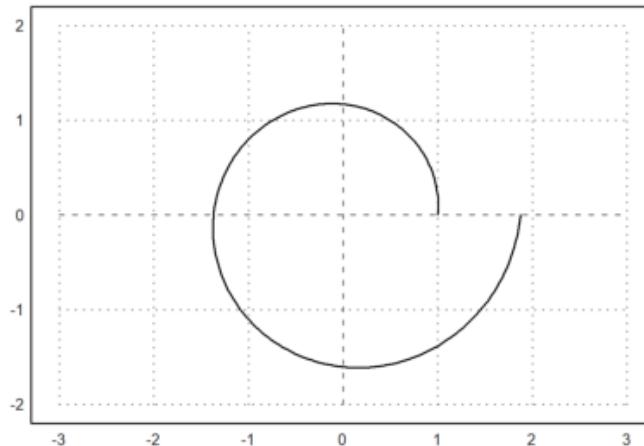
```
>integrate("ds(x)",0,2*pi)
```

```
Function ds not found.  
Try list ... to find functions!  
Error in expression: ds(x)  
%mapexpression1:  
    return expr(x,args());  
Error in map.  
%evalexpression:  
    if maps then return %mapexpression1(x,f$;args());  
gauss:  
    if maps then y=%evalexpression(f$,a+h-(h*xn)',maps;args());  
adaptivegauss:  
    t1=gauss(f$,c,c+h;args(),=maps);  
Try "trace errors" to inspect local variables after errors.  
integrate:  
    return adaptivegauss(f$,a,b,eps*1000;args(),=maps);
```

## Spiral Logaritmik

$$x = e^{ax} \cos x, \quad y = e^{ax} \sin x.$$

```
>a=0.1; plot2d("exp(a*x)*cos(x)","exp(a*x)*sin(x)",r=2,xmin=0,xmax=2*pi):
```



```
>&kill(a) // hapus expresi a
```

done

```
>function fx(t) &= exp(a*t)*cos(t); $'fx(t)=fx(t)
```

$fx([0, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17, 0.18, 0.19, 0.2, 0.21, 0.22])$

```
>function fy(t) &= exp(a*t)*sin(t); $'fy(t)=fy(t)
```

$fy([0, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17, 0.18, 0.19, 0.2, 0.21, 0.22])$

```
>function df(t) &= trigreduce(radcan(sqrt(diff(fx(t),t)^2+diff(fy(t),t)^2))); $'df(t)=df(t)
```

Maxima said:

```
diff: second argument must be a variable; found errexp1
-- an error. To debug this try: debugmode(true);
```

Error in:

```
... e(radcan(sqrt(diff(fx(t),t)^2+diff(fy(t),t)^2))); $'df(t)=df(t ...
```

```
>S &=integrate(df(t),t,0,2*pi); $S // panjang kurva (spiral)
```

Maxima said:

```
expt: undefined: 0 to a negative exponent.
```

```
#0: df(x=[0,0.01,0.02,0.03,0.04,0.05,0.06,0.07,0.08,0.09,0.1,0.11,0.12,0.13,0.14,0.15,0.16,0.17,0.
-- an error. To debug this try: debugmode(true);

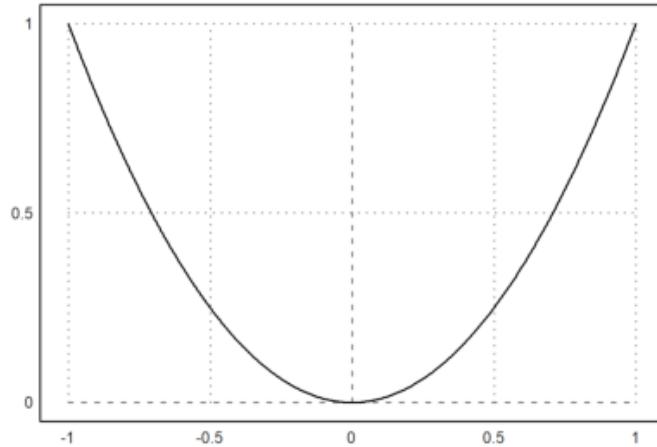
Error in:
S &=integrate(df(t),t,0,2*pi); $S // panjang kurva (spiral) ...
```

```
>S(a=0.1) // Panjang kurva untuk a=0.1
```

```
Function S not found.
Try list ... to find functions!
Error in:
S(a=0.1) // Panjang kurva untuk a=0.1 ...
```

Berikut adalah contoh menghitung panjang parabola

```
>plot2d("x^2",xmin=-1,xmax=1):
```



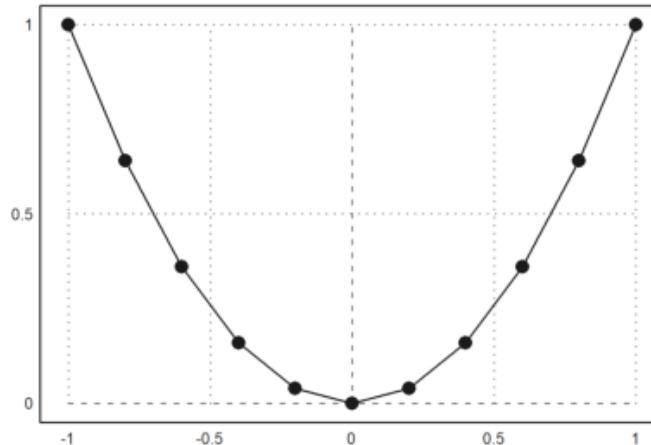
```
>$showev('integrate(sqrt(1+diff(x^2,x)^2),x,-1,1))
```

$$\int_{-1}^1 \sqrt{4x^2 + 1} dx = \frac{\operatorname{asinh} 2 + 2\sqrt{5}}{2}$$

```
>$float(%)
```

$$\int_{-1.0}^{1.0} \sqrt{4.0x^2 + 1.0} dx = 2.957885715089195$$

```
>x=-1:0.2:1; y=x^2; plot2d(x,y); ...
>plot2d(x,y,points=1,style="o#",add=1):
```



```
>i=1:cols(x)-1; sum(sqrt((x[i+1]-x[i])^2+(y[i+1]-y[i])^2))
```

2.95191957027

Hasilnya mendekati panjang yang dihitung secara eksak. Untuk mendapatkan hampiran yang cukup akurat, jarak antar titik dapat diperkecil, misalnya 0.1, 0.05, 0.01, dan seterusnya. Cobalah Anda ulangi perhitungannya dengan nilai-nilai tersebut.

## Koordinat Kartesius

---

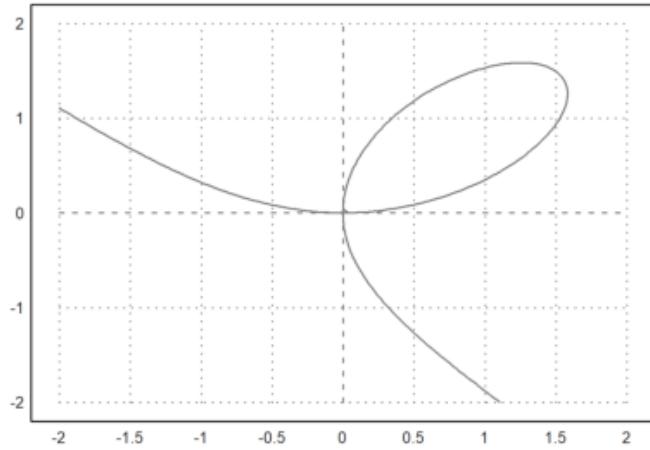
Berikut diberikan contoh perhitungan panjang kurva menggunakan koordinat Kartesius. Kita akan hitung panjang kurva dengan persamaan implisit:

$$x^3 + y^3 - 3xy = 0.$$

```
>z &= x^3+y^3-3*x*y; $z
```

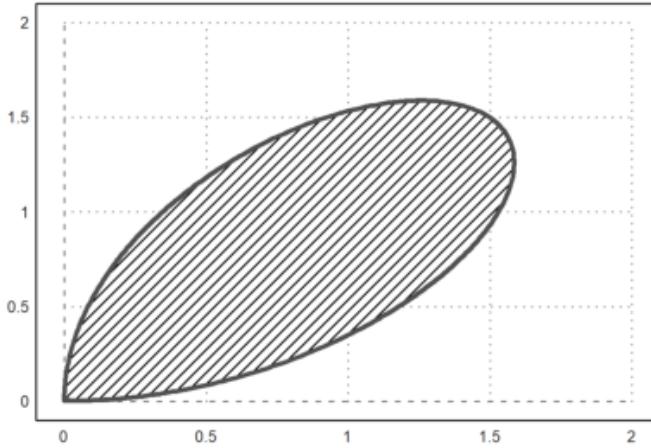
$$y^3 - 3 x y + x^3$$

```
>plot2d(z,r=2,level=0,n=100):
```



Kita tertarik pada kurva di kuadran pertama.

```
>plot2d(z,a=0,b=2,c=0,d=2,level=[-10;0],n=100,contourwidth=3,style="/"):
```



Kita selesaikan persamaannya untuk x.

```
>$z with y=l*x, sol &= solve(%,x); $sol
```

$$\left[ x = \frac{3l}{l^3 + 1}, x = 0 \right]$$

$$\left[ x = \frac{3l}{l^3 + 1}, x = 0 \right]$$

Kita gunakan solusi tersebut untuk mendefinisikan fungsi dengan Maxima.

```
>function f(l) &= rhs(sol[1]); $'f(l)=f(l)
```

$$f(l) = \frac{3l}{l^3 + 1}$$

Fungsi tersebut juga dapat digunakan untuk menggambar kurvanya. Ingat, bahwa fungsi tersebut adalah nilai x dan nilai y=l\*x, yakni x=f(l) dan y=l\*f(l).

```
>plot2d(&f(x),&x*f(x),xmin=-0.5,xmax=2,a=0,b=2,c=0,d=2,r=1.5):
```

```
Unexpected "(.". Index () not allowed in strict mode!
In Euler files, use relax to avoid this.
Error in expression: f(x)
adaptiveeval:
    sx=f$(t,args());
Try "trace errors" to inspect local variables after errors.
plot2d:
    dw/n,dw/n^2,dw/n,args());
```

Elemen panjang kurva adalah:

$$ds = \sqrt{f'(l)^2 + (lf'(l) + f(l))^2}.$$

```
>function ds(l) &= ratsimp(sqrt(diff(f(l),l)^2+diff(l*f(l),l)^2)); $'ds(l)=ds(l)
```

$$ds(l) = \sqrt{(l^2 + 1) \left( \frac{d}{dl} f(l) \right)^2 + 2 l f(l) \left( \frac{d}{dl} f(l) \right) + f^2(l)}$$

```
>$integrate(ds(l),l,0,1)
```

$$\int_0^1 \sqrt{(l^2 + 1) \left( \frac{d}{dl} f(l) \right)^2 + 2 l f(l) \left( \frac{d}{dl} f(l) \right) + f^2(l)} dl$$

Integral tersebut tidak dapat dihitung secara eksak menggunakan Maxima. Kita hitung integral tersebut secara numerik dengan Euler. Karena kurva simetris, kita hitung untuk nilai variabel integrasi dari 0 sampai 1, kemudian hasilnya dikalikan 2.

```
>2*integrate("ds(x)",0,1)
```

```
Syntax error in expression, or unfinished expression!
ds:
    useglobal; return sqrt((l^2+1)*('diff(f(l),l,1))^2+2*l*f(l)*' ...
Error in expression: ds(x)
```

```

%mapexpression1:
    return expr(x,args());
Error in map.
%evalexpression:
    if maps then return %mapexpression1(x,f$;args());
gauss:
    if maps then y=%evalexpression(f$,a+h-(h*xn)',maps;args());
adaptivegauss:
    t1=gauss(f$,c,c+h;args(),=maps);
Try "trace errors" to inspect local variables after errors.
integrate:
    return adaptivegauss(f$,a,b,eps*1000;args(),=maps);

```

```
>2*romberg(&ds(x),0,1)// perintah Euler lain untuk menghitung nilai hampiran integral yang sama
```

```

Syntax error in expression, or unfinished expression!
Error in expression: sqrt((x^2+1)*('diff(f(x),x,1))^2+2*x*f(x)*'diff(f(x),x,1)+f(x)^2)
%evalexpression:
    else return f$(x,args());
Try "trace errors" to inspect local variables after errors.
romberg:
y=%evalexpression(f$,linspace(a,b,m),maps;args());

```

Perhitungan di datas dapat dilakukan untuk sebarang fungsi x dan y dengan mendefinisikan fungsi EMT, misalnya kita beri nama panjangkurva. Fungsi ini selalu memanggil Maxima untuk menurunkan fungsi yang diberikan.

```
>function panjangkurva(fx,fy,a,b) ...
```

```
ds=mxm("sqrt(diff(@fx,x)^2+diff(@fy,x)^2)");
return romberg(ds,a,b);
endfunction
```

```
>panjangkurva("x","x^2",-1,1) // cek untuk menghitung panjang kurva sebelumnya
```

2.9579

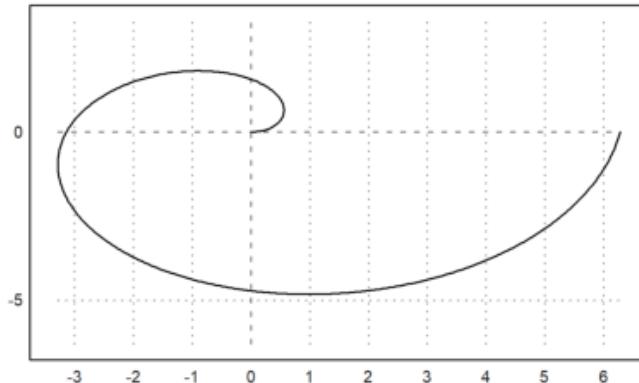
Bandingkan dengan nilai eksak di atas.

```
>2*panjangkurva(mxm("f(x)",mxm("x*f(x)",0,1)) // cek contoh terakhir, bandingkan hasilnya!
```

```
Syntax error in expression, or unfinished expression!
Error in expression: sqrt((x*diff(f(x),x,1)+f(x))^2+(diff(f(x),x,1))^2)
%evalexpression:
    if maps then return %mapexpression1(x,f$;args());
romberg:
    y=%evalexpression(f$,linspace(a,b,m),maps;args());
Try "trace errors" to inspect local variables after errors.
panjangkurva:
    return romberg(ds,a,b);
```

Kita hitung panjang spiral Archimedes berikut ini dengan fungsi tersebut.

```
>plot2d("x*cos(x)","x*sin(x)",xmin=0,xmax=2*pi,square=1):
```



```
>panjangkurva("x*cos(x)","x*sin(x)",0,2*pi)
```

21.256

Berikut kita definisikan fungsi yang sama namun dengan Maxima, untuk perhitungan eksak.

```
>&kill(ds,x,fx,fy)
```

done

```
>function ds(fx,fy) &=& sqrt(diff(fx,x)^2+diff(fy,x)^2)
```

$$\sqrt{\text{diff}^2(fy, x) + \text{diff}^2(fx, x)}$$

```
>sol &= ds(x*cos(x),x*sin(x)); $sol // // Kita gunakan untuk menghitung panjang kurva terakhir di at
```

$$\sqrt{(\cos x - x \sin x)^2 + (\sin x + x \cos x)^2}$$

```
>$sol | trigreduce | expand, $integrate(%,x,0,2*pi), %()
```

$$\sqrt{x^2 + 1}$$

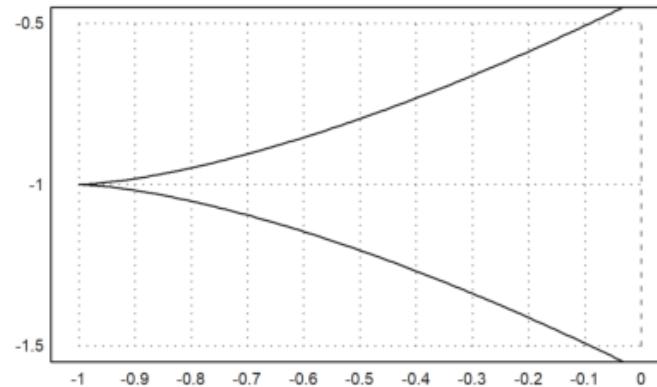
$$\frac{\operatorname{asinh}(2\pi) + 2\pi\sqrt{4\pi^2 + 1}}{2}$$

21.256

Hasilnya sama dengan perhitungan menggunakan fungsi EMT.

Berikut adalah contoh lain penggunaan fungsi Maxima tersebut.

```
>plot2d("3*x^2-1","3*x^3-1",xmin=-1/sqrt(3),xmax=1/sqrt(3),square=1):
```



```
>sol &= radcan(ds(3*x^2-1,3*x^3-1)); $sol
```

$$3x\sqrt{9x^2 + 4}$$

```
>$showev('integrate(sol,x,0,1/sqrt(3))), $2*float(%)
// panjang kurva di atas
```

$$3 \int_0^{\frac{1}{\sqrt{3}}} x \sqrt{9x^2 + 4} \, dx = 3 \left( \frac{7^{\frac{3}{2}}}{27} - \frac{8}{27} \right)$$
$$6.0 \int_{0.0}^{0.5773502691896258} x \sqrt{9.0x^2 + 4.0} \, dx = 2.337835372767141$$

Sikloid

---

Berikut kita akan menghitung panjang kurva lintasan (sikloid) suatu titik pada lingkaran yang berputar ke kanan pada permukaan datar. Misalkan jari-jari lingkaran tersebut adalah  $r$ . Posisi titik pusat lingkaran pada saat  $t$  adalah:

$$(rt, r).$$

Misalkan posisi titik pada lingkaran tersebut mula-mula  $(0,0)$  dan posisinya pada saat  $t$  adalah:

$$(r(t - \sin(t)), r(1 - \cos(t))).$$

Berikut kita plot lintasan tersebut dan beberapa posisi lingkaran ketika  $t=0$ ,  $t=\pi/2$ ,  $t=r\pi$ .

```
>x &= r*(t-sin(t))
```

$$r (t - \sin(t))$$

```
>y &= r*(1-cos(t))
```

$$r (1 - \cos(t))$$

Berikut kita gambar sikloid untuk r=1.

```
>ex &= x-sin(x); ey &= 1-cos(x); aspect(1);
>plot2d(ex,ey,xmin=0,xmax=4pi,square=1); ...
> plot2d("2+cos(x)","1+sin(x)",xmin=0,xmax=2pi,>add,color=blue); ...
> plot2d([2,ex(2)], [1,ey(2)],color=red,>add); ...
> plot2d(ex(2),ey(2),>points,>add,color=red); ...
> plot2d("2pi+cos(x)","1+sin(x)",xmin=0,xmax=2pi,>add,color=blue); ...
> plot2d([2pi,ex(2pi)], [1,ey(2pi)],color=red,>add); ...
> plot2d(ex(2pi),ey(2pi),>points,>add,color=red):
```

```
Cannot combine a 1x10000000 and a 1x21 matrix for *!
Error in expression: r*(t-sin(t))-sin(r*(t-sin(t)))
adaptiveeval:
sx=f$(t,args());
Try "trace errors" to inspect local variables after errors.
plot2d:
dw/n,dw/n^2,dw/n,args());
```

Berikut dihitung panjang lintasan untuk 1 putaran penuh. (Jangan salah menduga bahwa panjang lintasan 1 putaran penuh sama dengan keliling lingkaran!)

```
>ds &= radcan(sqrt(diff(ex,x)^2+diff(ey,x)^2)); $ds=trigsimp(ds) // elemen panjang kurva sikloid
```

```
Maxima said:
diff: second argument must be a variable; found r*(t-sin(t))
-- an error. To debug this try: debugmode(true);
```

```
Error in:  
ds &= radcan(sqrt(diff(ex,x)^2+diff(ey,x)^2)); $ds=trigsimp(ds ...  
^
```

```
>ds &= trigsimp(ds); $ds  
>$showev('integrate(ds,x,0,2*pi)) // hitung panjang sikloid satu putaran penuh
```

```
Maxima said:  
defint: variable of integration must be a simple or subscripted variable.  
defint: found r*(t-sin(t))  
#0: showev(f='integrate(ds,r*(t-sin(t)),0,2*pi))  
-- an error. To debug this try: debugmode(true);
```

```
Error in:  
$showev('integrate(ds,x,0,2*pi)) // hitung panjang sikloid sat ...  
^
```

```
>integrate(mxm("ds"),0,2*pi) // hitung secara numerik
```

```
Illegal function result in map.  
%evalexpression:  
    if maps then return %mapexpression1(x,f$;args());  
gauss:  
    if maps then y=%evalexpression(f$,a+h-(h*xn)',maps;args());  
adaptivegauss:  
    t1=gauss(f$,c,c+h;args(),=maps);  
Try "trace errors" to inspect local variables after errors.  
integrate:  
    return adaptivegauss(f$,a,b,eps*1000;args(),=maps);
```

```
>romberg(mxm("ds"),0,2*pi) // cara lain hitung secara numerik
```

Wrong argument!

Cannot combine a symbolic expression here.  
Did you want to create a symbolic expression?  
Then start with &.

Try "trace errors" to inspect local variables after errors.  
romberg:  
 if cols(y)==1 then return y\*(b-a); endif;  
Error in:  
romberg(mxm("ds"),0,2\*pi) // cara lain hitung secara numerik ...  
^

Perhatikan, seperti terlihat pada gambar, panjang sikloid lebih besar daripada keliling lingkarannya, yakni:

$\frac{2\pi}{r}$ .  
**Kurvatur (Kelengkungan) Kurva**

---

image: Osculating.png

Aslinya, kelengkungan kurva diferensiabel (yakni, kurva mulus yang tidak lancip) di titik P didefinisikan melalui lingkaran oskulasi (yaitu, lingkaran yang melalui titik P dan terbaik memperkirakan, paling banyak menyinggung kurva di sekitar P). Pusat dan radius kelengkungan kurva di P adalah pusat dan radius lingkaran oskulasi. Kelengkungan adalah kebalikan dari radius kelengkungan:

$$\kappa = \frac{1}{R}$$

dengan R adalah radius kelengkungan. (Setiap lingkaran memiliki kelengkungan ini pada setiap titiknya, dapat diartikan, setiap lingkaran berputar  $2\pi$  sejauh  $2\pi R$ .)

Definisi ini sulit dimanipulasi dan dinyatakan ke dalam rumus untuk kurva umum. Oleh karena itu digunakan definisi lain yang ekivalen.

---

### Definisi Kurvatur dengan Fungsi Parametrik Panjang Kurva Setiap

---

kurva diferensiabel dapat dinyatakan dengan persamaan parametrik terhadap panjang kurva s:

$$\gamma(s) = (x(s), y(s)),$$

dengan x dan y adalah fungsi riil yang diferensiabel, yang memenuhi:

$$\|\gamma'(s)\| = \sqrt{x'(s)^2 + y'(s)^2} = 1.$$

Ini berarti bahwa vektor singgung

$$\mathbf{T}(s) = (x'(s), y'(s))$$

memiliki norm 1 dan merupakan vektor singgung satuan.

Apabila kurvanya memiliki turunan kedua, artinya turunan kedua x dan y ada, maka  $\mathbf{T}'(s)$  ada. Vektor ini merupakan normal kurva yang arahnya menuju pusat kurvatur, norm-nya merupakan nilai kurvatur (kelengkungan):

$$\begin{aligned}\mathbf{T}(s) &= \gamma'(s), \\ \mathbf{T}^2(s) &= 1 \text{ (konstanta)} \Rightarrow \mathbf{T}'(s) \cdot \mathbf{T}(s) = 0 \\ \kappa(s) &= \|\mathbf{T}'(s)\| = \|\gamma''(s)\| = \sqrt{x''(s)^2 + y''(s)^2}.\end{aligned}$$

Nilai

$$R(s) = \frac{1}{\kappa(s)}$$

disebut jari-jari (radius) kelengkungan kurva.

Bilangan riil

$$k(s) = \pm \kappa(s)$$

disebut nilai kelengkungan bertanda.

Contoh:

Akan ditentukan kurvatur lingkaran

$$x = r \cos t, \quad y = r \sin t.$$

```
>fx &= r*cos(t); fy &=r*sin(t);
>&assume(t>0,r>0); s &=integrate(sqrt(diff(fx,t)^2+diff(fy,t)^2),t,0,t); s // elemen panjang kurva,
```

r t

```
>&kill(s); fx &= r*cos(s/r); fy &=r*sin(s/r); // definisi ulang persamaan parametrik terhadap s dengan r
>k &= trigsimp(sqrt(diff(fx,s,2)^2+diff(fy,s,2)^2)); $k // nilai kurvatur lingkaran dengan menggunakan r
```

$$\frac{1}{r}$$

Untuk representasi parametrik umum, misalkan

$$x = x(t), \quad y = y(t)$$

merupakan persamaan parametrik untuk kurva bidang yang terdiferensialkan dua kali. Kurvatur untuk kurva tersebut didefinisikan sebagai

$$\begin{aligned}\kappa &= \frac{d\phi}{ds} = \frac{\frac{d\phi}{dt}}{\frac{ds}{dt}} \quad (\phi \text{ adalah sudut kemiringan garis singgung dan } s \text{ adalah panjang kurva}) \\ &= \frac{\frac{d\phi}{dt}}{\sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2}} = \frac{\frac{d\phi}{dt}}{\sqrt{x'(t)^2 + y'(t)^2}}.\end{aligned}$$

Selanjutnya, pembilang pada persamaan di atas dapat dicari sebagai berikut.

$$\sec^2 \phi \frac{d\phi}{dt} = \frac{d}{dt} (\tan \phi) = \frac{d}{dt} \left( \frac{dy}{dx} \right) = \frac{d}{dt} \left( \frac{dy/dt}{dx/dt} \right) = \frac{d}{dt} \left( \frac{y'(t)}{x'(t)} \right) = \frac{x'(t)y''(t) - x''(t)y'(t)}{x'(t)^2}.$$

$$\begin{aligned}\frac{d\phi}{dt} &= \frac{1}{\sec^2 \phi} \frac{x'(t)y''(t) - x''(t)y'(t)}{x'(t)^2} \\ &= \frac{1}{1 + \tan^2 \phi} \frac{x'(t)y''(t) - x''(t)y'(t)}{x'(t)^2} \\ &= \frac{1}{1 + \left(\frac{y'(t)}{x'(t)}\right)^2} \frac{x'(t)y''(t) - x''(t)y'(t)}{x'(t)^2} \\ &= \frac{x'(t)y''(t) - x''(t)y'(t)}{x'(t)^2 + y'(t)^2}.\end{aligned}$$

Jadi, rumus kurvatur untuk kurva parametrik

$$x = x(t), \quad y = y(t)$$

adalah

$$\kappa(t) = \frac{x'(t)y''(t) - x''(t)y'(t)}{(x'(t)^2 + y'(t)^2)^{3/2}}.$$

Jika kurvanya dinyatakan dengan persamaan parametrik pada koordinat kutub

$$x = r(\theta) \cos \theta, \quad y = r(\theta) \sin \theta,$$

maka rumus kurvaturnya adalah

$$\kappa(\theta) = \frac{r(\theta)^2 + 2r'(\theta)^2 - r(\theta)r''(\theta)}{(r'(\theta)^2 + r''(\theta)^2)^{3/2}}.$$

(Silakan Anda turunkan rumus tersebut!)

Contoh:

Lingkaran dengan pusat (0,0) dan jari-jari r dapat dinyatakan dengan persamaan parametrik

$$x = r \cos t, \quad y = r \sin t.$$

Nilai kelengkungan lingkaran tersebut adalah

$$\kappa(t) = \frac{x'(t)y''(t) - x''(t)y'(t)}{(x'(t)^2 + y'(t)^2)^{3/2}} = \frac{r^2}{r^3} = \frac{1}{r}.$$

Hasil cocok dengan definisi kurvatur suatu kelengkungan.

## Kurva

$$y = f(x)$$

dapat dinyatakan ke dalam persamaan parametrik

$$x = t, \quad y = f(t), \quad \text{dengan } x'(t) = 1, \quad x''(t) = 0,$$

sehingga kurvaturnya adalah

$$\kappa(t) = \frac{y''(t)}{(1 + y'(t)^2)^{3/2}}.$$

Contoh:

Akan ditentukan kurvatur parabola

$$y = ax^2 + bx + c.$$

```
>function f(x) &= a*x^2+b*x+c; $y=f(x)
```

$$r (1 - \cos t) = b r (t - \sin t) + a r^2 (t - \sin t)^2 + c$$

```
>function k(x) &= (diff(f(x),x,2))/(1+diff(f(x),x)^2)^(3/2); $'k(x)=k(x) // kelengkungan parabola
```

```

Maxima said:
diff: second argument must be a variable; found r*(t-sin(t))
-- an error. To debug this try: debugmode(true);

Error in:
... (x) &= (diff(f(x),x,2))/(1+diff(f(x),x)^2)^(3/2); $'k(x)=k(x) ...

```

```
>function f(x) &= x^2+x+1; $y=f(x) // akan kita plot kelengkungan parabola untuk a=b=c=1
```

$$r(1 - \cos t) = r(t - \sin t) + r^2(t - \sin t)^2 + 1$$

```
>function k(x) &= (diff(f(x),x,2))/(1+diff(f(x),x)^2)^(3/2); $'k(x)=k(x) // kelengkungan parabola
```

```

Maxima said:
diff: second argument must be a variable; found r*(t-sin(t))
-- an error. To debug this try: debugmode(true);

Error in:
... (x) &= (diff(f(x),x,2))/(1+diff(f(x),x)^2)^(3/2); $'k(x)=k(x) ...

```

Berikut kita gambar parabola tersebut beserta kurva kelengkungan, kurva jari-jari kelengkungan dan salah satu lingkaran oskulasi di titik puncak parabola. Perhatikan, puncak parabola dan jari-jari lingkaran oskulasi di puncak parabola adalah

$$(-1/2, 3/4), \quad 1/k(2) = 1/2,$$

sehingga pusat lingkaran oskulasi adalah  $(-1/2, 5/4)$ .

```
>plot2d(["f(x)", "k(x)"], -2, 1, color=[blue, red]); plot2d("1/k(x)", -1.5, 1, color=green, >add); ...
>plot2d("-1/2+1/k(-1/2)*cos(x)", "5/4+1/k(-1/2)*sin(x)", xmin=0, xmax=2pi, >add, color=blue):
```

```
Cannot combine a 1x10000000 and a 1x21 matrix for *!
f:
useglobal; return r*(t-sin(t))+r^2*(t-sin(t))^2+1
Error in expression: f(x)
%ploteval:
y0=f$(x[1],args());
adaptiveevalone:
s=%ploteval(g$,t,args());
Try "trace errors" to inspect local variables after errors.
plot2d:
dw/n,dw/n^2,dw/n,auto;args());
```

Untuk kurva yang dinyatakan dengan fungsi implisit

$$F(x, y) = 0$$

dengan turunan-turunan parsial

$$F_x = \frac{\partial F}{\partial x}, \quad F_y = \frac{\partial F}{\partial y}, \quad F_{xy} = \frac{\partial}{\partial y} \left( \frac{\partial F}{\partial x} \right), \quad F_{xx} = \frac{\partial}{\partial x} \left( \frac{\partial F}{\partial x} \right), \quad F_{yy} = \frac{\partial}{\partial y} \left( \frac{\partial F}{\partial y} \right),$$

berlaku

$$F_x dx + F_y dy = 0 \text{ atau } \frac{dy}{dx} = -\frac{F_x}{F_y},$$

sehingga kurvaturnya adalah

$$\kappa = \frac{F_y^2 F_{xx} - 2F_x F_y F_{xy} + F_x^2 F_{yy}}{(F_x^2 + F_y^2)^{3/2}}.$$

(Silakan Anda turunkan sendiri!)

Contoh 1:

Parabola

$$y = ax^2 + bx + c$$

dapat dinyatakan ke dalam persamaan implisit

$$ax^2 + bx + c - y = 0.$$

```
>function F(x,y) &=a*x^2+b*x+c-y; $F(x,y)
```

$$b r (t - \sin t) + a r^2 (t - \sin t)^2 - r (1 - \cos t) + c$$

```
>Fx &= diff(F(x,y),x), Fxx &=diff(F(x,y),x,2), Fy &=diff(F(x,y),y), Fxy &=diff(diff(F(x,y),x),y), Fy
```

```
Maxima said:  
diff: second argument must be a variable; found errexp1  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
Fx &= diff(F(x,y),x), Fxx &=diff(F(x,y),x,2), Fy &=diff(F(x,y) ...  
^
```

```
>function k(x) &= (Fy^2*Fxx-2*Fx*Fy*Fxy+Fx^2*Fyy)/(Fx^2+Fy^2)^(3/2); $'k(x)=k(x) // kurvatur parabol
```

Hasilnya sama dengan sebelumnya yang menggunakan persamaan parabola biasa.

## Latihan

- 
- Bukalah buku Kalkulus.
  - Cari dan pilih beberapa (paling sedikit 5 fungsi berbeda tipe/bentuk/jenis) fungsi dari buku tersebut, kemudian definisikan di EMT pada baris-baris perintah berikut (jika perlu tambahkan lagi).
  - Untuk setiap fungsi, tentukan anti turunannya (jika ada), hitunglah integral tentu dengan batas-batas yang menarik (Anda tentukan sendiri), seperti contoh-contoh tersebut.
  - Lakukan hal yang sama untuk fungsi-fungsi yang tidak dapat diintegralkan (cari sedikitnya 3 fungsi).
  - Gambar grafik fungsi dan daerah integrasinya pada sumbu koordinat yang sama.
  - Gunakan integral tentu untuk mencari luas daerah yang dibatasi oleh dua kurva yang berpotongan di dua titik. (Cari dan gambar kedua kurva dan arsir (warnai) daerah yang dibatasi oleh keduanya.)
  - Gunakan integral tentu untuk menghitung volume benda putar kurva  $y=f(x)$  yang diputar mengelilingi sumbu x dari  $x=a$  sampai  $x=b$ , yakni

$$V = \int_a^b \pi(f(x)^2) dx.$$

(Pilih fungsinya dan gambar kurva dan benda putar yang dihasilkan. Anda dapat mencari contoh-contoh bagaimana cara menggambar benda hasil perputaran suatu kurva.)

- Gunakan integral tentu untuk menghitung panjang kurva  $y=f(x)$  dari  $x=a$  sampai  $x=b$  dengan menggunakan rumus:

$$S = \int_a^b \sqrt{1 + (f'(x))^2} dx.$$

(Pilih fungsi dan gambar kurvanya.)

- Apabila fungsi dinyatakan dalam koordinat kutub  $x=f(r,t)$ ,  $y=g(r,t)$ ,  $r=h(t)$ ,  $x=a$  bersesuaian dengan  $t=t_0$  dan  $x=b$  bersesuaian dengan  $t=t_1$ , maka rumus di atas akan menjadi:

$$S = \int_{t_0}^{t_1} \sqrt{x'(t)^2 + y'(t)^2} dt.$$

- Pilih beberapa kurva menarik (selain lingkaran dan parabola) dari buku kalkulus. Nyatakan setiap kurva tersebut dalam bentuk:
  - a. koordinat Kartesius (persamaan  $y=f(x)$ )
  - b. koordinat kutub ( $r=r(\theta)$ )
  - c. persamaan parametrik  $x=x(t)$ ,  $y=y(t)$
  - d. persamaan implisit  $F(x,y)=0$
- Tentukan kurvatur masing-masing kurva dengan menggunakan keempat representasi tersebut (hasilnya harus sama).
- Gambarlah kurva asli, kurva kurvatur, kurva jari-jari lingkaran oskulasi, dan salah satu lingkaran oskulasinya.

**Barisan dan Deret**

---

(Catatan: bagian ini belum lengkap. Anda dapat membaca contoh-contoh penggunaan EMT dan Maxima untuk menghitung limit barisan, rumus jumlah parsial suatu deret, jumlah tak hingga suatu deret konvergen, dan sebagainya. Anda dapat mengeksplor contoh-contoh di EMT atau perbagai panduan penggunaan Maxima di software Maxima atau dari Internet.)

Barisan dapat didefinisikan dengan beberapa cara di dalam EMT, di antaranya:

- dengan cara yang sama seperti mendefinisikan vektor dengan elemen-elemen beraturan (menggunakan titik dua ":");
- menggunakan perintah "sequence" dan rumus barisan (suku ke -n);
- menggunakan perintah "iterate" atau "niterate";
- menggunakan fungsi Maxima "create\_list" atau "makelist" untuk menghasilkan barisan simbolik;
- menggunakan fungsi biasa yang inputnya vektor atau barisan;
- menggunakan fungsi rekursif.

EMT menyediakan beberapa perintah (fungsi) terkait barisan, yakni:

- sum: menghitung jumlah semua elemen suatu barisan
- cumsum: jumlah kumulatif suatu barisan
- differences: selisih antar elemen-elemen berturutan

EMT juga dapat digunakan untuk menghitung jumlah deret berhingga maupun deret tak hingga, dengan menggunakan perintah (fungsi) "sum". Perhitungan dapat dilakukan secara numerik maupun simbolik dan eksak.

Berikut adalah beberapa contoh perhitungan barisan dan deret menggunakan EMT.

```
>1:10 // barisan sederhana
```

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

>1:2:30

```
[1,  3,  5,  7,  9,  11,  13,  15,  17,  19,  21,  23,  25,  27,  29]
```

## Iterasi dan Barisan

---

EMT menyediakan fungsi iterate("g(x)", x0, n) untuk melakukan iterasi

$$x_{k+1} = g(x_k), \quad x_0 = x_0, \quad k = 1, 2, 3, \dots, n.$$

Berikut ini disajikan contoh-contoh penggunaan iterasi dan rekursi dengan EMT. Contoh pertama menunjukkan pertumbuhan dari nilai awal 1000 dengan laju pertambahan 5%, selama 10 periode.

```
>q=1.05; iterate("x*q",1000,n=10)'
```

```
1000  
1050  
1102.5  
1157.63  
1215.51  
1276.28  
1340.1  
1407.1  
1477.46  
1551.33  
1628.89
```

Contoh berikutnya memperlihatkan bahaya menabung di bank pada masa sekarang! Dengan bunga tabungan sebesar 6% per tahun atau 0.5% per bulan dipotong pajak 20%, dan biaya administrasi 10000 per bulan, tabungan sebesar 1 juta tanpa diambil selama sekitar 10 tahunan akan habis diambil oleh bank!

```
>r=0.005; plot2d(iterate("(1+0.8*r)*x-10000",1000000,n=130)):
```

Silakan Anda coba-coba, dengan tabungan minimal berapa agar tidak akan habis diambil oleh bank dengan ketentuan bunga dan biaya administrasi seperti di atas.

Berikut adalah perhitungan minimal tabungan agar aman di bank dengan bunga sebesar r dan biaya administrasi a, pajak bunga 20%.

```
>$solve(0.8*r*A-a,A), $% with [r=0.005, a=10]
```

Berikut didefinisikan fungsi untuk menghitung saldo tabungan, kemudian dilakukan iterasi.

```
>function saldo(x,r,a) := round((1+0.8*r)*x-a,2);
>iterate({{"saldo",0.005,10}},1000,n=6)
```

```
[1000, 994, 987.98, 981.93, 975.86, 969.76, 963.64]
```

```
>iterate({{"saldo",0.005,10}},2000,n=6)
```

```
[2000, 1998, 1995.99, 1993.97, 1991.95, 1989.92, 1987.88]
```

```
>iterate( {"saldo",0.005,10} },2500,n=6)
```

```
[2500, 2500, 2500, 2500, 2500, 2500, 2500]
```

Tabungan senilai 2,5 juta akan aman dan tidak akan berubah nilai (jika tidak ada penarikan), sedangkan jika tabungan awal kurang dari 2,5 juta, lama kelamaan akan berkurang meskipun tidak pernah dilakukan penarikan uang tabungan.

```
>iterate( {"saldo",0.005,10} },3000,n=6)
```

```
[3000, 3002, 3004.01, 3006.03, 3008.05, 3010.08, 3012.12]
```

Tabungan yang lebih dari 2,5 juta baru akan bertambah jika tidak ada penarikan.

Untuk barisan yang lebih kompleks dapat digunakan fungsi "sequence()". Fungsi ini menghitung nilai-nilai  $x[n]$  dari semua nilai sebelumnya,  $x[1], \dots, x[n-1]$  yang diketahui.

Berikut adalah contoh barisan Fibonacci.

$$x_n = x_{n-1} + x_{n-2}, \quad x_1 = 1, \quad x_2 = 1$$

```
>sequence("x[n-1]+x[n-2]",[1,1],15)
```

```
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610]
```

Barisan Fibonacci memiliki banyak sifat menarik, salah satunya adalah akar pangkat ke-n suku ke-n akan konvergen ke pecahan emas:

```
>$(1+sqrt(5))/2=float((1+sqrt(5))/2)
>plot2d(sequence("x[n-1]+x[n-2]",[1,1],250)^(1/(1:250))):
```

Barisan yang sama juga dapat dihasilkan dengan menggunakan loop.

```
>x=ones(500); for k=3 to 500; x[k]=x[k-1]+x[k-2]; end;
```

Rekursi dapat dilakukan dengan menggunakan rumus yang tergantung pada semua elemen sebelumnya. Pada contoh berikut, elemen ke-n merupakan jumlah (n-1) elemen sebelumnya, dimulai dengan 1 (elemen ke-1). Jelas, nilai elemen ke-n adalah  $2^{(n-2)}$ , untuk n=2, 4, 5, ....

```
>sequence("sum(x)",1,10)
```

```
[1, 1, 2, 4, 8, 16, 32, 64, 128, 256]
```

Selain menggunakan ekspresi dalam x dan n, kita juga dapat menggunakan fungsi.

Pada contoh berikut, digunakan iterasi

$$x_n = A \cdot x_{n-1},$$

dengan A suatu matriks 2x2, dan setiap  $x[n]$  merupakan matriks/vektor 2x1.

```
>A=[1,1;1,2]; function suku(x,n) := A.x[,n-1]
>sequence("suku",[1;1],6)
```

Real 2 x 6 matrix

1	2	5	13	...
1	3	8	21	...

Hasil yang sama juga dapat diperoleh dengan menggunakan fungsi perpangkatan matriks "matrix-power()". Cara ini lebih cepat, karena hanya menggunakan perkalian matriks sebanyak  $\log_2(n)$ .

$$x_n = A \cdot x_{n-1} = A^2 \cdot x_{n-2} = A^3 \cdot x_{n-3} = \dots = A^{n-1} \cdot x_1.$$

```
>sequence("matrixpower(A,n).[1;1]",1,6)
```

Real 2 x 6 matrix

1	5	13	34	...
1	8	21	55	...

## Spiral Theodorus

---

image: Spiral\_of\_Theodorus.png

Spiral Theodorus (spiral segitiga siku-siku) dapat digambar secara rekursif. Rumus rekursifnya adalah:

$$x_n = \left(1 + \frac{i}{\sqrt{n-1}}\right) x_{n-1}, \quad x_1 = 1,$$

yang menghasilkan barisan bilangan kompleks.

```
>function g(n) := 1+I/sqrt(n)
```

Rekursinya dapat dijalankan sebanyak 17 untuk menghasilkan barisan 17 bilangan kompleks, kemudian digambar bilangan-bilangan kompleksnya.

```
>x=sequence("g(n-1)*x[n-1]",1,17); plot2d(x,r=3.5); textbox(latex("Spiral\ Theodorus"),0.4):
```

Selanjutnya dihubungan titik 0 dengan titik-titik kompleks tersebut menggunakan loop.

```
>for i=1:cols(x); plot2d([0,x[i]],>add); end:
```

Spiral tersebut juga dapat didefinisikan menggunakan fungsi rekursif, yang tidak memerlukan indeks dan bilangan kompleks. Dalam hal ini digunakan vektor kolom pada bidang.

```
>function gstep (v) ...
```

```
w=[-v[2];v[1]];
return v+w/norm(w);
endfunction
```

Jika dilakukan iterasi 16 kali dimulai dari [1;0] akan didapatkan matriks yang memuat vektor-vektor dari setiap iterasi.

```
>x=iterate("gstep",[1;0],16); plot2d(x[1],x[2],r=3.5,>points):
```

## Kekonvergenan

---

Terkadang kita ingin melakukan iterasi sampai konvergen. Apabila iterasinya tidak konvergen setelah ditunggu lama, Anda dapat menghentikannya dengan menekan tombol [ESC].

```
>iterate("cos(x)",1) // iterasi x(n+1)=cos(x(n)), dengan x(0)=1.
```

0.739085133216

Iterasi tersebut konvergen ke penyelesaian persamaan

$$x = \cos(x).$$

Iterasi ini juga dapat dilakukan pada interval, hasilnya adalah barisan interval yang memuat akar tersebut.

```
>hasil := iterate("cos(x)",~1,2~) //iterasi x(n+1)=cos(x(n)), dengan interval awal (1, 2)
```

~0.739085133211,0.7390851332133~

Jika interval hasil tersebut sedikit diperlebar, akan terlihat bahwa interval tersebut memuat akar persamaan  $x=\cos(x)$ .

```
>h=expand(hasil,100), cos(h) << h
```

```
~0.73908513309,0.73908513333~  
1
```

Iterasi juga dapat digunakan pada fungsi yang didefinisikan.

```
>function f(x) := (x+2/x)/2
```

Iterasi  $x(n+1)=f(x(n))$  akan konvergen ke akar kuadrat 2.

```
>iterate("f",2), sqrt(2)
```

```
1.41421356237  
1.41421356237
```

Jika pada perintah iterate diberikan tambahan parameter n, maka hasil iterasinya akan ditampilkan mulai dari iterasi pertama sampai ke-n.

```
>iterate("f",2,5)
```

```
[2, 1.5, 1.41667, 1.41422, 1.41421, 1.41421]
```

Untuk iterasi ini tidak dapat dilakukan terhadap interval.

```
>niterate("f",~1,2~,5)
```

```
[ ~1,2~, ~1,2~, ~1,2~, ~1,2~, ~1,2~, ~1,2~ ]
```

Perhatikan, hasil iterasinya sama dengan interval awal. Alasannya adalah perhitungan dengan interval bersifat terlalu longgar. Untuk meingkatkan perhitungan pada ekspresi dapat digunakan pembagian intervalnya, menggunakan fungsi ieval().

```
>function s(x) := ieval("(x+2/x)/2",x,10)
```

Selanjutnya dapat dilakukan iterasi hingga diperoleh hasil optimal, dan intervalnya tidak semakin mengecil. Hasilnya berupa interval yang memuat akar persamaan:

$$x = \frac{1}{2} \left( x + \frac{2}{x} \right).$$

Satu-satunya solusi adalah

$$x = \sqrt{2}.$$

```
>iterate("s",~1,2~)
```

```
~1.41421356236,1.41421356239~
```

Fungsi "iterate()" juga dapat bekerja pada vektor. Berikut adalah contoh fungsi vektor, yang menghasilkan rata-rata aritmetika dan rata-rata geometri.

$$(a_{n+1}, b_{n+1}) = \left( \frac{a_n + b_n}{2}, \sqrt{a_n b_n} \right)$$

Iterasi ke-n disimpan pada vektor kolom x[n].

```
>function g(x) := [(x[1]+x[2])/2,sqrt(x[1]*x[2])]
```

Iterasi dengan menggunakan fungsi tersebut akan konvergen ke rata-rata aritmetika dan geometri dari nilai-nilai awal.

```
>iterate("g", [1;5])
```

```
2.60401  
2.60401
```

Hasil tersebut konvergen agak cepat, seperti kita cek sebagai berikut.

```
>iterate("g", [1;5], 4)
```

1	3	2.61803	2.60403	2.60401
5	2.23607	2.59002	2.60399	2.60401

Iterasi pada interval dapat dilakukan dan stabil, namun tidak menunjukkan bahwa limitnya pada batas-batas yang dihitung.

```
>iterate("g", [~1~;~5~], 4)
```

Interval 2 x 5 matrix

```
~0.99999999999999778, 1.00000000000000022~ ...  
~4.999999999999911, 5.0000000000000089~ ...
```

Iterasi berikut konvergen sangat lambat.

$$x_{n+1} = \sqrt{x_n}.$$

```
>iterate("sqrt(x)",2,10)
```

```
[2, 1.41421, 1.18921, 1.09051, 1.04427, 1.0219, 1.01089,
1.00543, 1.00271, 1.00135, 1.00068]
```

Kekonvergenan iterasi tersebut dapat dipercepat dengan percepatan Steffenson:

```
>steffenson("sqrt(x)",2,10)
```

```
[1.04888, 1.00028, 1, 1]
```

## Iterasi menggunakan Loop yang ditulis Langsung

Berikut adalah beberapa contoh penggunaan loop untuk melakukan iterasi yang ditulis langsung pada baris perintah.

```
>x=2; repeat x=(x+2/x)/2; until x^2~=2; end; x,
```

1.41421356237

Penggabungan matriks menggunakan tanda "|" dapat digunakan untuk menyimpan semua hasil iterasi.

```
>v=[1]; for i=2 to 8; v=v|(v[i-1]*i); end; v,
```

[1, 2, 6, 24, 120, 720, 5040, 40320]

hasil iterasi juga dapat disimpan pada vektor yang sudah ada.

```
>v=ones(1,100); for i=2 to cols(v); v[i]=v[i-1]*i; end; ...
>plot2d(v,logplot=1); textbox(latex(&log(n)),x=0.5);
>A =[0.5,0.2;0.7,0.1]; b=[2;2]; ...
>x=[1;1]; repeat xnew=A.x-b; until all(xnew^~x); x=xnew; end; ...
>x
```

```
-7.09677
-7.74194
```

## Iterasi di dalam Fungsi

---

Fungsi atau program juga dapat menggunakan iterasi dan dapat digunakan untuk melakukan iterasi. Berikut adalah beberapa contoh iterasi di dalam fungsi.

Contoh berikut adalah suatu fungsi untuk menghitung berapa lama suatu iterasi konvergen. Nilai fungsi tersebut adalah hasil akhir iterasi dan banyak iterasi sampai konvergen.

```
>function map hiter(f$,x0) ...
```

```
x=x0;
maxiter=0;
repeat
    xnew=f$(x);
    maxiter=maxiter+1;
    until xnew~=x;
    x=xnew;
end;
return maxiter;
endfunction
```

Misalnya, berikut adalah iterasi untuk mendapatkan hampiran akar kuadrat 2, cukup cepat, konvergen pada iterasi ke-5, jika dimulai dari hampiran awal 2.

```
>hiter("(x+2/x)/2",2)
```

Karena fungsinya didefinisikan menggunakan "map". maka nilai awalnya dapat berupa vektor.

```
>x=1.5:0.1:10; hasil=hiter("(x+2/x)/2",x); ...
> plot2d(x,hasil);
```

Dari gambar di atas terlihat bahwa kekonvergenan iterasinya semakin lambat, untuk nilai awal semakin besar, namun penambahannya tidak kontinu. Kita dapat menemukan kapan maksimum iterasinya bertambah.

```
>hasil[1:10]
```

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

```
>x[nonzeros(differences(hasil))]
```

```
[1.5, 2, 3.4, 6.6]
```

maksimum iterasi sampai konvergen meningkat pada saat nilai awalnya 1.5, 2, 3.4, dan 6.6. Contoh berikutnya adalah metode Newton pada polinomial kompleks berderajat 3.

```
>p &= x^3-1; newton &= x-p/diff(p,x); $newton
```

```
Maxima said:  
diff: second argument must be a variable; found errexp1  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
p &= x^3-1; newton &= x-p/diff(p,x); $newton ...  
^
```

Selanjutnya didefinisikan fungsi untuk melakukan iterasi (aslinya 10 kali).

```
>function iterasi(f$,x,n=10) ...
```

```
loop 1 to n; x=f$(x); end;  
return x;  
endfunction
```

Kita mulai dengan menentukan titik-titik grid pada bidang kompleksnya.

```
>r=1.5; x=linspace(-r,r,501); Z=x+I*x'; W=iterasi(newton,Z);
```

```
Function newton needs at least 3 arguments!
Use: newton (f$: call, df$: call, x: scalar complex {, y: number, eps: none})
Error in:
... x=linspace(-r,r,501); Z=x+I*x'; W=iterasi(newton,Z); ...  
^
```

Berikut adalah akar-akar polinomial di atas.

```
>z=&solve(p)()
```

```
Maxima said:
solve: more equations than unknowns.
Unknowns given :
[r]
Equations given:
errexp1
-- an error. To debug this try: debugmode(true);

Error in:
z=&solve(p)() ...  
^
```

Untuk menggambar hasil iterasinya, dihitung jarak dari hasil iterasi ke-10 ke masing-masing akar, kemudian digunakan untuk menghitung warna yang akan digambar, yang menunjukkan limit untuk masing-masing nilai awal.

Fungsi `plotrgb()` menggunakan jendela gambar terkini untuk menggambar warna RGB sebagai matriks.

```
>C=rgb(max(abs(W-z[1]),1),max(abs(W-z[2]),1),max(abs(W-z[3]),1)); ...
> plot2d(none,-r,r,-r,r); plotrgb(C):
```

Variable W not found!

Error in:

```
C=rgb(max(abs(W-z[1]),1),max(abs(W-z[2]),1),max(abs(W-z[3]),1) ...
```

^

## Iterasi Simbolik

---

Seperti sudah dibahas sebelumnya, untuk menghasilkan barisan ekspresi simbolik dengan Maxima dapat digunakan fungsi makelist().

```
>&powerdisp:true // untuk menampilkan deret pangkat mulai dari suku berpangkat terkecil
```

```
true
```

```
>deret &= makelist(taylor(exp(x),x,0,k),k,1,3); $deret // barisan deret Taylor untuk e^x
```

```
Maxima said:  
taylor: 0.1539740213994798*r cannot be a variable.  
-- an error. To debug this try: debugmode(true);
```

```
Error in:  
deret &= makelist(taylor(exp(x),x,0,k),k,1,3); $deret // baris ...  
^
```

Untuk mengubah barisan deret tersebut menjadi vektor string di EMT digunakan fungsi mxm2str(). Selanjutnya, vektor string/ekspresi hasilnya dapat digambar seperti menggambar vektor ekspresi pada EMT.

```
>plot2d("exp(x)",0,3); // plot fungsi aslinya, e^x  
>plot2d(mxm2str("deret"),>add,color=4:6); // plot ketiga deret taylor hampiran fungsi tersebut
```

```
Maxima said:  
length: argument cannot be a symbol; found deret  
-- an error. To debug this try: debugmode(true);  
  
mxmeval:  
    return evaluate(mxm(s));  
Try "trace errors" to inspect local variables after errors.  
mxm2str:  
n=mxmeval("length(VVV)");
```

Selain cara di atas dapat juga dengan cara menggunakan indeks pada vektor/list yang dihasilkan.

```
>$deret[3]  
>plot2d(["exp(x)",&deret[1],&deret[2],&deret[3]],0,3,color=1:4):
```

```
deret is not a variable!  
Error in expression: deret[1]  
%ploteval:  
    y0=f$(x[1],args());  
Try "trace errors" to inspect local variables after errors.  
plot2d:  
u=u_(%ploteval(xx[#,t;args()));
```

```
>$sum(sin(k*x)/k,k,1,5)
```

Berikut adalah cara menggambar kurva

$$y = \sin(x) + \frac{\sin 3x}{3} + \frac{\sin 5x}{5} + \dots$$

```
>plot2d(&sum(sin((2*k+1)*x)/(2*k+1),k,0,20),0,2pi):
```

```
Maxima output too long!
Error in:
plot2d(&sum(sin((2*k+1)*x)/(2*k+1),k,0,20),0,2pi): ...  
^
```

Hal serupa juga dapat dilakukan dengan menggunakan matriks, misalkan kita akan menggambar kurva

$$y = \sum_{k=1}^{100} \frac{\sin(kx)}{k}, \quad 0 \leq x \leq 2\pi.$$

```
>x=linspace(0,2pi,1000); k=1:100; y=sum(sin(k*x')/k)'; plot2d(x,y):
```

## Tabel Fungsi

---

Terdapat cara menarik untuk menghasilkan barisan dengan ekspresi Maxima. Perintah mxmtable() berguna untuk menampilkan dan menggambar barisan dan menghasilkan barisan sebagai vektor kolom.

Sebagai contoh berikut adalah barisan turunan ke-n  $x^x$  di  $x=1$ .

```
>mxmtable("diffat(x^x,x=1,n)","n",1,8,frac=1);
```

Maxima said:

```
diff: second argument must be a variable; found errexp1
#0: diffat(expr=[0,1.66665833335744e-7*r,1.33330666692022e-6*r,4.499797504338432e-6*r,1.0665813365
-- an error. To debug this try: debugmode(true);
```

```
%mxmegttable:
```

```
    return mxm("@expr,@var=@value")();
```

```
Try "trace errors" to inspect local variables after errors.
```

```
mxmtable:
```

```
y[#,1]=%mxmegttable(expr,var,x[#]);
```

```
>$'sum(k, k, 1, n) = factor(ev(sum(k, k, 1, n),simpsum=true)) // simpsum:menghitung deret secara sim
>$'sum(1/(3^k+k), k, 0, inf) = factor(ev(sum(1/(3^k+k), k, 0, inf),simpsum=true))
```

Di sini masih gagal, hasilnya tidak dihitung.

```
>${'sum(1/x^2, x, 1, inf)= ev(sum(1/x^2, x, 1, inf),simpsum=true) // ev: menghitung nilai ekspresi
>${'sum((-1)^(k-1)/k, k, 1, inf) = factor(ev(sum((-1)^(x-1)/x, x, 1, inf),simpsum=true))}
```

Di sini masih gagal, hasilnya tidak dihitung.

```
>${'sum((-1)^k/(2*k-1), k, 1, inf) = factor(ev(sum((-1)^k/(2*k-1), k, 1, inf),simpsum=true))
>${'ev(sum(1/n!, n, 0, inf),simpsum=true)}
```

Di sini masih gagal, hasilnya tidak dihitung, harusnya hasilnya e.

```
>&assume(abs(x)<1); ${'sum(a*x^k, k, 0, inf)=ev(sum(a*x^k, k, 0, inf),simpsum=true), &forget(abs(x)<1)}
```

Answering "Is  $-94914474571+15819r$  positive, negative or zero?" with "positive"  
Maxima said:  
sum: sum is divergent.  
-- an error. To debug this try: debugmode(true);

Error in:  
... k, 0, inf)=ev(sum(a\*x^k, k, 0, inf),simpsum=true), &forget(abs ...  
^

Deret geometri tak hingga, dengan asumsi rasional antara -1 dan 1.

```
>$'sum(x^k/k!,k,0,inf)=ev(sum(x^k/k!,k,0,inf),simpsum=true)
>$limit(sum(x^k/k!,k,0,n),n,inf)
>function d(n) &= sum(1/(k^2-k),k,2,n); '$d(n)=d(n)
>$d(10)=ev(d(10),simpsum=true)
>$d(100)=ev(d(100),simpsum=true)
```

## Deret Taylor

---

Deret Taylor suatu fungsi  $f$  yang diferensiabel sampai tak hingga di sekitar  $x=a$  adalah:

$$f(x) = \sum_{k=0}^{\infty} \frac{(x-a)^k f^{(k)}(a)}{k!}.$$

```
> $'e^x =taylor(exp(x),x,0,10) // deret Taylor e^x di sekitar x=0, sampai suku ke-11
```

```
Maxima said:  
taylor: 0.1539740213994798*r cannot be a variable.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$'e^x =taylor(exp(x),x,0,10) // deret Taylor e^x di sekitar x= ...  
^
```

```
> $'log(x)=taylor(log(x),x,1,10)// deret log(x) di sekitar x=1
```

```
Maxima said:  
log: encountered log(0).  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$'log(x)=taylor(log(x),x,1,10)// deret log(x) di sekitar x=1 ...  
^
```

## Visualisasi dan Perhitungan Geometri dengan EMT

---

Euler menyediakan beberapa fungsi untuk melakukan visualisasi dan perhitungan geometri, baik secara numerik maupun analitik (seperti biasanya tentunya, menggunakan Maxima). Fungsi-fungsi untuk visualisasi dan perhitungan geometri tersebut disimpan di dalam file program "geometry.e", sehingga file tersebut harus dipanggil sebelum menggunakan fungsi-fungsi atau perintah-perintah untuk geometri.

```
>load geometry
```

Numerical and symbolic geometry.

### Fungsi-fungsi Geometri

---

Fungsi-fungsi untuk Menggambar Objek Geometri:

```
defaultd:=textheight()*1.5: nilai asli untuk parameter d  
setPlotrange(x1,x2,y1,y2): menentukan rentang x dan y pada bidang
```

koordinat

```
setPlotRange(r): pusat bidang koordinat (0,0) dan batas-batas
```

sumbu-x dan y adalah -r sd r

plotPoint (P, "P"): menggambar titik P dan diberi label "P"  
plotSegment (A,B, "AB", d): menggambar ruas garis AB, diberi label

"AB" sejauh d

plotLine (g, "g", d): menggambar garis g diberi label "g" sejauh d  
plotCircle (c,"c",v,d): Menggambar lingkaran c dan diberi label "c"  
plotLabel (label, P, V, d): menuliskan label pada posisi P

Fungsi-fungsi Geometri Analitik (numerik maupun simbolik):

turn(v, phi): memutar vektor v sejauh phi  
turnLeft(v): memutar vektor v ke kiri  
turnRight(v): memutar vektor v ke kanan  
normalize(v): normal vektor v  
crossProduct(v, w): hasil kali silang vektorv dan w.  
lineThrough(A, B): garis melalui A dan B, hasilnya [a,b,c] sdh.

$ax+by=c$ .

```
lineWithDirection(A,v): garis melalui A searah vektor v
getLineDirection(g): vektor arah (gradien) garis g
getNormal(g): vektor normal (tegak lurus) garis g
getPointOnLine(g): titik pada garis g
perpendicular(A, g): garis melalui A tegak lurus garis g
parallel (A, g): garis melalui A sejajar garis g
lineIntersection(g, h): titik potong garis g dan h
projectToLine(A, g): proyeksi titik A pada garis g
distance(A, B): jarak titik A dan B
distanceSquared(A, B): kuadrat jarak A dan B
quadrance(A, B): kuadrat jarak A dan B
areaTriangle(A, B, C): luas segitiga ABC
computeAngle(A, B, C): besar sudut <ABC
angleBisector(A, B, C): garis bagi sudut <ABC
circleWithCenter (A, r): lingkaran dengan pusat A dan jari-jari r
getCircleCenter(c): pusat lingkaran c
getCircleRadius(c): jari-jari lingkaran c
circleThrough(A,B,C): lingkaran melalui A, B, C
middlePerpendicular(A, B): titik tengah AB
lineCircleIntersections(g, c): titik potong garis g dan lingkran c
circleCircleIntersections (c1, c2): titik potong lingkaran c1 dan
```

c2

```
planeThrough(A, B, C): bidang melalui titik A, B, C
```

Fungsi-fungsi Khusus Untuk Geometri Simbolik:

```
getLineEquation (g,x,y): persamaan garis g dinyatakan dalam x dan y
getHesseForm (g,x,y,A): bentuk Hesse garis g dinyatakan dalam x dan
```

y dengan titik A pada

sisi positif (kanan/atas) garis

quad(A,B): kuadrat jarak AB

spread(a,b,c): Spread segitiga dengan panjang sisi-sisi a,b,c, yakni

$\sin(\alpha)^2$  dengan

alpha sudut yang menghadap sisi a.

crosslaw(a,b,c,sa): persamaan 3 quads dan 1 spread pada segitiga

dengan panjang sisi a, b, c.

triplespread(sa,sb,sc): persamaan 3 spread sa,sb,sc yang memebntuk

suatu segitiga

doublespread(sa): Spread sudut rangkap Spread  $2\phi$ , dengan

$sa = \sin(\phi)^2$  spread a.

## Contoh 1: Luas, Lingkaran Luar, Lingkaran Dalam Segitiga

---

Untuk menggambar objek-objek geometri, langkah pertama adalah menentukan rentang sumbu-sumbu koordinat. Semua objek geometri akan digambar pada satu bidang koordinat, sampai didefinisikan bidang koordinat yang baru.

```
>setPlotRange(-0.5,2.5,-0.5,2.5); // mendefinisikan bidang koordinat baru
```

Sekarang tetapkan tiga poin dan plot mereka.

```
>A=[1,0]; plotPoint(A,"A"); // definisi dan gambar tiga titik  
>B=[0,1]; plotPoint(B,"B");  
>C=[2,2]; plotPoint(C,"C");
```

Kemudian tiga segmen.

```
>plotSegment(A,B,"c"); // c=AB  
>plotSegment(B,C,"a"); // a=BC  
>plotSegment(A,C,"b"); // b=AC
```

Fungsi geometri meliputi fungsi untuk membuat garis dan lingkaran. Format garis adalah [a,b,c], yang mewakili garis dengan persamaan  $ax+by=c$ .

```
>lineThrough(B,C) // garis yang melalui B dan C
```

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

Hitunglah garis tegak lurus yang melalui A pada BC.

```
>h=perpendicular(A,lineThrough(B,C)); // garis h tegak lurus BC melalui A
```

Dan persimpangannya dengan BC.

```
>D=lineIntersection(h,lineThrough(B,C)); // D adalah titik potong h dan BC
```

Plot itu.

```
>plotPoint(D,value=1); // koordinat D ditampilkan  
>aspect(1); plotSegment(A,D); // tampilkan semua gambar hasil plot...()
```

Hitung luas ABC:

$$L_{\triangle ABC} = \frac{1}{2} AD \cdot BC.$$

```
>norm(A-D)*norm(B-C)/2 // AD=norm(A-D), BC=norm(B-C)
```

1.5

Bandingkan dengan rumus determinan.

```
>areaTriangle(A,B,C) // hitung luas segitiga langsung dengan fungsi
```

1.5

Cara lain menghitung luas segitiga ABC:

```
>distance(A,D)*distance(B,C)/2
```

1.5

Sudut di C

```
>degprint(computeAngle(B,C,A))
```

36°52'11.63''

Sekarang lingkaran luar segitiga.

```
>c=circleThrough(A,B,C); // lingkaran luar segitiga ABC
>R=getCircleRadius(c); // jari2 lingkaran luar
>O=getCircleCenter(c); // titik pusat lingkaran c
>plotPoint(O,"O"); // gambar titik "O"
>plotCircle(c,"Lingkaran luar segitiga ABC"):
```

Tampilkan koordinat titik pusat dan jari-jari lingkaran luar.

```
>O, R
```

```
[1.16667, 1.16667]
1.17851130198
```

Sekarang akan digambar lingkaran dalam segitiga ABC. Titik pusat lingkaran dalam adalah titik potong garis-garis bagi sudut.

```
>l=angleBisector(A,C,B); // garis bagi <ACB  
>g=angleBisector(C,A,B); // garis bagi <CAB  
>P=lineIntersection(l,g) // titik potong kedua garis bagi sudut
```

[0.86038, 0.86038]

Tambahkan semuanya ke plot.

```
>color(5); plotLine(l); plotLine(g); color(1); // gambar kedua garis bagi sudut  
>plotPoint(P,"P"); // gambar titik potongnya  
>r=norm(P-projectToLine(P,lineThrough(A,B))) // jari-jari lingkaran dalam
```

0.509653732104

```
>plotCircle(circleWithCenter(P,r),"Lingkaran dalam segitiga ABC"); // gambar lingkaran dalam
```

1. Tentukan ketiga titik singgung lingkaran dalam dengan sisi-sisi segitiga ABC.

```
>setPlotRange(-2.5,4.5,-2.5,4.5);
>A=[-2,1]; plotPoint(A,"A");
>B=[1,-2]; plotPoint(B,"B");
>C=[4,4]; plotPoint(C,"C");
```

2. Gambar segitiga dengan titik-titik sudut ketiga titik singgung tersebut.

```
>plotSegment(A,B,"c")
>plotSegment(B,C,"a")
>plotSegment(A,C,"b")
>aspect(1);
```

3. Tunjukkan bahwa garis bagi sudut yang ke tiga juga melalui titik pusat lingkaran dalam.

```
>l=angleBisector(A,C,B);
>g=angleBisector(C,A,B);
>P=lineIntersection(l,g)
```

[0.581139, 0.581139]

```
>color(5); plotLine(l); plotLine(g); color(1);
>plotPoint(P,"P");
>r=norm(P-projectToLine(P,lineThrough(A,B)))
```

1.52896119631

```
>plotCircle(circleWithCenter(P,r),"Lingkaran dalam segitiga ABC");
```

Jadi, terbukti bahwa garis bagi sudut yang ketiga juga melalui titik pusat lingkaran dalam.

4. Gambar jari-jari lingkaran dalam.

```
>r=norm(P-projectToLine(P,lineThrough(A,B)))
```

1.52896119631

```
>plotCircle(circleWithCenter(P,r),"Lingkaran dalam segitiga ABC");
```

## Contoh 2: Geometri Simbolik

---

Kita dapat menghitung geometri eksak dan simbolik menggunakan Maxima.

File geometri.e menyediakan fungsi yang sama (dan lebih banyak lagi) di Maxima. Namun, kita dapat menggunakan perhitungan simbolis sekarang.

```
>A &= [1,0]; B &= [0,1]; C &= [2,2]; // menentukan tiga titik A, B, C
```

Fungsi untuk garis dan lingkaran bekerja seperti fungsi Euler, tetapi memberikan perhitungan simbolis.

```
>c &= lineThrough(B,C) // c=BC
```

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

Kita bisa mendapatkan persamaan garis dengan mudah.

```
>$getLineEquation(c,x,y), $solve(%,y) | expand // persamaan garis c
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$getLineEquation(c,x,y), $solve(%,y) | expand // persamaan gar ...  
^
```

```
>$getLineEquation(lineThrough(A,[x1,y1]),x,y) // persamaan garis melalui A dan (x1, y1)  
>h &= perpendicular(A,lineThrough(B,C)) // h melalui A tegak lurus BC
```

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

```
>Q &= lineIntersection(c,h) // Q titik potong garis c=BC dan h
```

```
Maxima said:  
rat: replaced 9.983250083613754e-5 by 612914/6139423483 = 9.983250083613756e-5  
  
rat: replaced 3.986533601775671e-4 by 220554/553247563 = 3.986533601775666e-4  
  
rat: replaced 8.954327045205754e-4 by 584699/652979277 = 8.954327045205756e-4  
  
rat: replaced 0.001589120864678328 by 740868/466212493 = 0.00158912086467833  
  
rat: replaced 0.002478648480745763 by 878917/354595259 = 0.002478648480745762  
  
rat: replaced 0.003562926609036218 by 2735717/767828614 = 0.003562926609036219
```

rat: replaced 0.004840846830973591 by 1164348/240525685 = 0.004840846830973582  
rat: replaced 0.006311281363933816 by 16515210/2616776063 = 0.006311281363933816  
rat: replaced 0.007973083174022497 by 2414321/302808957 = 0.007973083174022491  
rat: replaced 0.009825086090776508 by 1144049/116441626 = 0.009825086090776506  
rat: replaced 0.01186610492378118 by 1659683/139867548 = 0.01186610492378118  
rat: replaced 0.01409493558118687 by 986877/70016425 = 0.01409493558118684  
rat: replaced 0.01651035519011868 by 1738361/105289134 = 0.01651035519011867  
rat: replaced 0.01911112221896202 by 1475047/77182647 = 0.01911112221896199  
rat: replaced 0.02189597660151474 by 7711274/352177669 = 0.02189597660151473  
rat: replaced 0.02486363986299212 by 3887839/156366446 = 0.02486363986299209  
rat: replaced 0.0280128152478745 by 2263313/80795628 = 0.02801281524787455  
rat: replaced 0.03134218784958129 by 1116362/35618509 = 0.03134218784958124  
rat: replaced 0.03485042474195996 by 3920507/112495243 = 0.03485042474195998  
rat: replaced 0.03853617511257795 by 5379408/139593719 = 0.03853617511257795  
rat: replaced 0.04239807039780302 by 3385918/79860191 = 0.04239807039780308  
rat: replaced 0.04643472441965829 by 10918553/235137672 = 0.04643472441965828  
rat: replaced 0.05064473352443885 by 5036501/99447675 = 0.05064473352443886

rat: replaced 0.05502667672307548 by 2932521/53292715 = 0.05502667672307557  
rat: replaced 0.05957911583323347 by 6320819/106091185 = 0.05957911583323346  
rat: replaced 0.06430059562312868 by 9893260/153859539 = 0.0643005956231287  
rat: replaced 0.06918964395705007 by 6012189/86894348 = 0.06918964395705  
rat: replaced 0.07424477194257195 by 6096479/82113243 = 0.07424477194257204  
rat: replaced 0.07946447407944118 by 5389689/67825139 = 0.07946447407944125  
rat: replaced 0.0848472284101276 by 9595393/113090235 = 0.08484722841012754  
rat: replaced 0.09039149667201674 by 3773144/41742245 = 0.09039149667201657  
rat: replaced 0.0960957244512361 by 5162056/53717853 = 0.09609572445123597  
rat: replaced 0.1019583413380946 by 1082663/10618680 = 0.1019583413380948  
rat: replaced 0.107977761084122 by 1922059/17800508 = 0.1079777610841219  
rat: replaced 0.1141523817606936 by 5923297/51889386 = 0.1141523817606938  
rat: replaced 0.1204805859192203 by 17634703/146369665 = 0.1204805859192204  
rat: replaced 0.1269607407528933 by 11368220/89541223 = 0.1269607407528932  
rat: replaced 0.1335911982599624 by 4657902/34866833 = 0.1335911982599624  
rat: replaced 0.1403702954085355 by 8528456/60756843 = 0.1403702954085353  
rat: replaced 0.1472963543028805 by 11128453/75551449 = 0.1472963543028804  
rat: replaced 0.1543676823512128 by 8170760/52930509 = 0.1543676823512126

rat: replaced 0.1615825724349539 by 188109817/1164171446 = 0.1615825724349539  
rat: replaced 0.1689393030794406 by 5046974/29874481 = 0.1689393030794409  
rat: replaced 0.1764361386260728 by 6530305/37012287 = 0.176436138626073  
rat: replaced 0.1840713294058766 by 25189859/136848357 = 0.1840713294058766  
rat: replaced 0.1918431119144694 by 24326967/126806570 = 0.1918431119144694  
rat: replaced 0.1997497089884105 by 14902039/74603558 = 0.1997497089884104  
rat: replaced 0.2077893299829148 by 7281351/35041987 = 0.2077893299829145  
rat: replaced 0.2159601709509153 by 11348921/52550991 = 0.2159601709509151  
rat: replaced 0.2242604148234577 by 22385730/99820247 = 0.2242604148234576  
rat: replaced 0.2326882315914051 by 25615030/110083049 = 0.2326882315914051  
rat: replaced 0.2412417784884371 by 14523232/60201977 = 0.2412417784884373  
rat: replaced 0.2499192001753251 by 11309023/45250717 = 0.2499192001753254  
rat: replaced 0.2587186289254649 by 7582961/29309683 = 0.2587186289254647  
rat: replaced 0.267638184811648 by 17912865/66929407 = 0.2676381848116479  
rat: replaced 0.2766759758940514 by 27538925/99534934 = 0.2766759758940514  
rat: replaced 0.2858300984094321 by 29258587/102363562 = 0.2858300984094321  
rat: replaced 0.2950986369614998 by 7877677/26695064 = 0.2950986369614997

rat: replaced 0.304479664712457 by  $14469542/47522195 = 0.304479664712457$   
rat: replaced 0.3139712435756791 by  $8375733/26676752 = 0.3139712435756797$   
rat: replaced 0.3235714244095225 by  $178371467/551258404 = 0.3235714244095225$   
rat: replaced 0.3332782472122374 by  $5743591/17233621 = 0.333278247212237$   
rat: replaced 0.3430897413179662 by  $15588245/45434891 = 0.3430897413179664$   
rat: replaced 0.3530039255938071 by  $6523425/18479752 = 0.3530039255938067$   
rat: replaced 0.3630188086379282 by  $51253958/141188161 = 0.3630188086379282$   
rat: replaced 0.373132388978704 by  $9370061/25111894 = 0.3731323889787047$   
rat: replaced 0.3833426552748616 by  $11820697/30835851 = 0.3833426552748617$   
rat: replaced 0.393647586516613 by  $9153768/23253713 = 0.3936475865166135$   
rat: replaced 0.4040451522277552 by  $16634707/41170416 = 0.404045152227755$   
rat: replaced 0.4145333126687146 by  $2088920/5039209 = 0.4145333126687145$   
rat: replaced 0.4251100190405208 by  $24667763/58026774 = 0.4251100190405209$   
rat: replaced 0.4357732136896836 by  $10448574/23977091 = 0.435773213689684$   
rat: replaced 0.4465208303139576 by  $8346266/18691773 = 0.4465208303139568$   
rat: replaced 0.4573507941689697 by  $20158688/44077081 = 0.4573507941689696$   
rat: replaced 0.4682610222756929 by  $12818601/27374905 = 0.4682610222756937$   
rat: replaced 0.4792494236287415 by  $13652513/28487281 = 0.4792494236287416$

rat: replaced 0.4903138994054704 by  $35114711/71616797 = 0.4903138994054705$   
rat: replaced 0.5014523431758559 by  $15102855/30118226 = 0.5014523431758564$   
rat: replaced 0.5126626411131362 by  $31697340/61828847 = 0.5126626411131361$   
rat: replaced 0.5239426722051925 by  $27432767/52358337 = 0.5239426722051924$   
rat: replaced 0.5352903084666492 by  $6124470/11441399 = 0.5352903084666482$   
rat: replaced 0.5467034151516694 by  $41717397/76307182 = 0.5467034151516694$   
rat: replaced 0.5581798509674292 by  $7494380/13426461 = 0.5581798509674292$   
rat: replaced 0.5697174682882435 by  $14609183/25642856 = 0.5697174682882438$   
rat: replaced 0.581314113370329 by  $14367580/24715691 = 0.5813141133703282$   
rat: replaced 0.5929676265671738 by  $9820294/16561265 = 0.5929676265671735$   
rat: replaced 0.6046758425455033 by  $23593213/39017952 = 0.6046758425455031$   
rat: replaced 0.6164365905018095 by  $15720181/25501700 = 0.6164365905018097$   
rat: replaced 0.6282476943794307 by  $53974636/85912987 = 0.6282476943794306$   
rat: replaced 0.640106973086155 by  $20459615/31962806 = 0.6401069730861552$   
rat: replaced 0.652012240712328 by  $51645100/79208789 = 0.652012240712328$   
rat: replaced 0.6639613067494411 by  $12215999/18398661 = 0.6639613067494422$   
rat: replaced 0.6759519763091814 by  $18558734/27455699 = 0.6759519763091808$

rat: replaced 0.6879820503429186 by  $23500536/34158647 = 0.687982050342919$   
rat: replaced 0.7000493258616074 by  $29992669/42843651 = 0.7000493258616078$   
rat: replaced 0.7121515961560857 by  $10685401/15004391 = 0.7121515961560853$   
rat: replaced 0.7242866510177421 by  $11795807/16286103 = 0.7242866510177419$   
rat: replaced 0.7364522769595366 by  $14940657/20287339 = 0.7364522769595362$   
rat: replaced 0.7486462574373463 by  $42508133/56779998 = 0.7486462574373461$   
rat: replaced 5.033291500140813e-5 by  $263336/5231884543 = 5.033291500140813e-5$   
rat: replaced 2.026599467560841e-4 by  $407727/2011877564 = 2.02659946756084e-4$   
rat: replaced 4.589658460211338e-4 by  $352373/767754296 = 4.589658460211339e-4$   
rat: replaced 8.21224965753764e-4 by  $219501/267284860 = 8.212249657537654e-4$   
rat: replaced 0.001291401063677061 by  $174589/135193477 = 0.001291401063677059$   
rat: replaced 0.001871447105906615 by  $1078337/576204904 = 0.001871447105906617$   
rat: replaced 0.002563305071654892 by  $1323915/516487489 = 0.002563305071654891$   
rat: replaced 0.003368905759035173 by  $820537/243561874 = 0.003368905759035176$   
rat: replaced 0.00429016859198364 by  $7572857/1765165363 = 0.00429016859198364$   
rat: replaced 0.005329001428317881 by  $3020890/566877311 = 0.005329001428317882$   
rat: replaced 0.006487300368953564 by  $2580732/397812935 = 0.006487300368953564$   
rat: replaced 0.007766949568295017 by  $1049181/135082762 = 0.007766949568295028$

rat: replaced 0.009169821045822202 by 2408608/262666849 = 0.009169821045822193  
rat: replaced 0.01069777449888981 by 2325322/217365023 = 0.01069777449888982  
rat: replaced 0.01235265711675931 by 7449711/603085711 = 0.01235265711675931  
rat: replaced 0.01413630339588112 by 3774568/267012379 = 0.01413630339588113  
rat: replaced 0.0160505349564472 by 2619104/163178611 = 0.0160505349564472  
rat: replaced 0.01809716036023018 by 3107690/171722521 = 0.01809716036023021  
rat: replaced 0.02027797492972855 by 6791343/334912289 = 0.02027797492972854  
rat: replaced 0.02259476056863596 by 2685790/118867823 = 0.02259476056863597  
rat: replaced 0.0250492855836526 by 2956693/118035023 = 0.02504928558365258  
rat: replaced 0.02764330450765584 by 2138111/77346433 = 0.02764330450765583  
rat: replaced 0.03037855792424843 by 1678577/55255322 = 0.03037855792424846  
rat: replaced 0.03325677229370128 by 1488397/44754704 = 0.03325677229370124  
rat: replaced 0.03627965978030939 by 3229091/89005548 = 0.03627965978030943  
rat: replaced 0.03944891808117656 by 6094420/154488901 = 0.03944891808117659  
rat: replaced 0.04276623025644721 by 206826/4836199 = 0.04276623025644726  
rat: replaced 0.04623326456100163 by 7175941/155211644 = 0.04623326456100162  
rat: replaced 0.04985167427763171 by 2856261/57295187 = 0.04985167427763173

rat: replaced 0.0536230975517149 by  $8075629/150599823 = 0.05362309755171492$   
rat: replaced 0.05754915722739962 by  $12314906/213989337 = 0.05754915722739961$   
rat: replaced 0.06163146068532366 by  $10145753/164619707 = 0.06163146068532366$   
rat: replaced 0.06587159968187639 by  $5154956/78257641 = 0.06587159968187643$   
rat: replaced 0.07027115019002506 by  $3189686/45391117 = 0.07027115019002507$   
rat: replaced 0.07483167224171838 by  $4757796/63579977 = 0.0748316722417185$   
rat: replaced 0.07955470977188528 by  $7059961/88743470 = 0.07955470977188518$   
rat: replaced 0.08444179046404166 by  $17285418/204702173 = 0.08444179046404163$   
rat: replaced 0.08949442559752452 by  $6119169/68374862 = 0.08949442559752442$   
rat: replaced 0.0947141098963642 by  $2739857/28927654 = 0.09471410989636422$   
rat: replaced 0.100102321379814 by  $21380147/213582929 = 0.100102321379814$   
rat: replaced 0.1056605212145493 by  $8628153/81659194 = 0.1056605212145493$   
rat: replaced 0.1113901535685515 by  $4925969/44222661 = 0.1113901535685517$   
rat: replaced 0.1172926454666934 by  $7052303/60125705 = 0.1172926454666935$   
rat: replaced 0.1233694066480375 by  $17851649/144700777 = 0.1233694066480376$   
rat: replaced 0.1296218294248629 by  $13037238/100579031 = 0.1296218294248629$   
rat: replaced 0.1360512885434353 by  $20468361/150445918 = 0.1360512885434353$   
rat: replaced 0.1426591410465347 by  $8451499/59242604 = 0.1426591410465347$

```
rat: replaced 0.1494467261377502 by 40350618/270000013 = 0.1494467261377502
rat: replaced 0.1564153650475627 by 30759845/196654881 = 0.1564153650475627
rat: replaced 0.1635663609012215 by 11970848/73186491 = 0.1635663609012215
rat: replaced 0.1709009985884339 by 3726835/21806982 = 0.1709009985884337
rat: replaced 0.1784205446348769 by 7050541/39516419 = 0.178420544634877
rat: replaced 0.1861262470755453 by 7913431/42516470 = 0.1861262470755451
rat: replaced 0.1940193353299499 by 15356416/79148895 = 0.19401933532995
rat: replaced 0.2021010200791761 by 21517868/106470853 = 0.202101020079176
rat: replaced 0.2103724931448173 by 10133132/48167571 = 0.2103724931448173
rat: replaced 0.2188349273697929 by 14393696/65774217 = 0.2188349273697929
rat: replaced 0.2274894765010662 by 2362445/10384854 = 0.2274894765010659
rat: replaced 0.2363372750742693 by 14238388/60246053 = 0.2363372750742692
rat: replaced 0.2453794383002513 by 11843947/48267887 = 0.2453794383002513
rat: replaced 0.2546170619535583 by 10437767/40993981 = 0.2546170619535585
rat: replaced 0.2640512222628563 by 18572095/70335198 = 0.2640512222628562
rat: replaced 0.2736829758033094 by 25733021/94024924 = 0.2736829758033094
rat: replaced 0.2835133593909236 by 5354031/18884581 = 0.2835133593909232
```

rat: replaced 0.2935433899788653 by  $33562265/114334937 = 0.2935433899788654$   
rat: replaced 0.3037740645557676 by  $12785981/42090430 = 0.3037740645557672$   
rat: replaced 0.3142063600460319 by  $13879096/44171913 = 0.314206360046032$   
rat: replaced 0.3248412332121354 by  $13048490/40168823 = 0.3248412332121357$   
rat: replaced 0.3356796205589581 by  $12520681/37299497 = 0.3356796205589582$   
rat: replaced 0.3467224382401299 by  $27133151/78256115 = 0.3467224382401299$   
rat: replaced 0.3579705819664191 by  $32019579/89447515 = 0.3579705819664191$   
rat: replaced 0.3694249269161592 by  $12845283/34771024 = 0.3694249269161587$   
rat: replaced 0.3810863276477343 by  $12790304/33562747 = 0.381086327647734$   
rat: replaced 0.3929556180141225 by  $27557157/70127912 = 0.3929556180141225$   
rat: replaced 0.4050336110795114 by  $12582391/31065054 = 0.4050336110795107$   
rat: replaced 0.4173210990379927 by  $17616979/42214446 = 0.4173210990379928$   
rat: replaced 0.4298188531343438 by  $28764336/66921997 = 0.4298188531343439$   
rat: replaced 0.4425276235869029 by  $52612738/118891421 = 0.4425276235869029$   
rat: replaced 0.4554481395125489 by  $12438812/27311149 = 0.4554481395125485$   
rat: replaced 0.4685811088537897 by  $12910499/27552325 = 0.46858110885379$   
rat: replaced 0.4819272183079686 by  $11623658/24119115 = 0.4819272183079686$   
rat: replaced 0.4954871332585954 by  $40137729/81006602 = 0.4954871332585954$

```
rat: replaced 0.5092614977088081 by 27060617/53136978 = 0.5092614977088084  
rat: replaced 0.523250934216974 by 57357723/109618004 = 0.5232509342169741  
rat: replaced 0.5374560438344332 by 19984722/37183919 = 0.5374560438344328  
rat: replaced 0.551877406045395 by 10637804/19275665 = 0.5518774060453946  
rat: replaced 0.5665155787089895 by 22241852/39260795 = 0.5665155787089895  
rat: replaced 0.5813710980034821 by 10844268/18652919 = 0.5813710980034814  
rat: replaced 0.5964444783726564 by 13079224/21928653 = 0.596444478372657  
rat: replaced 0.6117362124743696 by 11199699/18308053 = 0.6117362124743685  
rat: replaced 0.6272467711312885 by 11338738/18076997 = 0.6272467711312891  
rat: replaced 0.6429766032838061 by 10161473/15803799 = 0.6429766032838053  
rat: replaced 0.6589261359451484 by 11120191/16876233 = 0.6589261359451484  
rat: replaced 0.6750957741586742 by 11234073/16640710 = 0.6750957741586747  
rat: replaced 0.6914859009573701 by 9571673/13842181 = 0.6914859009573708  
rat: replaced 0.7080968773255479 by 20218829/28553761 = 0.7080968773255474  
rat: replaced 0.7249290421627467 by 10945526/15098755 = 0.7249290421627479  
rat: replaced 0.7419827122498429 by 23520179/31699093 = 0.7419827122498426  
rat: replaced 0.7592581822173726 by 16709871/22008154 = 0.7592581822173727  
part: invalid index of list or matrix.
```

```
#0: lineIntersection(g=[-1,2,2],h=[2,1,2])
-- an error. To debug this try: debugmode(true);

Error in:
... ersection(c,h) // Q titik potong garis c=BC dan h ...
^
```

```
>$projectToLine(A,lineThrough(B,C)) // proyeksi A pada BC
```

```
Maxima said:
rat: replaced 5.033291500140813e-5 by 263336/5231884543 = 5.033291500140813e-5

rat: replaced 2.026599467560841e-4 by 407727/2011877564 = 2.02659946756084e-4

rat: replaced 4.589658460211338e-4 by 352373/767754296 = 4.589658460211339e-4

rat: replaced 8.21224965753764e-4 by 219501/267284860 = 8.212249657537654e-4

rat: replaced 0.001291401063677061 by 174589/135193477 = 0.001291401063677059

rat: replaced 0.001871447105906615 by 1078337/576204904 = 0.001871447105906617

rat: replaced 0.002563305071654892 by 1323915/516487489 = 0.002563305071654891

rat: replaced 0.003368905759035173 by 820537/243561874 = 0.003368905759035176

rat: replaced 0.00429016859198364 by 7572857/1765165363 = 0.00429016859198364

rat: replaced 0.005329001428317881 by 3020890/566877311 = 0.005329001428317882

rat: replaced 0.006487300368953564 by 2580732/397812935 = 0.006487300368953564
```

rat: replaced 0.007766949568295017 by  $1049181/135082762 = 0.007766949568295028$   
rat: replaced 0.009169821045822202 by  $2408608/262666849 = 0.009169821045822193$   
rat: replaced 0.01069777449888981 by  $2325322/217365023 = 0.01069777449888982$   
rat: replaced 0.01235265711675931 by  $7449711/603085711 = 0.01235265711675931$   
rat: replaced 0.01413630339588112 by  $3774568/267012379 = 0.01413630339588113$   
rat: replaced 0.0160505349564472 by  $2619104/163178611 = 0.0160505349564472$   
rat: replaced 0.01809716036023018 by  $3107690/171722521 = 0.01809716036023021$   
rat: replaced 0.02027797492972855 by  $6791343/334912289 = 0.02027797492972854$   
rat: replaced 0.02259476056863596 by  $2685790/118867823 = 0.02259476056863597$   
rat: replaced 0.0250492855836526 by  $2956693/118035023 = 0.02504928558365258$   
rat: replaced 0.02764330450765584 by  $2138111/77346433 = 0.02764330450765583$   
rat: replaced 0.03037855792424843 by  $1678577/55255322 = 0.03037855792424846$   
rat: replaced 0.03325677229370128 by  $1488397/44754704 = 0.03325677229370124$   
rat: replaced 0.03627965978030939 by  $3229091/89005548 = 0.03627965978030943$   
rat: replaced 0.03944891808117656 by  $6094420/154488901 = 0.03944891808117659$   
rat: replaced 0.04276623025644721 by  $206826/4836199 = 0.04276623025644726$   
rat: replaced 0.04623326456100163 by  $7175941/155211644 = 0.04623326456100162$   
rat: replaced 0.04985167427763171 by  $2856261/57295187 = 0.04985167427763173$

rat: replaced 0.0536230975517149 by 8075629/150599823 = 0.05362309755171492  
rat: replaced 0.05754915722739962 by 12314906/213989337 = 0.05754915722739961  
rat: replaced 0.06163146068532366 by 10145753/164619707 = 0.06163146068532366  
rat: replaced 0.06587159968187639 by 5154956/78257641 = 0.06587159968187643  
rat: replaced 0.07027115019002506 by 3189686/45391117 = 0.07027115019002507  
rat: replaced 0.07483167224171838 by 4757796/63579977 = 0.0748316722417185  
rat: replaced 0.07955470977188528 by 7059961/88743470 = 0.07955470977188518  
rat: replaced 0.08444179046404166 by 17285418/204702173 = 0.08444179046404163  
rat: replaced 0.08949442559752452 by 6119169/68374862 = 0.08949442559752442  
rat: replaced 0.0947141098963642 by 2739857/28927654 = 0.09471410989636422  
rat: replaced 0.100102321379814 by 21380147/213582929 = 0.100102321379814  
rat: replaced 0.1056605212145493 by 8628153/81659194 = 0.1056605212145493  
rat: replaced 0.1113901535685515 by 4925969/44222661 = 0.1113901535685517  
rat: replaced 0.1172926454666934 by 7052303/60125705 = 0.1172926454666935  
rat: replaced 0.1233694066480375 by 17851649/144700777 = 0.1233694066480376  
rat: replaced 0.1296218294248629 by 13037238/100579031 = 0.1296218294248629  
rat: replaced 0.1360512885434353 by 20468361/150445918 = 0.1360512885434353

```
rat: replaced 0.1426591410465347 by 8451499/59242604 = 0.1426591410465347
rat: replaced 0.1494467261377502 by 40350618/270000013 = 0.1494467261377502
rat: replaced 0.1564153650475627 by 30759845/196654881 = 0.1564153650475627
rat: replaced 0.1635663609012215 by 11970848/73186491 = 0.1635663609012215
rat: replaced 0.1709009985884339 by 3726835/21806982 = 0.1709009985884337
rat: replaced 0.1784205446348769 by 7050541/39516419 = 0.178420544634877
rat: replaced 0.1861262470755453 by 7913431/42516470 = 0.1861262470755451
rat: replaced 0.1940193353299499 by 15356416/79148895 = 0.19401933532995
rat: replaced 0.2021010200791761 by 21517868/106470853 = 0.202101020079176
rat: replaced 0.2103724931448173 by 10133132/48167571 = 0.2103724931448173
rat: replaced 0.2188349273697929 by 14393696/65774217 = 0.2188349273697929
rat: replaced 0.2274894765010662 by 2362445/10384854 = 0.2274894765010659
rat: replaced 0.2363372750742693 by 14238388/60246053 = 0.2363372750742692
rat: replaced 0.2453794383002513 by 11843947/48267887 = 0.2453794383002513
rat: replaced 0.2546170619535583 by 10437767/40993981 = 0.2546170619535585
rat: replaced 0.2640512222628563 by 18572095/70335198 = 0.2640512222628562
rat: replaced 0.2736829758033094 by 25733021/94024924 = 0.2736829758033094
rat: replaced 0.2835133593909236 by 5354031/18884581 = 0.2835133593909232
```

```
rat: replaced 0.2935433899788653 by 33562265/114334937 = 0.2935433899788654
rat: replaced 0.3037740645557676 by 12785981/42090430 = 0.3037740645557672
rat: replaced 0.3142063600460319 by 13879096/44171913 = 0.314206360046032
rat: replaced 0.3248412332121354 by 13048490/40168823 = 0.3248412332121357
rat: replaced 0.3356796205589581 by 12520681/37299497 = 0.3356796205589582
rat: replaced 0.3467224382401299 by 27133151/78256115 = 0.3467224382401299
rat: replaced 0.3579705819664191 by 32019579/89447515 = 0.3579705819664191
rat: replaced 0.3694249269161592 by 12845283/34771024 = 0.3694249269161587
rat: replaced 0.3810863276477343 by 12790304/33562747 = 0.381086327647734
rat: replaced 0.3929556180141225 by 27557157/70127912 = 0.3929556180141225
rat: replaced 0.4050336110795114 by 12582391/31065054 = 0.4050336110795107
rat: replaced 0.4173210990379927 by 17616979/42214446 = 0.4173210990379928
rat: replaced 0.4298188531343438 by 28764336/66921997 = 0.4298188531343439
rat: replaced 0.4425276235869029 by 52612738/118891421 = 0.4425276235869029
rat: replaced 0.4554481395125489 by 12438812/27311149 = 0.4554481395125485
rat: replaced 0.4685811088537897 by 12910499/27552325 = 0.46858110885379
rat: replaced 0.4819272183079686 by 11623658/24119115 = 0.4819272183079686
```

rat: replaced 0.4954871332585954 by 40137729/81006602 = 0.4954871332585954  
rat: replaced 0.5092614977088081 by 27060617/53136978 = 0.5092614977088084  
rat: replaced 0.523250934216974 by 57357723/109618004 = 0.5232509342169741  
rat: replaced 0.5374560438344332 by 19984722/37183919 = 0.5374560438344328  
rat: replaced 0.551877406045395 by 10637804/19275665 = 0.5518774060453946  
rat: replaced 0.5665155787089895 by 22241852/39260795 = 0.5665155787089895  
rat: replaced 0.5813710980034821 by 10844268/18652919 = 0.5813710980034814  
rat: replaced 0.5964444783726564 by 13079224/21928653 = 0.596444478372657  
rat: replaced 0.6117362124743696 by 11199699/18308053 = 0.6117362124743685  
rat: replaced 0.6272467711312885 by 11338738/18076997 = 0.6272467711312891  
rat: replaced 0.6429766032838061 by 10161473/15803799 = 0.6429766032838053  
rat: replaced 0.6589261359451484 by 11120191/16876233 = 0.6589261359451484  
rat: replaced 0.6750957741586742 by 11234073/16640710 = 0.6750957741586747  
rat: replaced 0.6914859009573701 by 9571673/13842181 = 0.6914859009573708  
rat: replaced 0.7080968773255479 by 20218829/28553761 = 0.7080968773255474  
rat: replaced 0.7249290421627467 by 10945526/15098755 = 0.7249290421627479  
rat: replaced 0.7419827122498429 by 23520179/31699093 = 0.7419827122498426  
rat: replaced 0.7592581822173726 by 16709871/22008154 = 0.7592581822173727

rat: replaced  $9.983250083613754e-5$  by  $612914/6139423483 = 9.983250083613756e-5$   
rat: replaced  $3.986533601775671e-4$  by  $220554/553247563 = 3.986533601775666e-4$   
rat: replaced  $8.954327045205754e-4$  by  $584699/652979277 = 8.954327045205756e-4$   
rat: replaced  $0.001589120864678328$  by  $740868/466212493 = 0.00158912086467833$   
rat: replaced  $0.002478648480745763$  by  $878917/354595259 = 0.002478648480745762$   
rat: replaced  $0.003562926609036218$  by  $2735717/767828614 = 0.003562926609036219$   
rat: replaced  $0.004840846830973591$  by  $1164348/240525685 = 0.004840846830973582$   
rat: replaced  $0.006311281363933816$  by  $16515210/2616776063 = 0.006311281363933816$   
rat: replaced  $0.007973083174022497$  by  $2414321/302808957 = 0.007973083174022491$   
rat: replaced  $0.009825086090776508$  by  $1144049/116441626 = 0.009825086090776506$   
rat: replaced  $0.01186610492378118$  by  $1659683/139867548 = 0.01186610492378118$   
rat: replaced  $0.01409493558118687$  by  $986877/70016425 = 0.01409493558118684$   
rat: replaced  $0.01651035519011868$  by  $1738361/105289134 = 0.01651035519011867$   
rat: replaced  $0.01911112221896202$  by  $1475047/77182647 = 0.01911112221896199$   
rat: replaced  $0.02189597660151474$  by  $7711274/352177669 = 0.02189597660151473$   
rat: replaced  $0.02486363986299212$  by  $3887839/156366446 = 0.02486363986299209$   
rat: replaced  $0.0280128152478745$  by  $2263313/80795628 = 0.02801281524787455$

rat: replaced 0.03134218784958129 by 1116362/35618509 = 0.03134218784958124  
rat: replaced 0.03485042474195996 by 3920507/112495243 = 0.03485042474195998  
rat: replaced 0.03853617511257795 by 5379408/139593719 = 0.03853617511257795  
rat: replaced 0.04239807039780302 by 3385918/79860191 = 0.04239807039780308  
rat: replaced 0.04643472441965829 by 10918553/235137672 = 0.04643472441965828  
rat: replaced 0.05064473352443885 by 5036501/99447675 = 0.05064473352443886  
rat: replaced 0.05502667672307548 by 2932521/53292715 = 0.05502667672307557  
rat: replaced 0.05957911583323347 by 6320819/106091185 = 0.05957911583323346  
rat: replaced 0.06430059562312868 by 9893260/153859539 = 0.0643005956231287  
rat: replaced 0.06918964395705007 by 6012189/86894348 = 0.06918964395705  
rat: replaced 0.07424477194257195 by 6096479/82113243 = 0.07424477194257204  
rat: replaced 0.07946447407944118 by 5389689/67825139 = 0.07946447407944125  
rat: replaced 0.0848472284101276 by 9595393/113090235 = 0.08484722841012754  
rat: replaced 0.09039149667201674 by 3773144/41742245 = 0.09039149667201657  
rat: replaced 0.0960957244512361 by 5162056/53717853 = 0.09609572445123597  
rat: replaced 0.1019583413380946 by 1082663/10618680 = 0.1019583413380948  
rat: replaced 0.107977761084122 by 1922059/17800508 = 0.1079777610841219  
rat: replaced 0.1141523817606936 by 5923297/51889386 = 0.1141523817606938

```
rat: replaced 0.1204805859192203 by 17634703/146369665 = 0.1204805859192204
rat: replaced 0.1269607407528933 by 11368220/89541223 = 0.1269607407528932
rat: replaced 0.1335911982599624 by 4657902/34866833 = 0.1335911982599624
rat: replaced 0.1403702954085355 by 8528456/60756843 = 0.1403702954085353
rat: replaced 0.1472963543028805 by 11128453/75551449 = 0.1472963543028804
rat: replaced 0.1543676823512128 by 8170760/52930509 = 0.1543676823512126
rat: replaced 0.1615825724349539 by 188109817/1164171446 = 0.1615825724349539
rat: replaced 0.1689393030794406 by 5046974/29874481 = 0.1689393030794409
rat: replaced 0.1764361386260728 by 6530305/37012287 = 0.176436138626073
rat: replaced 0.1840713294058766 by 25189859/136848357 = 0.1840713294058766
rat: replaced 0.1918431119144694 by 24326967/126806570 = 0.1918431119144694
rat: replaced 0.1997497089884105 by 14902039/74603558 = 0.1997497089884104
rat: replaced 0.2077893299829148 by 7281351/35041987 = 0.2077893299829145
rat: replaced 0.2159601709509153 by 11348921/52550991 = 0.2159601709509151
rat: replaced 0.2242604148234577 by 22385730/99820247 = 0.2242604148234576
rat: replaced 0.2326882315914051 by 25615030/110083049 = 0.2326882315914051
rat: replaced 0.2412417784884371 by 14523232/60201977 = 0.2412417784884373
```

```
rat: replaced 0.2499192001753251 by 11309023/45250717 = 0.2499192001753254
rat: replaced 0.2587186289254649 by 7582961/29309683 = 0.2587186289254647
rat: replaced 0.267638184811648 by 17912865/66929407 = 0.2676381848116479
rat: replaced 0.2766759758940514 by 27538925/99534934 = 0.2766759758940514
rat: replaced 0.2858300984094321 by 29258587/102363562 = 0.2858300984094321
rat: replaced 0.2950986369614998 by 7877677/26695064 = 0.2950986369614997
rat: replaced 0.304479664712457 by 14469542/47522195 = 0.304479664712457
rat: replaced 0.3139712435756791 by 8375733/26676752 = 0.3139712435756797
rat: replaced 0.3235714244095225 by 178371467/551258404 = 0.3235714244095225
rat: replaced 0.3332782472122374 by 5743591/17233621 = 0.333278247212237
rat: replaced 0.3430897413179662 by 15588245/45434891 = 0.3430897413179664
rat: replaced 0.3530039255938071 by 6523425/18479752 = 0.3530039255938067
rat: replaced 0.3630188086379282 by 51253958/141188161 = 0.3630188086379282
rat: replaced 0.373132388978704 by 9370061/25111894 = 0.3731323889787047
rat: replaced 0.3833426552748616 by 11820697/30835851 = 0.3833426552748617
rat: replaced 0.393647586516613 by 9153768/23253713 = 0.3936475865166135
rat: replaced 0.4040451522277552 by 16634707/41170416 = 0.404045152227755
rat: replaced 0.4145333126687146 by 2088920/5039209 = 0.4145333126687145
```

rat: replaced 0.4251100190405208 by 24667763/58026774 = 0.4251100190405209  
rat: replaced 0.4357732136896836 by 10448574/23977091 = 0.435773213689684  
rat: replaced 0.4465208303139576 by 8346266/18691773 = 0.4465208303139568  
rat: replaced 0.4573507941689697 by 20158688/44077081 = 0.4573507941689696  
rat: replaced 0.4682610222756929 by 12818601/27374905 = 0.4682610222756937  
rat: replaced 0.4792494236287415 by 13652513/28487281 = 0.4792494236287416  
rat: replaced 0.4903138994054704 by 35114711/71616797 = 0.4903138994054705  
rat: replaced 0.5014523431758559 by 15102855/30118226 = 0.5014523431758564  
rat: replaced 0.5126626411131362 by 31697340/61828847 = 0.5126626411131361  
rat: replaced 0.5239426722051925 by 27432767/52358337 = 0.5239426722051924  
rat: replaced 0.5352903084666492 by 6124470/11441399 = 0.5352903084666482  
rat: replaced 0.5467034151516694 by 41717397/76307182 = 0.5467034151516694  
rat: replaced 0.5581798509674292 by 7494380/13426461 = 0.5581798509674292  
rat: replaced 0.5697174682882435 by 14609183/25642856 = 0.5697174682882438  
rat: replaced 0.581314113370329 by 14367580/24715691 = 0.5813141133703282  
rat: replaced 0.5929676265671738 by 9820294/16561265 = 0.5929676265671735  
rat: replaced 0.6046758425455033 by 23593213/39017952 = 0.6046758425455031

```
rat: replaced 0.6164365905018095 by 15720181/25501700 = 0.6164365905018097
rat: replaced 0.6282476943794307 by 53974636/85912987 = 0.6282476943794306
rat: replaced 0.640106973086155 by 20459615/31962806 = 0.6401069730861552
rat: replaced 0.652012240712328 by 51645100/79208789 = 0.652012240712328
rat: replaced 0.6639613067494411 by 12215999/18398661 = 0.6639613067494422
rat: replaced 0.6759519763091814 by 18558734/27455699 = 0.6759519763091808
rat: replaced 0.6879820503429186 by 23500536/34158647 = 0.687982050342919
rat: replaced 0.7000493258616074 by 29992669/42843651 = 0.7000493258616078
rat: replaced 0.7121515961560857 by 10685401/15004391 = 0.7121515961560853
rat: replaced 0.7242866510177421 by 11795807/16286103 = 0.7242866510177419
rat: replaced 0.7364522769595366 by 14940657/20287339 = 0.7364522769595362
rat: replaced 0.7486462574373463 by 42508133/56779998 = 0.7486462574373461
part: invalid index of list or matrix.
#0: lineIntersection(g=[2,1,2],h=[-1,2,2])
#1: projectToLine(a=[1,0],g=[-1,2,2])
-- an error. To debug this try: debugmode(true);

Error in:
$projectToLine(A,lineThrough(B,C)) // proyeksi A pada BC ...
^
```

```
>$distance(A,Q) // jarak AQ  
>cc &= circleThrough(A,B,C); $cc // (titik pusat dan jari-jari) lingkaran melalui A, B, C
```

Maxima said:

```
rat: replaced -4.98329175014009e-5 by -86001/1725786976 = -4.983291750140082e-5  
  
rat: replaced -1.986600267553235e-4 by -1133306/5704751069 = -1.986600267553234e-4  
  
rat: replaced -4.454664535081185e-4 by -474290/1064704191 = -4.454664535081181e-4  
  
rat: replaced -7.892275256562442e-4 by -1190199/1508055613 = -7.892275256562439e-4  
  
rat: replaced -0.001228908875712045 by -259907/211494119 = -0.001228908875712047  
  
rat: replaced -0.001763466544240408 by -5854594/3319934829 = -0.001763466544240408  
  
rat: replaced -0.002391847084253176 by -866601/362314550 = -0.002391847084253172  
  
rat: replaced -0.003112987666553255 by -5049204/1621980085 = -0.003112987666553255  
  
rat: replaced -0.00392581618601677 by -1241039/316122544 = -0.003925816186016774  
  
rat: replaced -0.004829251368802329 by -3015690/624463249 = -0.00482925136880233  
  
rat: replaced -0.005822202880477995 by -2532373/434951006 = -0.005822202880477991  
  
rat: replaced -0.006903571435053116 by -1331361/192851050 = -0.006903571435053115  
  
rat: replaced -0.008072248904906765 by -7953293/985263598 = -0.008072248904906766  
  
rat: replaced -0.009327118431599252 by -432515/46371771 = -0.009327118431599259  
  
rat: replaced -0.01066705453755698 by -2950074/276559381 = -0.01066705453755698
```

rat: replaced -0.01209092323861904 by -1254816/103781653 = -0.01209092323861907  
rat: replaced -0.01359758215743526 by -1827823/134422648 = -0.01359758215743526  
rat: replaced -0.01518588063770274 by -9199276/605778237 = -0.01518588063770274  
rat: replaced -0.01685465985923026 by -2516580/149310637 = -0.01685465985923026  
rat: replaced -0.01860275295381958 by -2032371/109251088 = -0.01860275295381955  
rat: replaced -0.02042898512195129 by -1413911/69211025 = -0.02042898512195131  
rat: replaced -0.02233217375026381 by -3647892/163346929 = -0.02233217375026377  
rat: replaced -0.02431112852981362 by -1377268/56651751 = -0.02431112852981367  
rat: replaced -0.02636465157510504 by -2533336/96088355 = -0.02636465157510502  
rat: replaced -0.0284915375438782 by -9699307/340427644 = -0.02849153754387819  
rat: replaced -0.03069057375764189 by -7938451/258660886 = -0.03069057375764189  
rat: replaced -0.0329605403229406 by -2936449/89089832 = -0.03296054032294056  
rat: replaced -0.03530021025334285 by -5224432/148000025 = -0.03530021025334287  
rat: replaced -0.03770834959213837 by -2448749/64939172 = -0.03770834959213832  
rat: replaced -0.04018371753573358 by -2461511/61256428 = -0.04018371753573356  
rat: replaced -0.04272506655773012 by -13954421/326609696 = -0.04272506655773012  
rat: replaced -0.04533114253367693 by -2051558/45257143 = -0.04533114253367695

rat: replaced  $-0.04800068486648146$  by  $-16995415/354066094 = -0.04800068486648145$   
rat: replaced  $-0.05073242661246818$  by  $-3970295/78259513 = -0.05073242661246818$   
rat: replaced  $-0.05352509460807248$  by  $-3894269/72755948 = -0.05352509460807246$   
rat: replaced  $-0.05637740959715515$  by  $-11093364/196769665 = -0.05637740959715513$   
rat: replaced  $-0.05928808635892763$  by  $-3489209/58851773 = -0.05928808635892754$   
rat: replaced  $-0.06225583383647254$  by  $-3380435/54299088 = -0.06225583383647254$   
rat: replaced  $-0.06527935526584844$  by  $-7571267/115982564 = -0.06527935526584841$   
rat: replaced  $-0.06835734830576551$  by  $-8050241/117767017 = -0.06835734830576544$   
rat: replaced  $-0.07148850516781785$  by  $-5513427/77123266 = -0.07148850516781798$   
rat: replaced  $-0.07467151274726203$  by  $-2259975/30265558 = -0.07467151274726208$   
rat: replaced  $-0.07790505275432569$  by  $-657797/8443573 = -0.07790505275432569$   
rat: replaced  $-0.08118780184603619$  by  $-4832180/59518547 = -0.08118780184603633$   
rat: replaced  $-0.08451843175855339$  by  $-3076049/36395008 = -0.08451843175855327$   
rat: replaced  $-0.08789560943999458$  by  $-7150621/81353563 = -0.08789560943999465$   
rat: replaced  $-0.0913179971837394$  by  $-20067867/219758072 = -0.0913179971837394$   
rat: replaced  $-0.09478425276219882$  by  $-5487749/57897265 = -0.09478425276219869$   
rat: replaced  $-0.09829302956103664$  by  $-4406725/44832528 = -0.09829302956103658$   
rat: replaced  $-0.1018429767138303$  by  $-3912367/38415678 = -0.1018429767138302$

rat: replaced -0.1054327392371563 by -8451941/80164293 = -0.1054327392371564  
rat: replaced -0.1090609581660869 by -13126833/120362348 = -0.1090609581660869  
rat: replaced -0.112726270690086 by -2754747/24437489 = -0.112726270690086  
rat: replaced -0.116427310289289 by -22239618/191017193 = -0.116427310289289  
rat: replaced -0.1201627068711536 by -9494831/79016454 = -0.1201627068711537  
rat: replaced -0.1239310869074673 by -3190398/25743323 = -0.1239310869074672  
rat: replaced -0.1277310735717007 by -15999330/125257931 = -0.1277310735717006  
rat: replaced -0.1315612868766867 by -13929723/105880106 = -0.1315612868766867  
rat: replaced -0.1354203438126204 by -28035370/207024803 = -0.1354203438126204  
rat: replaced -0.1393068584853572 by -11590983/83204683 = -0.1393068584853571  
rat: replaced -0.1432194422550018 by -12738764/88945773 = -0.1432194422550018  
rat: replaced -0.1471567038747712 by -5246589/35653075 = -0.147156703874771  
rat: replaced -0.1511172496301179 by -4676629/30947023 = -0.1511172496301179  
rat: replaced -0.1550996834780995 by -15854305/102220099 = -0.1550996834780995  
rat: replaced -0.1591026071869839 by -9026555/56734174 = -0.159102607186984  
rat: replaced -0.1631246204760689 by -10435073/63969945 = -0.1631246204760689  
rat: replaced -0.1671643211557106 by -164873401/986295400 = -0.1671643211557106

rat: replaced  $-0.1712203052675407$  by  $-7017638/40986015 = -0.1712203052675406$   
rat: replaced  $-0.1752911672248615$  by  $-3184915/18169284 = -0.1752911672248615$   
rat: replaced  $-0.1793754999532028$  by  $-2646709/14755131 = -0.1793754999532027$   
rat: replaced  $-0.1834718950310287$  by  $-8392143/45740755 = -0.1834718950310287$   
rat: replaced  $-0.1875789428305783$  by  $-12888313/68708741 = -0.1875789428305781$   
rat: replaced  $-0.1916952326588277$  by  $-16014703/83542521 = -0.1916952326588277$   
rat: replaced  $-0.1958193528985573$  by  $-21279927/108671215 = -0.1958193528985574$   
rat: replaced  $-0.1999498911495134$  by  $-5994245/29978736 = -0.1999498911495134$   
rat: replaced  $-0.2040854343696463$  by  $-17847769/87452439 = -0.2040854343696464$   
rat: replaced  $-0.2082245690164135$  by  $-5203892/24991729 = -0.2082245690164134$   
rat: replaced  $-0.2123658811881329$  by  $-20393053/96027916 = -0.2123658811881328$   
rat: replaced  $-0.2165079567653719$  by  $-8489188/39209589 = -0.2165079567653719$   
rat: replaced  $-0.2206493815523576$  by  $-14881929/67446049 = -0.2206493815523575$   
rat: replaced  $-0.2247887414183958$  by  $-11437558/50881365 = -0.2247887414183955$   
rat: replaced  $-0.2289246224392826$  by  $-17547464/76651711 = -0.2289246224392825$   
rat: replaced  $-0.2330556110386959$  by  $-11148764/47837355 = -0.2330556110386956$   
rat: replaced  $-0.2371802941295513$  by  $-11052217/46598378 = -0.237180294129551$   
rat: replaced  $-0.2412972592553108$  by  $-36037383/149348497 = -0.2412972592553108$

```
rat: replaced -0.2454050947312253 by -4652365/18957899 = -0.2454050947312252
rat: replaced -0.2495023897855041 by -6175634/24751803 = -0.2495023897855037
rat: replaced -0.2535877347003893 by -11299519/44558618 = -0.2535877347003895
rat: replaced -0.2576597209531272 by -6871877/26670358 = -0.2576597209531271
rat: replaced -0.2617169413568191 by -2245730/8580759 = -0.2617169413568194
rat: replaced -0.2657579902011391 by -10500993/39513367 = -0.2657579902011388
rat: replaced -0.2697814633929034 by -21050552/78028163 = -0.2697814633929034
rat: replaced -0.2737859585964791 by -1510231/5516101 = -0.2737859585964796
rat: replaced -0.2777700753740163 by -9819093/35349715 = -0.2777700753740164
rat: replaced -0.2817324153254904 by -10837378/38466919 = -0.2817324153254905
rat: replaced -0.2856715822285418 by -17041418/59653879 = -0.2856715822285421
rat: replaced -0.289586182178096 by -721506/2491507 = -0.2895861821780955
rat: replaced -0.2934748237257534 by -11793110/40184401 = -0.2934748237257537
rat: replaced -0.2973361180189332 by -15390047/51759763 = -0.2973361180189329
rat: replaced -1.00165832502809e-4 by -535089/5342031176 = -1.00165832502809e-4
rat: replaced -4.013199735114076e-4 by -779636/1942679287 = -4.013199735114075e-4
rat: replaced -9.044322995292522e-4 by -524677/580117495 = -9.044322995292531e-4
```

rat: replaced -0.001610452491410008 by  $-2370713/1472078818 = -0.001610452491410008$   
rat: replaced -0.002520309939389107 by  $-92559/36725245 = -0.002520309939389104$   
rat: replaced -0.003634913650147023 by  $-1950438/536584411 = -0.003634913650147022$   
rat: replaced -0.004955152155908069 by  $-1126716/227382725 = -0.004955152155908062$   
rat: replaced -0.006481893425588428 by  $-972955/150103517 = -0.006481893425588422$   
rat: replaced -0.00821598477800041 by  $-318177/38726581 = -0.008215984778000413$   
rat: replaced -0.01015825279712021 by  $-1696171/166974679 = -0.01015825279712021$   
rat: replaced -0.01230950324943156 by  $-3071603/249531028 = -0.01230950324943157$   
rat: replaced -0.01467052100334813 by  $-1108148/75535695 = -0.01467052100334815$   
rat: replaced -0.01724206995072897 by  $-3656561/212072043 = -0.01724206995072896$   
rat: replaced -0.02002489293048906 by  $-7918949/395455248 = -0.02002489293048907$   
rat: replaced -0.02301971165431629 by  $-747397/32467696 = -0.02301971165431634$   
rat: replaced -0.02622722663450017 by  $-4067487/155086432 = -0.02622722663450017$   
rat: replaced -0.02964811711388246 by  $-2777477/93681396 = -0.02964811711388246$   
rat: replaced -0.03328304099793292 by  $-563339/16925707 = -0.03328304099793291$   
rat: replaced -0.03713263478895881 by  $-8128846/218913795 = -0.03713263478895882$   
rat: replaced -0.04119751352245554 by  $-4789499/116256992 = -0.04119751352245549$   
rat: replaced -0.0454782707056039 by  $-4689976/103125645 = -0.04547827070560383$

rat: replaced  $-0.04997547825791965$  by  $-3780197/75641037 = -0.0499754782579197$   
rat: replaced  $-0.05468968645406205$  by  $-7762811/141942869 = -0.05468968645406202$   
rat: replaced  $-0.05962142386880631$  by  $-931586/15625021 = -0.05962142386880632$   
rat: replaced  $-0.0647711973241876$  by  $-3587937/55394020 = -0.06477119732418771$   
rat: replaced  $-0.07013949183881846$  by  $-8850563/126185160 = -0.07013949183881844$   
rat: replaced  $-0.07572677057938781$  by  $-6606579/87242318 = -0.07572677057938786$   
rat: replaced  $-0.08153347481434448$  by  $-9568762/117359919 = -0.0815334748143444$   
rat: replaced  $-0.08756002386977008$  by  $-10007103/114288491 = -0.08756002386977005$   
rat: replaced  $-0.09380681508744848$  by  $-4116683/43884690 = -0.0938068150874485$   
rat: replaced  $-0.1002742237851297$  by  $-7402097/73818542 = -0.1002742237851298$   
rat: replaced  $-0.1069626032190006$  by  $-4704154/43979427 = -0.1069626032190006$   
rat: replaced  $-0.1138722845483578$  by  $-6905284/60640603 = -0.1138722845483578$   
rat: replaced  $-0.1210035768024932$  by  $-4945013/40866668 = -0.1210035768024934$   
rat: replaced  $-0.1283567668497909$  by  $-174125156/1356571689 = -0.1283567668497909$   
rat: replaced  $-0.1359321193690404$  by  $-1669939/12285095 = -0.1359321193690403$   
rat: replaced  $-0.1437298768229693$  by  $-2883920/20064861 = -0.1437298768229693$   
rat: replaced  $-0.1517502594339971$  by  $-13646521/89927497 = -0.1517502594339971$

rat: replaced  $-0.1599934651622126$  by  $-8139181/50871959 = -0.1599934651622124$   
rat: replaced  $-0.1684596696855795$  by  $-7231383/42926494 = -0.1684596696855793$   
rat: replaced  $-0.1771490263823671$  by  $-8210161/46346069 = -0.177149026382367$   
rat: replaced  $-0.1860616663158136$  by  $-37562009/201879354 = -0.1860616663158136$   
rat: replaced  $-0.1951976982210191$  by  $-5455884/27950555 = -0.1951976982210192$   
rat: replaced  $-0.2045572084940737$  by  $-22523003/110106132 = -0.2045572084940737$   
rat: replaced  $-0.2141402611834163$  by  $-32619650/152328431 = -0.2141402611834163$   
rat: replaced  $-0.2239468979834299$  by  $-13386607/59775809 = -0.2239468979834301$   
rat: replaced  $-0.2339771382302741$  by  $-10271620/43900101 = -0.2339771382302742$   
rat: replaced  $-0.244230978899949$  by  $-8014993/32817266 = -0.2442309788999486$   
rat: replaced  $-0.2547083946085993$  by  $-6543653/25690763 = -0.2547083946085992$   
rat: replaced  $-0.2654093376150518$  by  $-8928803/33641631 = -0.265409337615052$   
rat: replaced  $-0.2763337378255902$  by  $-18341761/66375395 = -0.2763337378255903$   
rat: replaced  $-0.2874815028009638$  by  $-9829665/34192339 = -0.2874815028009637$   
rat: replaced  $-0.2988525177656313$  by  $-54659344/182897385 = -0.2988525177656313$   
rat: replaced  $-0.3104466456192388$  by  $-11128869/35847928 = -0.3104466456192391$   
rat: replaced  $-0.3222637269503297$  by  $-3000119/9309515 = -0.3222637269503298$   
rat: replaced  $-0.3343035800522847$  by  $-17486063/52305940 = -0.3343035800522847$

rat: replaced  $-0.3465660009414936$  by  $-56802607/163901268 = -0.3465660009414936$   
rat: replaced  $-0.3590507633777529$  by  $-11187457/31158427 = -0.3590507633777533$   
rat: replaced  $-0.3717576188868896$  by  $-26309122/70769557 = -0.3717576188868895$   
rat: replaced  $-0.3846862967856085$  by  $-11423435/29695456 = -0.3846862967856092$   
rat: replaced  $-0.3978365042085601$  by  $-16989224/42704035 = -0.3978365042085601$   
rat: replaced  $-0.4112079261376275$  by  $-11659135/28353381 = -0.4112079261376271$   
rat: replaced  $-0.4248002254334272$  by  $-67609726/159156521 = -0.4248002254334273$   
rat: replaced  $-0.4386130428690231$  by  $-22452660/51190133 = -0.4386130428690232$   
rat: replaced  $-0.4526459971658492$  by  $-5841603/12905456 = -0.4526459971658499$   
rat: replaced  $-0.4668986850318365$  by  $-9748690/20879669 = -0.4668986850318365$   
rat: replaced  $-0.4813706812017424$  by  $-28304029/58798822 = -0.4813706812017424$   
rat: replaced  $-0.4960615384796762$  by  $-18513661/37321299 = -0.4960615384796762$   
rat: replaced  $-0.5109707877838195$  by  $-12122645/23724732 = -0.5109707877838199$   
rat: replaced  $-0.5260979381933327$  by  $-2102905/3997174 = -0.5260979381933336$   
rat: replaced  $-0.5414424769974477$  by  $-13834851/25551839 = -0.5414424769974482$   
rat: replaced  $-0.5570038697467375$  by  $-46945257/84281743 = -0.5570038697467374$   
rat: replaced  $-0.572781560306562$  by  $-17582077/30695955 = -0.5727815603065616$

rat: replaced  $-0.5887749709126798$  by  $-30776397/52271918 = -0.5887749709126802$   
rat: replaced  $-0.6049835022290249$  by  $-19188269/31717012 = -0.6049835022290246$   
rat: replaced  $-0.6214065334076391$  by  $-11602067/18670655 = -0.6214065334076389$   
rat: replaced  $-0.6380434221507573$  by  $-6649723/10422054 = -0.6380434221507584$   
rat: replaced  $-0.6548935047750358$  by  $-43869993/66987980 = -0.6548935047750357$   
rat: replaced  $-0.6719560962779209$  by  $-31668213/47128396 = -0.6719560962779213$   
rat: replaced  $-0.6892304904061473$  by  $-31252599/45344191 = -0.689230490406147$   
rat: replaced  $-0.7067159597263644$  by  $-7239052/10243227 = -0.7067159597263636$   
rat: replaced  $-0.724411755697878$  by  $-65966965/91062803 = -0.7244117556978781$   
rat: replaced  $-0.742317108747504$  by  $-29643877/39934250 = -0.7423171087475037$   
rat: replaced  $-0.7604312283465253$  by  $-19521554/25671689 = -0.760431228346526$   
rat: replaced  $-0.778753303089744$  by  $-17717453/22751047 = -0.7787533030897436$   
rat: replaced  $-0.7972825007766203$  by  $-6544613/8208650 = -0.7972825007766198$   
rat: replaced  $-0.8160179684944936$  by  $-15744063/19293770 = -0.8160179684944933$   
rat: replaced  $-0.8349588327038714$  by  $-31965589/38284030 = -0.8349588327038716$   
rat: replaced  $-0.8541041993257835$  by  $-22076179/25847173 = -0.8541041993257832$   
rat: replaced  $-0.8734531538311887$  by  $-59286729/67876255 = -0.8734531538311888$   
rat: replaced  $-0.8930047613324276$  by  $-6137127/6872446 = -0.8930047613324281$

```

rat: replaced -0.9127580666767096 by -13137137/14392792 = -0.9127580666767088

rat: replaced -0.9327120945416275 by -15972523/17124816 = -0.932712094541629

rat: replaced -0.9528658495326905 by -44894507/47115244 = -0.9528658495326905

rat: replaced -0.9732183162828605 by -25482581/26183828 = -0.9732183162828598

rat: replaced -0.9937684595540898 by -23211595/23357146 = -0.9937684595540911

rat: replaced -1.014515224340843 by -33401253/32923363 = -1.014515224340843

rat: replaced -1.035457535975596 by -9211102/8895683 = -1.035457535975596

rat: replaced -1.056594300236306 by -24469996/23159311 = -1.056594300236307
part: invalid index of list or matrix.
#0: lineIntersection(g=[1,-1,0],h=[-1,-2,-7/2])
#1: circleThrough(a=[1,0],b=[0,1],c=[2,2])
-- an error. To debug this try: debugmode(true);

Error in:
cc &= circleThrough(A,B,C); $cc // (titik pusat dan jari-jari) ...
^
```

```

>r&=getCircleRadius(cc); $r , $float(r) // tampilan nilai jari-jari
>$computeAngle(A,C,B) // nilai <ACB
>$solve(getLineEquation(angleBisector(A,C,B),x,y),y)[1] // persamaan garis bagi <ACB
```

```

Maxima said:
solve: all variables must not be numbers.
-- an error. To debug this try: debugmode(true);
```

```
Error in:  
... (getLineEquation(angleBisector(A,C,B),x,y),y)[1] // persamaan ...  
^
```

```
>P &= lineIntersection(angleBisector(A,C,B),angleBisector(C,B,A)); $P // titik potong 2 garis bagi s
```

```
Maxima said:  
rat: replaced -4.98329175014009e-5 by -86001/1725786976 = -4.983291750140082e-5  
  
rat: replaced -1.986600267553235e-4 by -1133306/5704751069 = -1.986600267553234e-4  
  
rat: replaced -4.454664535081185e-4 by -474290/1064704191 = -4.454664535081181e-4  
  
rat: replaced -7.892275256562442e-4 by -1190199/1508055613 = -7.892275256562439e-4  
  
rat: replaced -0.001228908875712045 by -259907/211494119 = -0.001228908875712047  
  
rat: replaced -0.001763466544240408 by -5854594/3319934829 = -0.001763466544240408  
  
rat: replaced -0.002391847084253176 by -866601/362314550 = -0.002391847084253172  
  
rat: replaced -0.003112987666553255 by -5049204/1621980085 = -0.003112987666553255  
  
rat: replaced -0.00392581618601677 by -1241039/316122544 = -0.003925816186016774  
  
rat: replaced -0.004829251368802329 by -3015690/624463249 = -0.00482925136880233  
  
rat: replaced -0.005822202880477995 by -2532373/434951006 = -0.005822202880477991  
  
rat: replaced -0.006903571435053116 by -1331361/192851050 = -0.006903571435053115
```

rat: replaced  $-0.008072248904906765$  by  $-7953293/985263598 = -0.008072248904906766$   
rat: replaced  $-0.009327118431599252$  by  $-432515/46371771 = -0.009327118431599259$   
rat: replaced  $-0.01066705453755698$  by  $-2950074/276559381 = -0.01066705453755698$   
rat: replaced  $-0.01209092323861904$  by  $-1254816/103781653 = -0.01209092323861907$   
rat: replaced  $-0.01359758215743526$  by  $-1827823/134422648 = -0.01359758215743526$   
rat: replaced  $-0.01518588063770274$  by  $-9199276/605778237 = -0.01518588063770274$   
rat: replaced  $-0.01685465985923026$  by  $-2516580/149310637 = -0.01685465985923026$   
rat: replaced  $-0.01860275295381958$  by  $-2032371/109251088 = -0.01860275295381955$   
rat: replaced  $-0.02042898512195129$  by  $-1413911/69211025 = -0.02042898512195131$   
rat: replaced  $-0.02233217375026381$  by  $-3647892/163346929 = -0.02233217375026377$   
rat: replaced  $-0.02431112852981362$  by  $-1377268/56651751 = -0.02431112852981367$   
rat: replaced  $-0.02636465157510504$  by  $-2533336/96088355 = -0.02636465157510502$   
rat: replaced  $-0.0284915375438782$  by  $-9699307/340427644 = -0.02849153754387819$   
rat: replaced  $-0.03069057375764189$  by  $-7938451/258660886 = -0.03069057375764189$   
rat: replaced  $-0.0329605403229406$  by  $-2936449/89089832 = -0.03296054032294056$   
rat: replaced  $-0.03530021025334285$  by  $-5224432/148000025 = -0.03530021025334287$   
rat: replaced  $-0.03770834959213837$  by  $-2448749/64939172 = -0.03770834959213832$   
rat: replaced  $-0.04018371753573358$  by  $-2461511/61256428 = -0.04018371753573356$

rat: replaced -0.04272506655773012 by  $-13954421/326609696 = -0.04272506655773012$

rat: replaced -0.04533114253367693 by  $-2051558/45257143 = -0.04533114253367695$

rat: replaced -0.04800068486648146 by  $-16995415/354066094 = -0.04800068486648145$

rat: replaced -0.05073242661246818 by  $-3970295/78259513 = -0.05073242661246818$

rat: replaced -0.05352509460807248 by  $-3894269/72755948 = -0.05352509460807246$

rat: replaced -0.05637740959715515 by  $-11093364/196769665 = -0.05637740959715513$

rat: replaced -0.05928808635892763 by  $-3489209/58851773 = -0.05928808635892754$

rat: replaced -0.06225583383647254 by  $-3380435/54299088 = -0.06225583383647254$

rat: replaced -0.06527935526584844 by  $-7571267/115982564 = -0.06527935526584841$

rat: replaced -0.06835734830576551 by  $-8050241/117767017 = -0.06835734830576544$

rat: replaced -0.07148850516781785 by  $-5513427/77123266 = -0.07148850516781798$

rat: replaced -0.07467151274726203 by  $-2259975/30265558 = -0.07467151274726208$

rat: replaced -0.07790505275432569 by  $-657797/8443573 = -0.07790505275432569$

rat: replaced -0.08118780184603619 by  $-4832180/59518547 = -0.08118780184603633$

rat: replaced -0.08451843175855339 by  $-3076049/36395008 = -0.08451843175855327$

rat: replaced -0.08789560943999458 by  $-7150621/81353563 = -0.08789560943999465$

rat: replaced -0.0913179971837394 by  $-20067867/219758072 = -0.0913179971837394$

rat: replaced  $-0.09478425276219882$  by  $-5487749/57897265 = -0.09478425276219869$

rat: replaced  $-0.09829302956103664$  by  $-4406725/44832528 = -0.09829302956103658$

rat: replaced  $-0.1018429767138303$  by  $-3912367/38415678 = -0.1018429767138302$

rat: replaced  $-0.1054327392371563$  by  $-8451941/80164293 = -0.1054327392371564$

rat: replaced  $-0.1090609581660869$  by  $-13126833/120362348 = -0.1090609581660869$

rat: replaced  $-0.112726270690086$  by  $-2754747/24437489 = -0.112726270690086$

rat: replaced  $-0.116427310289289$  by  $-22239618/191017193 = -0.116427310289289$

rat: replaced  $-0.1201627068711536$  by  $-9494831/79016454 = -0.1201627068711537$

rat: replaced  $-0.1239310869074673$  by  $-3190398/25743323 = -0.1239310869074672$

rat: replaced  $-0.1277310735717007$  by  $-15999330/125257931 = -0.1277310735717006$

rat: replaced  $-0.1315612868766867$  by  $-13929723/105880106 = -0.1315612868766867$

rat: replaced  $-0.1354203438126204$  by  $-28035370/207024803 = -0.1354203438126204$

rat: replaced  $-0.1393068584853572$  by  $-11590983/83204683 = -0.1393068584853571$

rat: replaced  $-0.1432194422550018$  by  $-12738764/88945773 = -0.1432194422550018$

rat: replaced  $-0.1471567038747712$  by  $-5246589/35653075 = -0.147156703874771$

rat: replaced  $-0.1511172496301179$  by  $-4676629/30947023 = -0.1511172496301179$

rat: replaced  $-0.1550996834780995$  by  $-15854305/102220099 = -0.1550996834780995$

rat: replaced  $-0.1591026071869839$  by  $-9026555/56734174 = -0.159102607186984$

rat: replaced  $-0.1631246204760689$  by  $-10435073/63969945 = -0.1631246204760689$

rat: replaced  $-0.1671643211557106$  by  $-164873401/986295400 = -0.1671643211557106$

rat: replaced  $-0.1712203052675407$  by  $-7017638/40986015 = -0.1712203052675406$

rat: replaced  $-0.1752911672248615$  by  $-3184915/18169284 = -0.1752911672248615$

rat: replaced  $-0.1793754999532028$  by  $-2646709/14755131 = -0.1793754999532027$

rat: replaced  $-0.1834718950310287$  by  $-8392143/45740755 = -0.1834718950310287$

rat: replaced  $-0.1875789428305783$  by  $-12888313/68708741 = -0.1875789428305781$

rat: replaced  $-0.1916952326588277$  by  $-16014703/83542521 = -0.1916952326588277$

rat: replaced  $-0.1958193528985573$  by  $-21279927/108671215 = -0.1958193528985574$

rat: replaced  $-0.1999498911495134$  by  $-5994245/29978736 = -0.1999498911495134$

rat: replaced  $-0.2040854343696463$  by  $-17847769/87452439 = -0.2040854343696464$

rat: replaced  $-0.2082245690164135$  by  $-5203892/24991729 = -0.2082245690164134$

rat: replaced  $-0.2123658811881329$  by  $-20393053/96027916 = -0.2123658811881328$

rat: replaced  $-0.2165079567653719$  by  $-8489188/39209589 = -0.2165079567653719$

rat: replaced  $-0.2206493815523576$  by  $-14881929/67446049 = -0.2206493815523575$

rat: replaced  $-0.2247887414183958$  by  $-11437558/50881365 = -0.2247887414183955$

rat: replaced  $-0.2289246224392826$  by  $-17547464/76651711 = -0.2289246224392825$

rat: replaced  $-0.2330556110386959$  by  $-11148764/47837355 = -0.2330556110386956$   
rat: replaced  $-0.2371802941295513$  by  $-11052217/46598378 = -0.237180294129551$   
rat: replaced  $-0.2412972592553108$  by  $-36037383/149348497 = -0.2412972592553108$   
rat: replaced  $-0.2454050947312253$  by  $-4652365/18957899 = -0.2454050947312252$   
rat: replaced  $-0.2495023897855041$  by  $-6175634/24751803 = -0.2495023897855037$   
rat: replaced  $-0.2535877347003893$  by  $-11299519/44558618 = -0.2535877347003895$   
rat: replaced  $-0.2576597209531272$  by  $-6871877/26670358 = -0.2576597209531271$   
rat: replaced  $-0.2617169413568191$  by  $-2245730/8580759 = -0.2617169413568194$   
rat: replaced  $-0.2657579902011391$  by  $-10500993/39513367 = -0.2657579902011388$   
rat: replaced  $-0.2697814633929034$  by  $-21050552/78028163 = -0.2697814633929034$   
rat: replaced  $-0.2737859585964791$  by  $-1510231/5516101 = -0.2737859585964796$   
rat: replaced  $-0.2777700753740163$  by  $-9819093/35349715 = -0.2777700753740164$   
rat: replaced  $-0.2817324153254904$  by  $-10837378/38466919 = -0.2817324153254905$   
rat: replaced  $-0.2856715822285418$  by  $-17041418/59653879 = -0.2856715822285421$   
rat: replaced  $-0.289586182178096$  by  $-721506/2491507 = -0.2895861821780955$   
rat: replaced  $-0.2934748237257534$  by  $-11793110/40184401 = -0.2934748237257537$   
rat: replaced  $-0.2973361180189332$  by  $-15390047/51759763 = -0.2973361180189329$   
rat: replaced  $1.66665833335744e-7$  by  $15819/94914474571 = 1.66665833335744e-7$

rat: replaced  $4.999958333473664e-5$  by  $201389/4027813565 = 4.99995833347366e-5$   
rat: replaced  $1.33330666692022e-6$  by  $31771/23828726570 = 1.333306666920221e-6$   
rat: replaced  $1.999933334222437e-4$  by  $200030/1000183339 = 1.999933334222437e-4$   
rat: replaced  $4.499797504338432e-6$  by  $24036/5341573699 = 4.499797504338431e-6$   
rat: replaced  $4.499662510124569e-4$  by  $1162901/2584418270 = 4.499662510124571e-4$   
rat: replaced  $1.066581336583994e-5$  by  $58861/5518660226 = 1.066581336583993e-5$   
rat: replaced  $7.998933390220841e-4$  by  $1137431/1421978337 = 7.998933390220838e-4$   
rat: replaced  $2.083072932167196e-5$  by  $35635/1710693824 = 2.0830729321672e-5$   
rat: replaced  $0.001249739605033717$  by  $567943/454449069 = 0.001249739605033716$   
rat: replaced  $3.599352055540239e-5$  by  $98277/2730408098 = 3.599352055540234e-5$   
rat: replaced  $0.00179946006479581$  by  $479561/266502719 = 0.001799460064795812$   
rat: replaced  $5.71526624672386e-5$  by  $51154/895041417 = 5.715266246723866e-5$   
rat: replaced  $0.002448999746720415$  by  $1946227/794702818 = 0.002448999746720415$   
rat: replaced  $8.530603082730626e-5$  by  $121691/1426522824 = 8.530603082730627e-5$   
rat: replaced  $0.003198293697380561$  by  $2986741/933854512 = 0.003198293697380562$   
rat: replaced  $1.214508019889565e-4$  by  $158455/1304684674 = 1.214508019889563e-4$   
rat: replaced  $0.004047266988005727$  by  $2125334/525128193 = 0.004047266988005727$

```
rat: replaced 1.665833531718508e-4 by 142521/855553675 = 1.66583353171851e-4
rat: replaced 0.004995834721974179 by 1957223/391770967 = 0.004995834721974179
rat: replaced 2.216991628251896e-4 by 179571/809975995 = 2.216991628251896e-4
rat: replaced 0.006043902043303184 by 1800665/297930871 = 0.006043902043303193
rat: replaced 2.877927110806339e-4 by 1167733/4057548906 = 2.877927110806339e-4
rat: replaced 0.00719136414613375 by 2476362/344352191 = 0.007191364146133747
rat: replaced 3.658573803051457e-4 by 386279/1055818526 = 3.658573803051454e-4
rat: replaced 0.00843810628521191 by 2079855/246483622 = 0.008438106285211924
rat: replaced 4.5688535576352e-4 by 262978/575588595 = 4.568853557635206e-4
rat: replaced 0.009784003787362772 by 1752551/179124113 = 0.009784003787362787
rat: replaced 5.618675264007778e-4 by 150595/268025812 = 5.618675264007782e-4
rat: replaced 0.01122892206395776 by 5450241/485375263 = 0.01122892206395776
rat: replaced 6.817933857540259e-4 by 192316/282073725 = 6.817933857540258e-4
rat: replaced 0.01277271662437307 by 3258991/255152533 = 0.01277271662437308
rat: replaced 8.176509330039827e-4 by 105841/129445214 = 8.176509330039812e-4
rat: replaced 0.01441523309043924 by 2330472/161667313 = 0.01441523309043925
rat: replaced 9.704265741758145e-4 by 651321/671169790 = 9.704265741758132e-4
rat: replaced 0.01615630721187855 by 19391318/1200232067 = 0.01615630721187855
```

rat: replaced 0.001141105023499428 by 1259907/1104111343 = 0.001141105023499428

rat: replaced 0.01799576488272969 by 4765614/264818641 = 0.01799576488272969

rat: replaced 0.001330669204938795 by 1231154/925214167 = 0.001330669204938796

rat: replaced 0.01993342215875837 by 2504519/125644206 = 0.01993342215875836

rat: replaced 0.001540100153900437 by 276884/179783113 = 0.001540100153900439

rat: replaced 0.02196908527585173 by 1298306/59096953 = 0.0219690852758517

rat: replaced 0.001770376919130678 by 644389/363984072 = 0.001770376919130681

rat: replaced 0.02410255066939448 by 2001286/83032125 = 0.02410255066939453

rat: replaced 0.002022476464811601 by 1271955/628909667 = 0.002022476464811599

rat: replaced 0.02633360499462523 by 2978115/113091808 = 0.02633360499462525

rat: replaced 0.002297373572865413 by 1020913/444382669 = 0.002297373572865417

rat: replaced 0.02866202514797045 by 1770713/61779061 = 0.02866202514797044

rat: replaced 0.002596040745477063 by 1097643/422814242 = 0.002596040745477065

rat: replaced 0.03108757828935527 by 5034207/161936287 = 0.03108757828935525

rat: replaced 0.002919448107844891 by 906221/310408326 = 0.002919448107844891

rat: replaced 0.03361002186548678 by 4553215/135471944 = 0.03361002186548678

rat: replaced 0.003268563311168871 by 1379071/421919623 = 0.003268563311168867

rat: replaced 0.03622910363410947 by 3082649/85087642 = 0.0362291036341094  
rat: replaced 0.003644351435886262 by 5966577/1637212301 = 0.003644351435886261  
rat: replaced 0.03894456168922911 by 4913415/126164342 = 0.03894456168922911  
rat: replaced 0.004047774895164447 by 572425/141417202 = 0.004047774895164451  
rat: replaced 0.04175612448730281 by 1734727/41544253 = 0.04175612448730273  
rat: replaced 0.004479793338660443 by 2952779/659132861 = 0.004479793338660444  
rat: replaced 0.04466351087439402 by 4691119/105032473 = 0.04466351087439405  
rat: replaced 0.0049413635565565 by 2524919/510976165 = 0.004941363556556498  
rat: replaced 0.04766643011428662 by 3536207/74186529 = 0.04766643011428665  
rat: replaced 0.005433439383882244 by 1361584/250593391 = 0.005433439383882235  
rat: replaced 0.05076458191755917 by 7710025/151878036 = 0.05076458191755916  
rat: replaced 0.005956971605131645 by 1447422/242979503 = 0.005956971605131648  
rat: replaced 0.0539576564716131 by 3377975/62604183 = 0.05395765647161309  
rat: replaced 0.006512907859185624 by 3695063/567344584 = 0.006512907859185626  
rat: replaced 0.05724533447165381 by 2560865/44734912 = 0.05724533447165382  
rat: replaced 0.007102192544548636 by 1363981/192050693 = 0.007102192544548642  
rat: replaced 0.06062728715262111 by 8274761/136485754 = 0.06062728715262107  
rat: replaced 0.007725766724910044 by 1464384/189545459 = 0.007725766724910038

rat: replaced 0.06410317632206519 by 5287663/82486755 = 0.06410317632206528  
rat: replaced 0.00838456803503801 by 1113589/132814117 = 0.008384568035038023  
rat: replaced 0.06767265439396564 by 2921400/43169579 = 0.06767265439396572  
rat: replaced 0.009079530587017326 by 433906/47789475 = 0.00907953058701733  
rat: replaced 0.07133536442348987 by 7236103/101437808 = 0.07133536442348991  
rat: replaced 0.009811584876838586 by 1363090/138926587 = 0.009811584876838586  
rat: replaced 0.07509094014268702 by 9209133/122639735 = 0.07509094014268704  
rat: replaced 0.0105816576913495 by 1163729/109976058 = 0.01058165769134951  
rat: replaced 0.07893900599711501 by 5197067/65836489 = 0.07893900599711506  
rat: replaced 0.01139067201557714 by 13426050/1178688139 = 0.01139067201557714  
rat: replaced 0.08287917718339499 by 11217158/135343501 = 0.082879177183395  
rat: replaced 0.01223954694042984 by 2283101/186534764 = 0.01223954694042983  
rat: replaced 0.08691105968769186 by 5213115/59982182 = 0.08691105968769192  
rat: replaced 0.01312919757078923 by 3499615/266552086 = 0.01312919757078922  
rat: replaced 0.09103425032511492 by 5893225/64736349 = 0.09103425032511488  
rat: replaced 0.01406053493400045 by 2280713/162206702 = 0.01406053493400045  
rat: replaced 0.09524833678003664 by 9601787/100807923 = 0.09524833678003662

rat: replaced 0.01503446588876983 by 200490/13335359 = 0.01503446588876985  
rat: replaced 0.09955289764732322 by 5687088/57126293 = 0.09955289764732328  
rat: replaced 0.01605189303448024 by 951971/59305840 = 0.01605189303448025  
rat: replaced 0.1039475024744748 by 10260011/98703776 = 0.1039475024744747  
rat: replaced 0.01711371462093175 by 9432386/551159477 = 0.01711371462093176  
rat: replaced 0.1084317118046711 by 14939691/137779721 = 0.1084317118046712  
rat: replaced 0.01822082445851714 by 2559788/140486947 = 0.01822082445851713  
rat: replaced 0.113005077220716 by 8478529/75027859 = 0.1130050772207161  
rat: replaced 0.01937411182884202 by 2983799/154009589 = 0.01937411182884203  
rat: replaced 0.1176671413898787 by 7123715/60541243 = 0.1176671413898786  
rat: replaced 0.02057446139579705 by 7167743/348380590 = 0.02057446139579705  
rat: replaced 0.1224174381096274 by 12172179/99431741 = 0.1224174381096274  
rat: replaced 0.02182275311709253 by 7415562/339808729 = 0.02182275311709253  
rat: replaced 0.1272554923542488 by 7277933/57191504 = 0.127255492354249  
rat: replaced 0.02311986215626333 by 2988661/129268115 = 0.02311986215626336  
rat: replaced 0.1321808203223502 by 3633064/27485561 = 0.1321808203223503  
rat: replaced 0.02446665879515308 by 1991976/81415939 = 0.02446665879515312  
rat: replaced 0.1371929294852391 by 56235017/409897341 = 0.1371929294852391

rat: replaced 0.02586400834688696 by 5000736/193347293 = 0.02586400834688697  
rat: replaced 0.1422913186361759 by 9349741/65708443 = 0.1422913186361759  
rat: replaced 0.02731277106934082 by 858413/31428997 = 0.02731277106934084  
rat: replaced 0.1474754779404944 by 1549881/10509415 = 0.1474754779404943  
rat: replaced 0.02881380207911666 by 3754753/130310918 = 0.02881380207911666  
rat: replaced 0.152744888986584 by 5264425/34465474 = 0.1527448889865841  
rat: replaced 0.03036795126603076 by 4118329/135614318 = 0.03036795126603077  
rat: replaced 0.1580990248377314 by 5442776/34426373 = 0.1580990248377312  
rat: replaced 0.03197606320812652 by 3497683/109384416 = 0.03197606320812647  
rat: replaced 0.1635373500848132 by 12328488/75386375 = 0.1635373500848131  
rat: replaced 0.0336389770872163 by 3971799/118071337 = 0.03363897708721635  
rat: replaced 0.1690593208998367 by 20896917/123607009 = 0.1690593208998367  
rat: replaced 0.03535752660496472 by 1815732/51353479 = 0.03535752660496478  
rat: replaced 0.1746643850903219 by 2841592/16268869 = 0.1746643850903219  
rat: replaced 0.03713253989951881 by 3333721/89778965 = 0.03713253989951878  
rat: replaced 0.1803519821545206 by 4461007/24735004 = 0.1803519821545208  
rat: replaced 0.03896483946269502 by 8785771/225479461 = 0.03896483946269501

rat: replaced 0.1861215433374662 by  $4381209/23539505 = 0.1861215433374661$   
rat: replaced 0.0408552420577305 by  $3189084/78058135 = 0.04085524205773043$   
rat: replaced 0.1919724916878484 by  $72809759/379271834 = 0.1919724916878484$   
rat: replaced 0.04280455863760801 by  $7646593/178639688 = 0.04280455863760801$   
rat: replaced 0.1979042421157076 by  $26318167/132984350 = 0.1979042421157076$   
rat: replaced 0.04481359426396048 by  $20610430/459914683 = 0.04481359426396048$   
rat: replaced 0.2039162014509444 by  $8519416/41779005 = 0.2039162014509441$   
rat: replaced 0.04688314802656623 by  $3439140/73355569 = 0.04688314802656633$   
rat: replaced 0.2100077685026351 by  $50962787/242670961 = 0.2100077685026351$   
rat: replaced 0.04901401296344043 by  $4006732/81746663 = 0.04901401296344048$   
rat: replaced 0.216178334119151 by  $1347531/6233423 = 0.2161783341191509$   
rat: replaced 0.05120697598153157 by  $4148974/81023609 = 0.0512069759815315$   
rat: replaced 0.2224272812490723 by  $23234851/104460437 = 0.2224272812490723$   
rat: replaced 0.05346281777803219 by  $11998448/224426031 = 0.05346281777803218$   
rat: replaced 0.2287539850028937 by  $8185268/35781969 = 0.2287539850028935$   
rat: replaced 0.05578231276230905 by  $1398019/25062048 = 0.05578231276230897$   
rat: replaced 0.2351578127155118 by  $12642104/53760085 = 0.2351578127155119$   
rat: replaced 0.05816622897846346 by  $4451048/76522891 = 0.05816622897846345$

rat: replaced 0.2416381240094921 by 8002142/33116223 = 0.2416381240094923  
rat: replaced 0.06061532802852698 by 2146337/35409146 = 0.06061532802852686  
rat: replaced 0.2481942708591053 by 8882901/35790113 = 0.2481942708591057  
rat: replaced 0.0631303649963022 by 14651447/232082406 = 0.06313036499630222  
rat: replaced 0.2548255976551299 by 868346/3407609 = 0.25482559765513  
rat: replaced 0.06571208837185505 by 4240309/64528599 = 0.06571208837185509  
rat: replaced 0.2615314412704124 by 8212450/31401387 = 0.2615314412704127  
rat: replaced 0.06836123997666599 by 2716643/39739522 = 0.06836123997666604  
rat: replaced 0.2683111311261794 by 34459769/128432126 = 0.2683111311261794  
rat: replaced 0.07107855488944881 by 3146673/44270357 = 0.07107855488944893  
rat: replaced 0.2751639892590951 by 12552159/45617012 = 0.2751639892590949  
rat: replaced 0.07386476137264342 by 12898997/174629915 = 0.0738647613726434  
rat: replaced 0.2820893303890569 by 11134456/39471383 = 0.2820893303890568  
rat: replaced 0.07672058079958999 by 5073506/66129661 = 0.07672058079959007  
rat: replaced 0.2890864619877229 by 9583357/33150487 = 0.2890864619877228  
rat: replaced 0.07964672758239233 by 5672399/71219486 = 0.07964672758239227  
rat: replaced 0.2961546843477643 by 11052271/37319251 = 0.2961546843477647

rat: replaced 0.08264390910047736 by 4686067/56701904 = 0.08264390910047748  
rat: replaced 0.3032932906528349 by 9918077/32701274 = 0.3032932906528351  
rat: replaced 0.0857128256298576 by 3585977/41837111 = 0.08571282562985766  
rat: replaced 0.3105015670482534 by 9320011/30015987 = 0.3105015670482533  
rat: replaced 0.08885417027310427 by 5751353/64728003 = 0.0888541702731042  
rat: replaced 0.3177787927123868 by 248395525/781661743 = 0.3177787927123868  
rat: replaced 0.09206862889003742 by 7305460/79347983 = 0.09206862889003745  
rat: replaced 0.3251242399287333 by 13842845/42577093 = 0.3251242399287335  
rat: replaced 0.09535688002914089 by 5971998/62627867 = 0.09535688002914103  
rat: replaced 0.3325371741586922 by 9318229/28021616 = 0.3325371741586923  
rat: replaced 0.0987195948597075 by 9821211/99485933 = 0.09871959485970745  
rat: replaced 0.3400168541150183 by 13391981/39386227 = 0.3400168541150184  
rat: replaced 0.1021574371047232 by 8336413/81603584 = 0.1021574371047232  
rat: replaced 0.3475625318359485 by 10097818/29053241 = 0.347562531835949  
rat: replaced 0.1056710629744951 by 5741011/54329074 = 0.105671062974495  
rat: replaced 0.3551734527599992 by 15867851/44676343 = 0.3551734527599987  
rat: replaced 0.1092611211010309 by 5551873/50812887 = 0.1092611211010309  
rat: replaced 0.3628488558014202 by 6897641/19009681 = 0.3628488558014203

```
rat: replaced 0.1129282524731764 by 11548693/102265755 = 0.1129282524731764
rat: replaced 0.3705879734263036 by 23358661/63031352 = 0.3705879734263038
rat: replaced 0.1166730903725168 by 5656228/48479285 = 0.1166730903725168
rat: replaced 0.3783900317293359 by 14241382/37636779 = 0.3783900317293358
rat: replaced 0.1204962603100498 by 4057613/33674182 = 0.12049626031005
rat: replaced 0.3862542505111889 by 3461217/8960981 = 0.3862542505111884
rat: replaced 0.1243983799636342 by 7966447/64039797 = 0.1243983799636342
rat: replaced 0.3941798433565377 by 5314214/13481699 = 0.3941798433565384
rat: replaced 0.1283800591162231 by 796346/6203035 = 0.1283800591162229
rat: replaced 0.4021660177127022 by 11567173/28762184 = 0.4021660177127022
rat: replaced 0.1324418995948859 by 4716124/35609003 = 0.1324418995948862
rat: replaced 0.4102119749689023 by 11320633/27597032 = 0.4102119749689024
rat: replaced 0.1365844952106265 by 612971/4487852 = 0.1365844952106264
rat: replaced 0.418316910536117 by 12225195/29224721 = 0.4183169105361177
rat: replaced 0.140808431699002 by 10431632/74083859 = 0.1408084316990021
rat: replaced 0.4264800139275439 by 7978696/18708253 = 0.4264800139275431
rat: replaced 0.1451142866615502 by 3554077/24491572 = 0.1451142866615504
```

```

rat: replaced 0.4347004688396462 by 20489554/47134879 = 0.4347004688396463

rat: replaced 0.1495026295080298 by 26759297/178988805 = 0.1495026295080298

rat: replaced 0.4429774532337832 by 23449796/52936771 = 0.4429774532337834

rat: replaced 0.1539740213994798 by 16145763/104860306 = 0.1539740213994798

rat: replaced 0.451310139418413 by 8841241/19590167 = 0.4513101394184133
part: invalid index of list or matrix.
#0: lineIntersection(g=[1,-1,0],h=[2-sqrt(5)/sqrt(2),1+sqrt(5)/sqrt(2),(3-sqrt(5)/sqrt(2))*(1+sqrt(5)/sqrt(2))]);
-- an error. To debug this try: debugmode(true);

Error in:
... ection(angleBisector(A,C,B),angleBisector(C,B,A)); $P // titik ...
^
```

>P() // hasilnya sama dengan perhitungan sebelumnya

```

Function P needs at least 2 arguments!
Use: P (x, n)
Error in:
P() // hasilnya sama dengan perhitungan sebelumnya ...
^
```

## Garis dan Lingkaran yang Berpotongan

Tentu saja, kita juga dapat memotong garis dengan lingkaran, dan lingkaran dengan lingkaran.

```
>A &:= [1,0]; c=circleWithCenter(A,4);
>B &:= [1,2]; C &:= [2,1]; l=lineThrough(B,C);
>setPlotRange(5); plotCircle(c); plotLine(l);
```

Perpotongan garis dengan lingkaran menghasilkan dua titik dan jumlah titik potong.

```
>{P1,P2,f}=lineCircleIntersections(l,c);
>P1, P2,
```

```
[4.64575, -1.64575]
[-0.645751, 3.64575]
```

```
>plotPoint(P1); plotPoint(P2);
```

Begitu pula di Maxima.

```
>c &= circleWithCenter(A,4) // lingkaran dengan pusat A jari-jari 4
```

```
[1, 0, 4]
```

```
>l &= lineThrough(B,C) // garis l melalui B dan C
```

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

```
>$lineCircleIntersections(l,c) | radcan, // titik potong lingkaran c dan garis l
```

Maxima said:

```
rat: replaced -4.98329175014009e-5 by -86001/1725786976 = -4.983291750140082e-5
```

```
rat: replaced -1.986600267553235e-4 by -1133306/5704751069 = -1.986600267553234e-4
```

```
rat: replaced -4.454664535081185e-4 by -474290/1064704191 = -4.454664535081181e-4
```

```
rat: replaced -7.892275256562442e-4 by -1190199/1508055613 = -7.892275256562439e-4
```

```
rat: replaced -0.001228908875712045 by -259907/211494119 = -0.001228908875712047
```

rat: replaced -0.001763466544240408 by  $-5854594/3319934829 = -0.001763466544240408$   
rat: replaced -0.002391847084253176 by  $-866601/362314550 = -0.002391847084253172$   
rat: replaced -0.003112987666553255 by  $-5049204/1621980085 = -0.003112987666553255$   
rat: replaced -0.00392581618601677 by  $-1241039/316122544 = -0.003925816186016774$   
rat: replaced -0.004829251368802329 by  $-3015690/624463249 = -0.00482925136880233$   
rat: replaced -0.005822202880477995 by  $-2532373/434951006 = -0.005822202880477991$   
rat: replaced -0.006903571435053116 by  $-1331361/192851050 = -0.006903571435053115$   
rat: replaced -0.008072248904906765 by  $-7953293/985263598 = -0.008072248904906766$   
rat: replaced -0.009327118431599252 by  $-432515/46371771 = -0.009327118431599259$   
rat: replaced -0.01066705453755698 by  $-2950074/276559381 = -0.01066705453755698$   
rat: replaced -0.01209092323861904 by  $-1254816/103781653 = -0.01209092323861907$   
rat: replaced -0.01359758215743526 by  $-1827823/134422648 = -0.01359758215743526$   
rat: replaced -0.01518588063770274 by  $-9199276/605778237 = -0.01518588063770274$   
rat: replaced -0.01685465985923026 by  $-2516580/149310637 = -0.01685465985923026$   
rat: replaced -0.01860275295381958 by  $-2032371/109251088 = -0.01860275295381955$   
rat: replaced -0.02042898512195129 by  $-1413911/69211025 = -0.02042898512195131$   
rat: replaced -0.02233217375026381 by  $-3647892/163346929 = -0.02233217375026377$

rat: replaced  $-0.02431112852981362$  by  $-1377268/56651751 = -0.02431112852981367$   
rat: replaced  $-0.02636465157510504$  by  $-2533336/96088355 = -0.02636465157510502$   
rat: replaced  $-0.0284915375438782$  by  $-9699307/340427644 = -0.02849153754387819$   
rat: replaced  $-0.03069057375764189$  by  $-7938451/258660886 = -0.03069057375764189$   
rat: replaced  $-0.0329605403229406$  by  $-2936449/89089832 = -0.03296054032294056$   
rat: replaced  $-0.03530021025334285$  by  $-5224432/148000025 = -0.03530021025334287$   
rat: replaced  $-0.03770834959213837$  by  $-2448749/64939172 = -0.03770834959213832$   
rat: replaced  $-0.04018371753573358$  by  $-2461511/61256428 = -0.04018371753573356$   
rat: replaced  $-0.04272506655773012$  by  $-13954421/326609696 = -0.04272506655773012$   
rat: replaced  $-0.04533114253367693$  by  $-2051558/45257143 = -0.04533114253367695$   
rat: replaced  $-0.04800068486648146$  by  $-16995415/354066094 = -0.04800068486648145$   
rat: replaced  $-0.05073242661246818$  by  $-3970295/78259513 = -0.05073242661246818$   
rat: replaced  $-0.05352509460807248$  by  $-3894269/72755948 = -0.05352509460807246$   
rat: replaced  $-0.05637740959715515$  by  $-11093364/196769665 = -0.05637740959715513$   
rat: replaced  $-0.05928808635892763$  by  $-3489209/58851773 = -0.05928808635892754$   
rat: replaced  $-0.06225583383647254$  by  $-3380435/54299088 = -0.06225583383647254$   
rat: replaced  $-0.06527935526584844$  by  $-7571267/115982564 = -0.06527935526584841$   
rat: replaced  $-0.06835734830576551$  by  $-8050241/117767017 = -0.06835734830576544$

rat: replaced  $-0.07148850516781785$  by  $-5513427/77123266 = -0.07148850516781798$   
rat: replaced  $-0.07467151274726203$  by  $-2259975/30265558 = -0.07467151274726208$   
rat: replaced  $-0.07790505275432569$  by  $-657797/8443573 = -0.07790505275432569$   
rat: replaced  $-0.08118780184603619$  by  $-4832180/59518547 = -0.08118780184603633$   
rat: replaced  $-0.08451843175855339$  by  $-3076049/36395008 = -0.08451843175855327$   
rat: replaced  $-0.08789560943999458$  by  $-7150621/81353563 = -0.08789560943999465$   
rat: replaced  $-0.0913179971837394$  by  $-20067867/219758072 = -0.0913179971837394$   
rat: replaced  $-0.09478425276219882$  by  $-5487749/57897265 = -0.09478425276219869$   
rat: replaced  $-0.09829302956103664$  by  $-4406725/44832528 = -0.09829302956103658$   
rat: replaced  $-0.1018429767138303$  by  $-3912367/38415678 = -0.1018429767138302$   
rat: replaced  $-0.1054327392371563$  by  $-8451941/80164293 = -0.1054327392371564$   
rat: replaced  $-0.1090609581660869$  by  $-13126833/120362348 = -0.1090609581660869$   
rat: replaced  $-0.112726270690086$  by  $-2754747/24437489 = -0.112726270690086$   
rat: replaced  $-0.116427310289289$  by  $-22239618/191017193 = -0.116427310289289$   
rat: replaced  $-0.1201627068711536$  by  $-9494831/79016454 = -0.1201627068711537$   
rat: replaced  $-0.1239310869074673$  by  $-3190398/25743323 = -0.1239310869074672$   
rat: replaced  $-0.1277310735717007$  by  $-15999330/125257931 = -0.1277310735717006$

rat: replaced  $-0.1315612868766867$  by  $-13929723/105880106 = -0.1315612868766867$

rat: replaced  $-0.1354203438126204$  by  $-28035370/207024803 = -0.1354203438126204$

rat: replaced  $-0.1393068584853572$  by  $-11590983/83204683 = -0.1393068584853571$

rat: replaced  $-0.1432194422550018$  by  $-12738764/88945773 = -0.1432194422550018$

rat: replaced  $-0.1471567038747712$  by  $-5246589/35653075 = -0.147156703874771$

rat: replaced  $-0.1511172496301179$  by  $-4676629/30947023 = -0.1511172496301179$

rat: replaced  $-0.1550996834780995$  by  $-15854305/102220099 = -0.1550996834780995$

rat: replaced  $-0.1591026071869839$  by  $-9026555/56734174 = -0.159102607186984$

rat: replaced  $-0.1631246204760689$  by  $-10435073/63969945 = -0.1631246204760689$

rat: replaced  $-0.1671643211557106$  by  $-164873401/986295400 = -0.1671643211557106$

rat: replaced  $-0.1712203052675407$  by  $-7017638/40986015 = -0.1712203052675406$

rat: replaced  $-0.1752911672248615$  by  $-3184915/18169284 = -0.1752911672248615$

rat: replaced  $-0.1793754999532028$  by  $-2646709/14755131 = -0.1793754999532027$

rat: replaced  $-0.1834718950310287$  by  $-8392143/45740755 = -0.1834718950310287$

rat: replaced  $-0.1875789428305783$  by  $-12888313/68708741 = -0.1875789428305781$

rat: replaced  $-0.1916952326588277$  by  $-16014703/83542521 = -0.1916952326588277$

rat: replaced  $-0.1958193528985573$  by  $-21279927/108671215 = -0.1958193528985574$

rat: replaced  $-0.1999498911495134$  by  $-5994245/29978736 = -0.1999498911495134$

```
rat: replaced -0.2040854343696463 by -17847769/87452439 = -0.2040854343696464
rat: replaced -0.2082245690164135 by -5203892/24991729 = -0.2082245690164134
rat: replaced -0.2123658811881329 by -20393053/96027916 = -0.2123658811881328
rat: replaced -0.2165079567653719 by -8489188/39209589 = -0.2165079567653719
rat: replaced -0.2206493815523576 by -14881929/67446049 = -0.2206493815523575
rat: replaced -0.2247887414183958 by -11437558/50881365 = -0.2247887414183955
rat: replaced -0.2289246224392826 by -17547464/76651711 = -0.2289246224392825
rat: replaced -0.2330556110386959 by -11148764/47837355 = -0.2330556110386956
rat: replaced -0.2371802941295513 by -11052217/46598378 = -0.237180294129551
rat: replaced -0.2412972592553108 by -36037383/149348497 = -0.2412972592553108
rat: replaced -0.2454050947312253 by -4652365/18957899 = -0.2454050947312252
rat: replaced -0.2495023897855041 by -6175634/24751803 = -0.2495023897855037
rat: replaced -0.2535877347003893 by -11299519/44558618 = -0.2535877347003895
rat: replaced -0.2576597209531272 by -6871877/26670358 = -0.2576597209531271
rat: replaced -0.2617169413568191 by -2245730/8580759 = -0.2617169413568194
rat: replaced -0.2657579902011391 by -10500993/39513367 = -0.2657579902011388
rat: replaced -0.2697814633929034 by -21050552/78028163 = -0.2697814633929034
```

rat: replaced  $-0.2737859585964791$  by  $-1510231/5516101 = -0.2737859585964796$   
rat: replaced  $-0.2777700753740163$  by  $-9819093/35349715 = -0.2777700753740164$   
rat: replaced  $-0.2817324153254904$  by  $-10837378/38466919 = -0.2817324153254905$   
rat: replaced  $-0.2856715822285418$  by  $-17041418/59653879 = -0.2856715822285421$   
rat: replaced  $-0.289586182178096$  by  $-721506/2491507 = -0.2895861821780955$   
rat: replaced  $-0.2934748237257534$  by  $-11793110/40184401 = -0.2934748237257537$   
rat: replaced  $-0.2973361180189332$  by  $-15390047/51759763 = -0.2973361180189329$   
rat: replaced  $5.016624916807239e-5$  by  $153117/3052191514 = 5.016624916807235e-5$   
rat: replaced  $2.013266400891639e-4$  by  $232411/1154397649 = 2.013266400891639e-4$   
rat: replaced  $4.544660485167953e-4$  by  $444871/978887205 = 4.544660485167952e-4$   
rat: replaced  $8.105591523879241e-4$  by  $1425236/1758336817 = 8.105591523879239e-4$   
rat: replaced  $0.001270570334355389$  by  $696221/547959433 = 0.00127057033435539$   
rat: replaced  $0.001835453585351213$  by  $1018402/554850315 = 0.001835453585351213$   
rat: replaced  $0.002506152409187654$  by  $484773/193433168 = 0.002506152409187653$   
rat: replaced  $0.003283599728207867$  by  $1007483/306822720 = 0.003283599728207872$   
rat: replaced  $0.004168717789994683$  by  $897113/215201183 = 0.004168717789994677$   
rat: replaced  $0.00516241807514603$  by  $757433/146720585 = 0.005162418075146034$   
rat: replaced  $0.006265601206128374$  by  $1194190/190594639 = 0.006265601206128363$

rat: replaced 0.007479156857214384 by 1971251/263565939 = 0.007479156857214391  
rat: replaced 0.008803963665517056 by 365844/41554465 = 0.008803963665517051  
rat: replaced 0.01024088914312629 by 1345773/131411734 = 0.01024088914312629  
rat: replaced 0.01179078959035854 by 1519715/128890011 = 0.01179078959035856  
rat: replaced 0.0134545100101271 by 2242921/166704027 = 0.01345451001012711  
rat: replaced 0.01523288402344322 by 1950407/128039247 = 0.01523288402344322  
rat: replaced 0.01712673378605437 by 1362867/79575418 = 0.01712673378605438  
rat: replaced 0.01913686990622912 by 1694449/88543686 = 0.01913686990622911  
rat: replaced 0.02126409136369717 by 9814128/461535263 = 0.02126409136369716  
rat: replaced 0.02350918542975217 by 2315819/98506986 = 0.02350918542975216  
rat: replaced 0.02587292758852516 by 3386321/130882792 = 0.02587292758852516  
rat: replaced 0.02835608145943683 by 10230271/360778728 = 0.02835608145943682  
rat: replaced 0.03095939872083586 by 14307719/462144602 = 0.03095939872083587  
rat: replaced 0.03368361903483233 by 4712088/139892569 = 0.03368361903483236  
rat: replaced 0.03652946997333167 by 4111522/112553563 = 0.03652946997333172  
rat: replaced 0.03949766694527834 by 8626745/218411508 = 0.03949766694527836  
rat: replaced 0.04258891312511537 by 3115258/73147159 = 0.04258891312511536

rat: replaced 0.04580389938246726 by 2358579/51492974 = 0.04580389938246721  
rat: replaced 0.04914330421305446 by 2180747/44375262 = 0.04914330421305456  
rat: replaced 0.05260779367084312 by 4975224/94571995 = 0.05260779367084304  
rat: replaced 0.05619802130144141 by 1396735/24853811 = 0.05619802130144146  
rat: replaced 0.05991462807674475 by 6603037/110207427 = 0.05991462807674477  
rat: replaced 0.06375824233083943 by 6198842/97224167 = 0.0637582423308394  
rat: replaced 0.06772947969716975 by 4012504/59243095 = 0.06772947969716978  
rat: replaced 0.07182894304697524 by 5813372/80933559 = 0.07182894304697511  
rat: replaced 0.07605722242900365 by 14672328/192911699 = 0.07605722242900365  
rat: replaced 0.08041489501050719 by 3507279/43614793 = 0.0804148950105071  
rat: replaced 0.08490252501952561 by 2460362/28978667 = 0.08490252501952557  
rat: replaced 0.08952066368846451 by 4304415/48082921 = 0.08952066368846436  
rat: replaced 0.09426984919897213 by 3898288/41352437 = 0.09426984919897224  
rat: replaced 0.0991506066281217 by 11428253/115261554 = 0.09915060662812164  
rat: replaced 0.1041634478959041 by 7209817/69216382 = 0.1041634478959042  
rat: replaced 0.1093088717140371 by 3826731/35008421 = 0.109308871714037  
rat: replaced 0.1145873635360931 by 5173172/45146095 = 0.1145873635360932  
rat: replaced 0.1199993955089551 by 23218093/193485083 = 0.1199993955089551

rat: replaced 0.1255454264256029 by 2445819/19481546 = 0.125545426425603  
rat: replaced 0.1312259016792331 by 9111136/69430927 = 0.131225901679233  
rat: replaced 0.1370412532187207 by 16597683/121114501 = 0.1370412532187207  
rat: replaced 0.1429918995054244 by 34253454/239548213 = 0.1429918995054244  
rat: replaced 0.1490782454713414 by 11997679/80479073 = 0.1490782454713414  
rat: replaced 0.1553006824786136 by 13065213/84128497 = 0.1553006824786136  
rat: replaced 0.1616595882803922 by 12686167/78474572 = 0.1616595882803923  
rat: replaced 0.1681553269830629 by 4527449/26924208 = 0.168155326983063  
rat: replaced 0.1747882490098353 by 23565700/134824281 = 0.1747882490098353  
rat: replaced 0.1815586910657007 by 4563713/25136296 = 0.1815586910657004  
rat: replaced 0.1884669761037622 by 8213146/43578701 = 0.1884669761037623  
rat: replaced 0.1955134132929397 by 7172626/36686107 = 0.1955134132929395  
rat: replaced 0.202698297987053 by 17668607/87167022 = 0.2026982979870529  
rat: replaced 0.2100219116952866 by 8269584/39374863 = 0.2100219116952864  
rat: replaced 0.2174845220540395 by 56596301/260231397 = 0.2174845220540395  
rat: replaced 0.2250863828001612 by 8187128/36373271 = 0.2250863828001611  
rat: replaced 0.2328277337455789 by 10320856/44328293 = 0.2328277337455787

rat: replaced 0.2407088007533156 by  $16964872/70478819 = 0.2407088007533157$   
rat: replaced 0.2487297957149048 by  $11063220/44478869 = 0.2487297957149045$   
rat: replaced 0.2568909165292014 by  $17200949/66958183 = 0.2568909165292015$   
rat: replaced 0.2651923470825914 by  $8866093/33432688 = 0.2651923470825918$   
rat: replaced 0.2736342572306039 by  $12664159/46281336 = 0.2736342572306037$   
rat: replaced 0.2822168027809259 by  $8116045/28758192 = 0.2822168027809259$   
rat: replaced 0.2909401254778209 by  $24764749/85119744 = 0.290940125477821$   
rat: replaced 0.2998043529879556 by  $28498628/95057419 = 0.2998043529879556$   
rat: replaced 0.3088095988876323 by  $13390352/43361191 = 0.308809598887632$   
rat: replaced 0.3179559626514321 by  $26241235/82531036 = 0.3179559626514321$   
rat: replaced 0.3272435296422674 by  $8247573/25203166 = 0.3272435296422679$   
rat: replaced 0.3366723711028454 by  $10805861/32096073 = 0.3366723711028449$   
rat: replaced 0.3462425441485439 by  $20967050/60555961 = 0.3462425441485438$   
rat: replaced 0.3559540917617003 by  $19053013/53526602 = 0.3559540917617001$   
rat: replaced 0.3658070427873129 by  $10401097/28433288 = 0.3658070427873132$   
rat: replaced 0.3758014119301566 by  $5923743/15762961 = 0.375801411930157$   
rat: replaced 0.3859371997533123 by  $2934328/7603123 = 0.3859371997533119$   
rat: replaced 0.396214392678111 by  $30414315/76762267 = 0.396214392678111$

```
rat: replaced 0.4066329629854911 by 13711485/33719561 = 0.4066329629854908
rat: replaced 0.4171928688187707 by 20838614/49949593 = 0.4171928688187709
rat: replaced 0.4278940541878331 by 16106690/37641771 = 0.427894054187833
rat: replaced 0.4387364489747257 by 4869080/11097961 = 0.4387364489747261
rat: replaced 0.4497199689406718 by 4550581/10118699 = 0.4497199689406711
rat: replaced 0.4608445157344944 by 7970699/17295853 = 0.4608445157344943
rat: replaced 0.4721099769024512 by 25424083/53852035 = 0.4721099769024513
rat: replaced 0.48351622589948 by 17675673/36556525 = 0.4835162258994803
rat: replaced 0.4950631221018528 by 7053395/14247466 = 0.495063122101853
rat: replaced 0.5067505108212387 by 13754758/27143057 = 0.5067505108212388
rat: replaced 0.5185782233201719 by 21662467/41772805 = 0.518578223320172
rat: replaced 0.5305460768289253 by 10488897/19770002 = 0.530546076828925
rat: replaced 0.5426538745637882 by 22388393/41257225 = 0.5426538745637886
rat: replaced 0.5549014057467435 by 9960301/17949677 = 0.5549014057467441
rat: replaced 0.5672884456265459 by 28078535/49496046 = 0.5672884456265456
rat: replaced 0.5798147555011964 by 18086313/31193261 = 0.5798147555011962
rat: replaced 0.5924800827418131 by 20592707/34756792 = 0.5924800827418134
```

```

rat: replaced 0.6052841608178928 by 26813845/44299598 = 0.6052841608178927
part: invalid index of list or matrix.
#0: lineIntersection(g=[1,-1,1],h=[1,1,3])
#1: projectToLine(a=[1,0],g=[1,1,3])
#2: lineCircleIntersections(g=[1,1,3],c=[1,0,4])
-- an error. To debug this try: debugmode(true);

Error in:
$lineCircleIntersections(l,c) | radcan, // titik potong lingka ...
^

```

Akan ditunjukkan bahwa sudut-sudut yang menghadap bsuusr yang sama adalah sama besar.

```

>C=A+normalize([-2,-3])*4; plotPoint(C); plotSegment(P1,C); plotSegment(P2,C);
>degprint(computeAngle(P1,C,P2))

```

$69^\circ 17' 42.68''$

```

>C=A+normalize([-4,-3])*4; plotPoint(C); plotSegment(P1,C); plotSegment(P2,C);
>degprint(computeAngle(P1,C,P2))

```

$69^\circ 17' 42.68''$

```

>insimg;

```

Berikut adalah langkah-langkah menggambar garis sumbu ruas garis AB:

1. Gambar lingkaran dengan pusat A melalui B.
2. Gambar lingkaran dengan pusat B melalui A.
3. Tarik garis melalui kedua titik potong kedua lingkaran tersebut. Garis ini merupakan garis sumbu (melalui titik tengah dan tegak lurus) AB.

```
>A=[2,2]; B=[-1,-2];
>c1=circleWithCenter(A,distance(A,B));
>c2=circleWithCenter(B,distance(A,B));
>{P1,P2,f}=circleCircleIntersections(c1,c2);
>l=lineThrough(P1,P2);
>setPlotRange(5); plotCircle(c1); plotCircle(c2);
>plotPoint(A); plotPoint(B); plotSegment(A,B); plotLine(l);
```

Selanjutnya, kami melakukan hal yang sama di Maxima dengan koordinat umum.

```
>A &= [a1,a2]; B &= [b1,b2];
>c1 &= circleWithCenter(A,distance(A,B));
>c2 &= circleWithCenter(B,distance(A,B));
>P &= circleCircleIntersections(c1,c2); P1 &= P[1]; P2 &= P[2];
```

Persamaan untuk persimpangan cukup terlibat. Tetapi kita dapat menyederhanakannya, jika kita memecahkan  $y$ .

```
>g &= getLineEquation(lineThrough(P1,P2),x,y);  
>$solve(g,y)
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$solve(g,y) ...  
^
```

Ini memang sama dengan tegak lurus tengah, yang dihitung dengan cara yang sama sekali berbeda.

```
>$solve(getLineEquation(middlePerpendicular(A,B),x,y),y)
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
... (getLineEquation(middlePerpendicular(A,B),x,y),y) ...  
^
```

```
>h &=getLineEquation(lineThrough(A,B),x,y);  
>$solve(h,y)
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
 $solve(h,y) ...  
 ^
```

Perhatikan hasil kali gradien garis g dan h adalah:

$$\frac{-(b_1 - a_1)}{(b_2 - a_2)} \times \frac{(b_2 - a_2)}{(b_1 - a_1)} = -1.$$

Artinya kedua garis tegak lurus.

### Contoh 3: Rumus Heron

Rumus Heron menyatakan bahwa luas segitiga dengan panjang sisi-sisi a, b dan c adalah:

$$L = \sqrt{s(s-a)(s-b)(s-c)} \quad \text{dengan } s = (a+b+c)/2,$$

Untuk membuktikan hal ini kita misalkan C(0,0), B(a,0) dan A(x,y), b=AC, c=AB. Luas segitiga ABC adalah

$$L_{\triangle ABC} = \frac{1}{2}a \times y.$$

Nilai y didapat dengan menyelesaikan sistem persamaan:

$$x^2 + y^2 = b^2, \quad (x - a)^2 + y^2 = c^2.$$

```
>setPlotRange(-1,10,-1,8); plotPoint([0,0], "C(0,0)"); plotPoint([5.5,0], "B(a,0)"); ...
>plotPoint([7.5,6], "A(x,y)");
>plotSegment([0,0],[5.5,0], "a",25); plotSegment([5.5,0],[7.5,6],"c",15); ...
>plotSegment([0,0],[7.5,6],"b",25);
>plotSegment([7.5,6],[7.5,0],"t=y",25);
>&assume(a>0); sol &= solve([x^2+y^2=b^2,(x-a)^2+y^2=c^2],[x,y])
```

```
Maxima said:
fullmap: arguments must have same formal structure.
-- an error. To debug this try: debugmode(true);

Error in:
... sol &= solve([x^2+y^2=b^2,(x-a)^2+y^2=c^2],[x,y]) ...
```

Ekstrak solusi y.

```
>ysol &= y with sol[2][2]; $'y=sqrt(factor(ysol^2))
```

```
Maxima said:  
at: improper argument: (3*x*sqrt(4+9*x^2))[2][2]  
#0: with(expr=[0,4.999958333473664e-5*r,1.999933334222437e-4*r,4.499662510124569e-4*r,7.9989333902  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
ysol &= y with sol[2][2]; $'y=sqrt(factor(ysol^2)) ...  
^
```

Kami mendapatkan rumus Heron.

```
>function H(a,b,c) &= sqrt(factor((ysol*a/2)^2)); $'H(a,b,c)=H(a,b,c)  
>$'Luas=H(2,5,6) // luas segitiga dengan panjang sisi-sisi 2, 5, 6
```

Tentu saja, setiap segitiga persegi panjang adalah kasus yang terkenal.

```
>H(3,4,5) //luas segitiga siku-siku dengan panjang sisi 3, 4, 5
```

```
Variable or function ysol not found.  
Try "trace errors" to inspect local variables after errors.  
H:  
    useglobal; return a*abs(ysol)/2  
Error in:  
H(3,4,5) //luas segitiga siku-siku dengan panjang sisi 3, 4, 5 ...  
^
```

Dan juga jelas, bahwa ini adalah segitiga dengan luas maksimal dan dua sisi 3 dan 4.

```
>aspect (1.5); plot2d(&H(3,4,x),1,7); // Kurva luas segitiga sengan panjang sisi 3, 4, x (1<= x <=7)
```

```
Variable or function ysol not found.  
Error in expression: 3*abs(ysol)/2  
%ploteval:  
    y0=f$(x[1],args());  
adaptiveevalone:  
    s=%ploteval(g$,t,args());  
Try "trace errors" to inspect local variables after errors.  
plot2d:  
    dw/n,dw/n^2,dw/n,auto;args());
```

Kasus umum juga berfungsi.

```
>$solve(diff(H(a,b,c)^2,c)=0,c)
```

```
Maxima said:  
diff: second argument must be a variable; found [1,0,4]  
-- an error. To debug this try: debugmode(true);
```

```
Error in:  
$solve(diff(H(a,b,c)^2,c)=0,c) ...  
^
```

Sekarang mari kita cari himpunan semua titik di mana  $b+c=d$  untuk beberapa konstanta d. Diketahui bahwa ini adalah ellips.

```
>s1 &= subst(d-c,b,sol[2]); $s1
```

Dan buat fungsi ini.

```
>function fx(a,c,d) &= rhs(s1[1]); $fx(a,c,d), function fy(a,c,d) &= rhs(s1[2]); $fy(a,c,d)
```

Sekarang kita bisa menggambar setnya. Sisi b bervariasi dari 1 hingga 4. Diketahui bahwa kita mendapatkan elips.

```
>aspect(1); plot2d(&fx(3,x,5),&fy(3,x,5),xmin=1,xmax=4,square=1):
```

Kita dapat memeriksa persamaan umum untuk elips ini, yaitu.

$$\frac{(x - x_m)^2}{u^2} + \frac{(y - y_m)^2}{v^2} = 1,$$

di mana  $(xm,ym)$  adalah pusat, dan  $u$  dan  $v$  adalah setengah sumbu.

```
>$ratsimp((fx(a,c,d)-a/2)^2/u^2+fy(a,c,d)^2/v^2 with [u=d/2,v=sqrt(d^2-a^2)/2])
```

Kita lihat bahwa tinggi dan luas segitiga adalah maksimal untuk  $x=0$ . Jadi luas segitiga dengan  $a+b+c=d$  maksimal jika segitiga sama sisi. Kami ingin menurunkan ini secara analitis.

```
>eqns &= [diff(H(a,b,d-(a+b))^2,a)=0,diff(H(a,b,d-(a+b))^2,b)=0]; $eqns
```

Kami mendapatkan beberapa minima, yang termasuk dalam segitiga dengan satu sisi 0, dan solusinya  $a=b=c=d/3$ .

```
>$solve(eqns,[a,b])
```

Ada juga metode Lagrange, memaksimalkan  $H(a,b,c)^2$  terhadap  $a+b+d=d$ .

```
>&solve([diff(H(a,b,c)^2,a)=la,diff(H(a,b,c)^2,b)=la, ...
>    diff(H(a,b,c)^2,c)=la,a+b+c=d],[a,b,c,la])
```

$$\begin{aligned} & [[a = 0, b = \frac{d}{2}, c = \frac{d}{2}, la = 0], \\ & [a = \frac{d}{2}, b = 0, c = \frac{d}{2}, la = 0], [a = \frac{d}{2}, b = \frac{d}{2}, c = 0, la = 0], \\ & [a = \frac{d}{3}, b = \frac{d}{3}, c = \frac{d}{3}, la = \frac{3d}{108}]] \end{aligned}$$

Kita bisa membuat plot situasinya

Pertama-tama atur poin di Maxima.

```
>A &= at([x,y],sol[2]); $A  
  
Maxima said:  
at: improper argument: (3*x*sqrt(4+9*x^2))[2]  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
A &= at([x,y],sol[2]); $A ...  
^
```

```
>B &= [0,0]; $B, C &= [a,0]; $C
```

Kemudian atur rentang plot, dan plot titik-titiknya.

```
>setPlotRange(0,5,-2,3); ...  
>a=4; b=3; c=2; ...  
>plotPoint(mxmeval("B"),"B"); plotPoint(mxmeval("C"),"C"); ...  
>plotPoint(mxmeval("A"),"A");
```

```
Variable a1 not found!  
Use global variables or parameters for string evaluation.  
Error in Evaluate, superfluous characters found.  
Try "trace errors" to inspect local variables after errors.
```

```
mxmeval:  
    return evaluate(mxm(s));  
Error in:  
... otPoint(mxmeval("C"),"C"); plotPoint(mxmeval("A"),"A"): ...  
^
```

Plot segmen.

```
>plotSegment(mxmeval("A"),mxmeval("C")); ...  
>plotSegment(mxmeval("B"),mxmeval("C")); ...  
>plotSegment(mxmeval("B"),mxmeval("A")):
```

```
Variable a1 not found!  
Use global variables or parameters for string evaluation.  
Error in Evaluate, superfluous characters found.  
Try "trace errors" to inspect local variables after errors.  
mxmeval:  
    return evaluate(mxm(s));  
Error in:  
plotSegment(mxmeval("A"),mxmeval("C")); plotSegment(mxmeval("B") ...  
^
```

Hitung tegak lurus tengah di Maxima.

```
>h &= middlePerpendicular(A,B); g &= middlePerpendicular(B,C);
```

Dan pusat lingkaran.

```
>U &= lineIntersection(h,g);
```

Maxima said:

```
rat: replaced 1.66665833335744e-7 by 15819/94914474571 = 1.66665833335744e-7  
rat: replaced 4.999958333473664e-5 by 201389/4027813565 = 4.99995833347366e-5  
rat: replaced 1.33330666692022e-6 by 31771/23828726570 = 1.333306666920221e-6  
rat: replaced 1.999933334222437e-4 by 200030/1000183339 = 1.999933334222437e-4  
rat: replaced 4.499797504338432e-6 by 24036/5341573699 = 4.499797504338431e-6  
rat: replaced 4.499662510124569e-4 by 1162901/2584418270 = 4.499662510124571e-4  
rat: replaced 1.066581336583994e-5 by 58861/5518660226 = 1.066581336583993e-5  
rat: replaced 7.998933390220841e-4 by 1137431/1421978337 = 7.998933390220838e-4  
rat: replaced 2.083072932167196e-5 by 35635/1710693824 = 2.0830729321672e-5  
rat: replaced 0.001249739605033717 by 567943/454449069 = 0.001249739605033716  
rat: replaced 3.599352055540239e-5 by 98277/2730408098 = 3.599352055540234e-5  
rat: replaced 0.00179946006479581 by 479561/266502719 = 0.001799460064795812  
rat: replaced 5.71526624672386e-5 by 51154/895041417 = 5.715266246723866e-5  
rat: replaced 0.002448999746720415 by 1946227/794702818 = 0.002448999746720415
```

rat: replaced  $8.530603082730626e-5$  by  $121691/1426522824 = 8.530603082730627e-5$   
rat: replaced  $0.003198293697380561$  by  $2986741/933854512 = 0.003198293697380562$   
rat: replaced  $1.214508019889565e-4$  by  $158455/1304684674 = 1.214508019889563e-4$   
rat: replaced  $0.004047266988005727$  by  $2125334/525128193 = 0.004047266988005727$   
rat: replaced  $1.665833531718508e-4$  by  $142521/855553675 = 1.66583353171851e-4$   
rat: replaced  $0.004995834721974179$  by  $1957223/391770967 = 0.004995834721974179$   
rat: replaced  $2.216991628251896e-4$  by  $179571/809975995 = 2.216991628251896e-4$   
rat: replaced  $0.006043902043303184$  by  $1800665/297930871 = 0.006043902043303193$   
rat: replaced  $2.877927110806339e-4$  by  $1167733/4057548906 = 2.877927110806339e-4$   
rat: replaced  $0.00719136414613375$  by  $2476362/344352191 = 0.007191364146133747$   
rat: replaced  $3.658573803051457e-4$  by  $386279/1055818526 = 3.658573803051454e-4$   
rat: replaced  $0.00843810628521191$  by  $2079855/246483622 = 0.008438106285211924$   
rat: replaced  $4.5688535576352e-4$  by  $262978/575588595 = 4.568853557635206e-4$   
rat: replaced  $0.009784003787362772$  by  $1752551/179124113 = 0.009784003787362787$   
rat: replaced  $5.618675264007778e-4$  by  $150595/268025812 = 5.618675264007782e-4$   
rat: replaced  $0.01122892206395776$  by  $5450241/485375263 = 0.01122892206395776$   
rat: replaced  $6.817933857540259e-4$  by  $192316/282073725 = 6.817933857540258e-4$   
rat: replaced  $0.01277271662437307$  by  $3258991/255152533 = 0.01277271662437308$

```
rat: replaced 8.176509330039827e-4 by 105841/129445214 = 8.176509330039812e-4
rat: replaced 0.01441523309043924 by 2330472/161667313 = 0.01441523309043925
rat: replaced 9.704265741758145e-4 by 651321/671169790 = 9.704265741758132e-4
rat: replaced 0.01615630721187855 by 19391318/1200232067 = 0.01615630721187855
rat: replaced 0.001141105023499428 by 1259907/1104111343 = 0.001141105023499428
rat: replaced 0.01799576488272969 by 4765614/264818641 = 0.01799576488272969
rat: replaced 0.001330669204938795 by 1231154/925214167 = 0.001330669204938796
rat: replaced 0.01993342215875837 by 2504519/125644206 = 0.01993342215875836
rat: replaced 0.001540100153900437 by 276884/179783113 = 0.001540100153900439
rat: replaced 0.02196908527585173 by 1298306/59096953 = 0.0219690852758517
rat: replaced 0.001770376919130678 by 644389/363984072 = 0.001770376919130681
rat: replaced 0.02410255066939448 by 2001286/83032125 = 0.02410255066939453
rat: replaced 0.002022476464811601 by 1271955/628909667 = 0.002022476464811599
rat: replaced 0.02633360499462523 by 2978115/113091808 = 0.02633360499462525
rat: replaced 0.002297373572865413 by 1020913/444382669 = 0.002297373572865417
rat: replaced 0.02866202514797045 by 1770713/61779061 = 0.02866202514797044
rat: replaced 0.002596040745477063 by 1097643/422814242 = 0.002596040745477065
```

rat: replaced 0.03108757828935527 by 5034207/161936287 = 0.03108757828935525  
rat: replaced 0.002919448107844891 by 906221/310408326 = 0.002919448107844891  
rat: replaced 0.03361002186548678 by 4553215/135471944 = 0.03361002186548678  
rat: replaced 0.003268563311168871 by 1379071/421919623 = 0.003268563311168867  
rat: replaced 0.03622910363410947 by 3082649/85087642 = 0.0362291036341094  
rat: replaced 0.003644351435886262 by 5966577/1637212301 = 0.003644351435886261  
rat: replaced 0.03894456168922911 by 4913415/126164342 = 0.03894456168922911  
rat: replaced 0.004047774895164447 by 572425/141417202 = 0.004047774895164451  
rat: replaced 0.04175612448730281 by 1734727/41544253 = 0.04175612448730273  
rat: replaced 0.004479793338660443 by 2952779/659132861 = 0.004479793338660444  
rat: replaced 0.04466351087439402 by 4691119/105032473 = 0.04466351087439405  
rat: replaced 0.0049413635565565 by 2524919/510976165 = 0.004941363556556498  
rat: replaced 0.04766643011428662 by 3536207/74186529 = 0.04766643011428665  
rat: replaced 0.005433439383882244 by 1361584/250593391 = 0.005433439383882235  
rat: replaced 0.05076458191755917 by 7710025/151878036 = 0.05076458191755916  
rat: replaced 0.005956971605131645 by 1447422/242979503 = 0.005956971605131648  
rat: replaced 0.0539576564716131 by 3377975/62604183 = 0.05395765647161309  
rat: replaced 0.006512907859185624 by 3695063/567344584 = 0.006512907859185626

rat: replaced 0.05724533447165381 by 2560865/44734912 = 0.05724533447165382  
rat: replaced 0.007102192544548636 by 1363981/192050693 = 0.007102192544548642  
rat: replaced 0.06062728715262111 by 8274761/136485754 = 0.06062728715262107  
rat: replaced 0.007725766724910044 by 1464384/189545459 = 0.007725766724910038  
rat: replaced 0.06410317632206519 by 5287663/82486755 = 0.06410317632206528  
rat: replaced 0.00838456803503801 by 1113589/132814117 = 0.008384568035038023  
rat: replaced 0.06767265439396564 by 2921400/43169579 = 0.06767265439396572  
rat: replaced 0.009079530587017326 by 433906/47789475 = 0.00907953058701733  
rat: replaced 0.07133536442348987 by 7236103/101437808 = 0.07133536442348991  
rat: replaced 0.009811584876838586 by 1363090/138926587 = 0.009811584876838586  
rat: replaced 0.07509094014268702 by 9209133/122639735 = 0.07509094014268704  
rat: replaced 0.0105816576913495 by 1163729/109976058 = 0.01058165769134951  
rat: replaced 0.07893900599711501 by 5197067/65836489 = 0.07893900599711506  
rat: replaced 0.01139067201557714 by 13426050/1178688139 = 0.01139067201557714  
rat: replaced 0.08287917718339499 by 11217158/135343501 = 0.082879177183395  
rat: replaced 0.01223954694042984 by 2283101/186534764 = 0.01223954694042983  
rat: replaced 0.08691105968769186 by 5213115/59982182 = 0.08691105968769192

rat: replaced 0.01312919757078923 by 3499615/266552086 = 0.01312919757078922  
rat: replaced 0.09103425032511492 by 5893225/64736349 = 0.09103425032511488  
rat: replaced 0.01406053493400045 by 2280713/162206702 = 0.01406053493400045  
rat: replaced 0.09524833678003664 by 9601787/100807923 = 0.09524833678003662  
rat: replaced 0.01503446588876983 by 200490/13335359 = 0.01503446588876985  
rat: replaced 0.09955289764732322 by 5687088/57126293 = 0.09955289764732328  
rat: replaced 0.01605189303448024 by 951971/59305840 = 0.01605189303448025  
rat: replaced 0.1039475024744748 by 10260011/98703776 = 0.1039475024744747  
rat: replaced 0.01711371462093175 by 9432386/551159477 = 0.01711371462093176  
rat: replaced 0.1084317118046711 by 14939691/137779721 = 0.1084317118046712  
rat: replaced 0.01822082445851714 by 2559788/140486947 = 0.01822082445851713  
rat: replaced 0.113005077220716 by 8478529/75027859 = 0.1130050772207161  
rat: replaced 0.01937411182884202 by 2983799/154009589 = 0.01937411182884203  
rat: replaced 0.1176671413898787 by 7123715/60541243 = 0.1176671413898786  
rat: replaced 0.02057446139579705 by 7167743/348380590 = 0.02057446139579705  
rat: replaced 0.1224174381096274 by 12172179/99431741 = 0.1224174381096274  
rat: replaced 0.02182275311709253 by 7415562/339808729 = 0.02182275311709253  
rat: replaced 0.1272554923542488 by 7277933/57191504 = 0.127255492354249

rat: replaced 0.02311986215626333 by 2988661/129268115 = 0.02311986215626336  
rat: replaced 0.1321808203223502 by 3633064/27485561 = 0.1321808203223503  
rat: replaced 0.02446665879515308 by 1991976/81415939 = 0.02446665879515312  
rat: replaced 0.1371929294852391 by 56235017/409897341 = 0.1371929294852391  
rat: replaced 0.02586400834688696 by 5000736/193347293 = 0.02586400834688697  
rat: replaced 0.1422913186361759 by 9349741/65708443 = 0.1422913186361759  
rat: replaced 0.02731277106934082 by 858413/31428997 = 0.02731277106934084  
rat: replaced 0.1474754779404944 by 1549881/10509415 = 0.1474754779404943  
rat: replaced 0.02881380207911666 by 3754753/130310918 = 0.02881380207911666  
rat: replaced 0.152744888986584 by 5264425/34465474 = 0.1527448889865841  
rat: replaced 0.03036795126603076 by 4118329/135614318 = 0.03036795126603077  
rat: replaced 0.1580990248377314 by 5442776/34426373 = 0.1580990248377312  
rat: replaced 0.03197606320812652 by 3497683/109384416 = 0.03197606320812647  
rat: replaced 0.1635373500848132 by 12328488/75386375 = 0.1635373500848131  
rat: replaced 0.0336389770872163 by 3971799/118071337 = 0.03363897708721635  
rat: replaced 0.1690593208998367 by 20896917/123607009 = 0.1690593208998367  
rat: replaced 0.03535752660496472 by 1815732/51353479 = 0.03535752660496478

rat: replaced 0.1746643850903219 by 2841592/16268869 = 0.1746643850903219  
rat: replaced 0.03713253989951881 by 3333721/89778965 = 0.03713253989951878  
rat: replaced 0.1803519821545206 by 4461007/24735004 = 0.1803519821545208  
rat: replaced 0.03896483946269502 by 8785771/225479461 = 0.03896483946269501  
rat: replaced 0.1861215433374662 by 4381209/23539505 = 0.1861215433374661  
rat: replaced 0.0408552420577305 by 3189084/78058135 = 0.04085524205773043  
rat: replaced 0.1919724916878484 by 72809759/379271834 = 0.1919724916878484  
rat: replaced 0.04280455863760801 by 7646593/178639688 = 0.04280455863760801  
rat: replaced 0.1979042421157076 by 26318167/132984350 = 0.1979042421157076  
rat: replaced 0.04481359426396048 by 20610430/459914683 = 0.04481359426396048  
rat: replaced 0.2039162014509444 by 8519416/41779005 = 0.2039162014509441  
rat: replaced 0.04688314802656623 by 3439140/73355569 = 0.04688314802656633  
rat: replaced 0.2100077685026351 by 50962787/242670961 = 0.2100077685026351  
rat: replaced 0.04901401296344043 by 4006732/81746663 = 0.04901401296344048  
rat: replaced 0.216178334119151 by 1347531/6233423 = 0.2161783341191509  
rat: replaced 0.05120697598153157 by 4148974/81023609 = 0.0512069759815315  
rat: replaced 0.2224272812490723 by 23234851/104460437 = 0.2224272812490723  
rat: replaced 0.05346281777803219 by 11998448/224426031 = 0.05346281777803218

```
rat: replaced 0.2287539850028937 by 8185268/35781969 = 0.2287539850028935
rat: replaced 0.05578231276230905 by 1398019/25062048 = 0.05578231276230897
rat: replaced 0.2351578127155118 by 12642104/53760085 = 0.2351578127155119
rat: replaced 0.05816622897846346 by 4451048/76522891 = 0.05816622897846345
rat: replaced 0.2416381240094921 by 8002142/33116223 = 0.2416381240094923
rat: replaced 0.06061532802852698 by 2146337/35409146 = 0.06061532802852686
rat: replaced 0.2481942708591053 by 8882901/35790113 = 0.2481942708591057
rat: replaced 0.0631303649963022 by 14651447/232082406 = 0.06313036499630222
rat: replaced 0.2548255976551299 by 868346/3407609 = 0.25482559765513
rat: replaced 0.06571208837185505 by 4240309/64528599 = 0.06571208837185509
rat: replaced 0.2615314412704124 by 8212450/31401387 = 0.2615314412704127
rat: replaced 0.06836123997666599 by 2716643/39739522 = 0.06836123997666604
rat: replaced 0.2683111311261794 by 34459769/128432126 = 0.2683111311261794
rat: replaced 0.07107855488944881 by 3146673/44270357 = 0.07107855488944893
rat: replaced 0.2751639892590951 by 12552159/45617012 = 0.2751639892590949
rat: replaced 0.07386476137264342 by 12898997/174629915 = 0.0738647613726434
rat: replaced 0.2820893303890569 by 11134456/39471383 = 0.2820893303890568
```

rat: replaced 0.07672058079958999 by 5073506/66129661 = 0.07672058079959007  
rat: replaced 0.2890864619877229 by 9583357/33150487 = 0.2890864619877228  
rat: replaced 0.07964672758239233 by 5672399/71219486 = 0.07964672758239227  
rat: replaced 0.2961546843477643 by 11052271/37319251 = 0.2961546843477647  
rat: replaced 0.08264390910047736 by 4686067/56701904 = 0.08264390910047748  
rat: replaced 0.3032932906528349 by 9918077/32701274 = 0.3032932906528351  
rat: replaced 0.0857128256298576 by 3585977/41837111 = 0.08571282562985766  
rat: replaced 0.3105015670482534 by 9320011/30015987 = 0.3105015670482533  
rat: replaced 0.08885417027310427 by 5751353/64728003 = 0.0888541702731042  
rat: replaced 0.3177787927123868 by 248395525/781661743 = 0.3177787927123868  
rat: replaced 0.09206862889003742 by 7305460/79347983 = 0.09206862889003745  
rat: replaced 0.3251242399287333 by 13842845/42577093 = 0.3251242399287335  
rat: replaced 0.09535688002914089 by 5971998/62627867 = 0.09535688002914103  
rat: replaced 0.3325371741586922 by 9318229/28021616 = 0.3325371741586923  
rat: replaced 0.0987195948597075 by 9821211/99485933 = 0.09871959485970745  
rat: replaced 0.3400168541150183 by 13391981/39386227 = 0.3400168541150184  
rat: replaced 0.1021574371047232 by 8336413/81603584 = 0.1021574371047232  
rat: replaced 0.3475625318359485 by 10097818/29053241 = 0.347562531835949

```
rat: replaced 0.1056710629744951 by 5741011/54329074 = 0.105671062974495
rat: replaced 0.3551734527599992 by 15867851/44676343 = 0.3551734527599987
rat: replaced 0.1092611211010309 by 5551873/50812887 = 0.1092611211010309
rat: replaced 0.3628488558014202 by 6897641/19009681 = 0.3628488558014203
rat: replaced 0.1129282524731764 by 11548693/102265755 = 0.1129282524731764
rat: replaced 0.3705879734263036 by 23358661/63031352 = 0.3705879734263038
rat: replaced 0.1166730903725168 by 5656228/48479285 = 0.1166730903725168
rat: replaced 0.3783900317293359 by 14241382/37636779 = 0.3783900317293358
rat: replaced 0.1204962603100498 by 4057613/33674182 = 0.12049626031005
rat: replaced 0.3862542505111889 by 3461217/8960981 = 0.3862542505111884
rat: replaced 0.1243983799636342 by 7966447/64039797 = 0.1243983799636342
rat: replaced 0.3941798433565377 by 5314214/13481699 = 0.3941798433565384
rat: replaced 0.1283800591162231 by 796346/6203035 = 0.1283800591162229
rat: replaced 0.4021660177127022 by 11567173/28762184 = 0.4021660177127022
rat: replaced 0.1324418995948859 by 4716124/35609003 = 0.1324418995948862
rat: replaced 0.4102119749689023 by 11320633/27597032 = 0.4102119749689024
rat: replaced 0.1365844952106265 by 612971/4487852 = 0.1365844952106264
```

```
rat: replaced 0.418316910536117 by 12225195/29224721 = 0.4183169105361177
rat: replaced 0.140808431699002 by 10431632/74083859 = 0.1408084316990021
rat: replaced 0.4264800139275439 by 7978696/18708253 = 0.4264800139275431
rat: replaced 0.1451142866615502 by 3554077/24491572 = 0.1451142866615504
rat: replaced 0.4347004688396462 by 20489554/47134879 = 0.4347004688396463
rat: replaced 0.1495026295080298 by 26759297/178988805 = 0.1495026295080298
rat: replaced 0.4429774532337832 by 23449796/52936771 = 0.4429774532337834
rat: replaced 0.1539740213994798 by 16145763/104860306 = 0.1539740213994798
rat: replaced 0.451310139418413 by 8841241/19590167 = 0.4513101394184133
rat: replaced -1.66665833335744e-7 by -15819/94914474571 = -1.66665833335744e-7
rat: replaced -1.33330666692022e-6 by -31771/23828726570 = -1.333306666920221e-6
rat: replaced -4.499797504338432e-6 by -24036/5341573699 = -4.499797504338431e-6
rat: replaced -1.066581336583994e-5 by -58861/5518660226 = -1.066581336583993e-5
rat: replaced -2.083072932167196e-5 by -35635/1710693824 = -2.0830729321672e-5
rat: replaced -3.599352055540239e-5 by -98277/2730408098 = -3.599352055540234e-5
rat: replaced -5.71526624672386e-5 by -51154/895041417 = -5.715266246723866e-5
rat: replaced -8.530603082730626e-5 by -121691/1426522824 = -8.530603082730627e-5
rat: replaced -1.214508019889565e-4 by -158455/1304684674 = -1.214508019889563e-4
```

```
rat: replaced -1.665833531718508e-4 by -142521/855553675 = -1.66583353171851e-4
rat: replaced -2.216991628251896e-4 by -179571/809975995 = -2.216991628251896e-4
rat: replaced -2.877927110806339e-4 by -1167733/4057548906 = -2.877927110806339e-4
rat: replaced -3.658573803051457e-4 by -386279/1055818526 = -3.658573803051454e-4
rat: replaced -4.5688535576352e-4 by -262978/575588595 = -4.568853557635206e-4
rat: replaced -5.618675264007778e-4 by -150595/268025812 = -5.618675264007782e-4
rat: replaced -6.817933857540259e-4 by -192316/282073725 = -6.817933857540258e-4
rat: replaced -8.176509330039827e-4 by -105841/129445214 = -8.176509330039812e-4
rat: replaced -9.704265741758145e-4 by -651321/671169790 = -9.704265741758132e-4
rat: replaced -0.001141105023499428 by -1259907/1104111343 = -0.001141105023499428
rat: replaced -0.001330669204938795 by -1231154/925214167 = -0.001330669204938796
rat: replaced -0.001540100153900437 by -276884/179783113 = -0.001540100153900439
rat: replaced -0.001770376919130678 by -644389/363984072 = -0.001770376919130681
rat: replaced -0.002022476464811601 by -1271955/628909667 = -0.002022476464811599
rat: replaced -0.002297373572865413 by -1020913/444382669 = -0.002297373572865417
rat: replaced -0.002596040745477063 by -1097643/422814242 = -0.002596040745477065
rat: replaced -0.002919448107844891 by -906221/310408326 = -0.002919448107844891
```

rat: replaced  $-0.003268563311168871$  by  $-1379071/421919623 = -0.003268563311168867$   
rat: replaced  $-0.003644351435886262$  by  $-5966577/1637212301 = -0.003644351435886261$   
rat: replaced  $-0.004047774895164447$  by  $-572425/141417202 = -0.004047774895164451$   
rat: replaced  $-0.004479793338660443$  by  $-2952779/659132861 = -0.004479793338660444$   
rat: replaced  $-0.0049413635565565$  by  $-2524919/510976165 = -0.004941363556556498$   
rat: replaced  $-0.005433439383882244$  by  $-1361584/250593391 = -0.005433439383882235$   
rat: replaced  $-0.005956971605131645$  by  $-1447422/242979503 = -0.005956971605131648$   
rat: replaced  $-0.006512907859185624$  by  $-3695063/567344584 = -0.006512907859185626$   
rat: replaced  $-0.007102192544548636$  by  $-1363981/192050693 = -0.007102192544548642$   
rat: replaced  $-0.007725766724910044$  by  $-1464384/189545459 = -0.007725766724910038$   
rat: replaced  $-0.00838456803503801$  by  $-1113589/132814117 = -0.008384568035038023$   
rat: replaced  $-0.009079530587017326$  by  $-433906/47789475 = -0.00907953058701733$   
rat: replaced  $-0.009811584876838586$  by  $-1363090/138926587 = -0.009811584876838586$   
rat: replaced  $-0.0105816576913495$  by  $-1163729/109976058 = -0.01058165769134951$   
rat: replaced  $-0.01139067201557714$  by  $-13426050/1178688139 = -0.01139067201557714$   
rat: replaced  $-0.01223954694042984$  by  $-2283101/186534764 = -0.01223954694042983$   
rat: replaced  $-0.01312919757078923$  by  $-3499615/266552086 = -0.01312919757078922$   
rat: replaced  $-0.01406053493400045$  by  $-2280713/162206702 = -0.01406053493400045$

rat: replaced -0.01503446588876983 by -200490/13335359 = -0.01503446588876985  
rat: replaced -0.01605189303448024 by -951971/59305840 = -0.01605189303448025  
rat: replaced -0.01711371462093175 by -9432386/551159477 = -0.01711371462093176  
rat: replaced -0.01822082445851714 by -2559788/140486947 = -0.01822082445851713  
rat: replaced -0.01937411182884202 by -2983799/154009589 = -0.01937411182884203  
rat: replaced -0.02057446139579705 by -7167743/348380590 = -0.02057446139579705  
rat: replaced -0.02182275311709253 by -7415562/339808729 = -0.02182275311709253  
rat: replaced -0.02311986215626333 by -2988661/129268115 = -0.02311986215626336  
rat: replaced -0.02446665879515308 by -1991976/81415939 = -0.02446665879515312  
rat: replaced -0.02586400834688696 by -5000736/193347293 = -0.02586400834688697  
rat: replaced -0.02731277106934082 by -858413/31428997 = -0.02731277106934084  
rat: replaced -0.02881380207911666 by -3754753/130310918 = -0.02881380207911666  
rat: replaced -0.03036795126603076 by -4118329/135614318 = -0.03036795126603077  
rat: replaced -0.03197606320812652 by -3497683/109384416 = -0.03197606320812647  
rat: replaced -0.0336389770872163 by -3971799/118071337 = -0.03363897708721635  
rat: replaced -0.03535752660496472 by -1815732/51353479 = -0.03535752660496478  
rat: replaced -0.03713253989951881 by -3333721/89778965 = -0.03713253989951878

rat: replaced  $-0.03896483946269502$  by  $-8785771/225479461 = -0.03896483946269501$

rat: replaced  $-0.0408552420577305$  by  $-3189084/78058135 = -0.04085524205773043$

rat: replaced  $-0.04280455863760801$  by  $-7646593/178639688 = -0.04280455863760801$

rat: replaced  $-0.04481359426396048$  by  $-20610430/459914683 = -0.04481359426396048$

rat: replaced  $-0.04688314802656623$  by  $-3439140/73355569 = -0.04688314802656633$

rat: replaced  $-0.04901401296344043$  by  $-4006732/81746663 = -0.04901401296344048$

rat: replaced  $-0.05120697598153157$  by  $-4148974/81023609 = -0.0512069759815315$

rat: replaced  $-0.05346281777803219$  by  $-11998448/224426031 = -0.05346281777803218$

rat: replaced  $-0.05578231276230905$  by  $-1398019/25062048 = -0.05578231276230897$

rat: replaced  $-0.05816622897846346$  by  $-4451048/76522891 = -0.05816622897846345$

rat: replaced  $-0.06061532802852698$  by  $-2146337/35409146 = -0.06061532802852686$

rat: replaced  $-0.0631303649963022$  by  $-14651447/232082406 = -0.06313036499630222$

rat: replaced  $-0.06571208837185505$  by  $-4240309/64528599 = -0.06571208837185509$

rat: replaced  $-0.06836123997666599$  by  $-2716643/39739522 = -0.06836123997666604$

rat: replaced  $-0.07107855488944881$  by  $-3146673/44270357 = -0.07107855488944893$

rat: replaced  $-0.07386476137264342$  by  $-12898997/174629915 = -0.0738647613726434$

rat: replaced  $-0.07672058079958999$  by  $-5073506/66129661 = -0.07672058079959007$

rat: replaced  $-0.07964672758239233$  by  $-5672399/71219486 = -0.07964672758239227$

rat: replaced -0.08264390910047736 by -4686067/56701904 = -0.08264390910047748  
rat: replaced -0.0857128256298576 by -3585977/41837111 = -0.08571282562985766  
rat: replaced -0.08885417027310427 by -5751353/64728003 = -0.0888541702731042  
rat: replaced -0.09206862889003742 by -7305460/79347983 = -0.09206862889003745  
rat: replaced -0.09535688002914089 by -5971998/62627867 = -0.09535688002914103  
rat: replaced -0.0987195948597075 by -9821211/99485933 = -0.09871959485970745  
rat: replaced -0.1021574371047232 by -8336413/81603584 = -0.1021574371047232  
rat: replaced -0.1056710629744951 by -5741011/54329074 = -0.105671062974495  
rat: replaced -0.1092611211010309 by -5551873/50812887 = -0.1092611211010309  
rat: replaced -0.1129282524731764 by -11548693/102265755 = -0.1129282524731764  
rat: replaced -0.1166730903725168 by -5656228/48479285 = -0.1166730903725168  
rat: replaced -0.1204962603100498 by -4057613/33674182 = -0.12049626031005  
rat: replaced -0.1243983799636342 by -7966447/64039797 = -0.1243983799636342  
rat: replaced -0.1283800591162231 by -796346/6203035 = -0.1283800591162229  
rat: replaced -0.1324418995948859 by -4716124/35609003 = -0.1324418995948862  
rat: replaced -0.1365844952106265 by -612971/4487852 = -0.1365844952106264  
rat: replaced -0.140808431699002 by -10431632/74083859 = -0.1408084316990021

```

rat: replaced -0.1451142866615502 by -3554077/24491572 = -0.1451142866615504

rat: replaced -0.1495026295080298 by -26759297/178988805 = -0.1495026295080298

rat: replaced -0.1539740213994798 by -16145763/104860306 = -0.1539740213994798
part: invalid index of list or matrix.
#0: lineIntersection(g=[a1,a2,a1^2/2+a2^2/2],h=[-a,0,-a^2/2])
-- an error. To debug this try: debugmode(true);

Error in:
U &= lineIntersection(h,g); ...

```

Kami mendapatkan rumus untuk jari-jari lingkaran.

```
>&assume(a>0,b>0,c>0); $distance(U,B) | radcan
```

Mari kita tambahkan ini ke plot.

```
>plotPoint(U()); ...
>plotCircle(circleWithCenter(mxmeval("U"),mxmeval("distance(U,C)"))):
```

```

Function U not found.
Try list ... to find functions!
Error in:
plotPoint(U()); plotCircle(circleWithCenter(mxmeval("U"),mxmev ...

```

Menggunakan geometri, kami memperoleh rumus sederhana

$$\frac{a}{\sin(\alpha)} = 2r$$

untuk radiusnya. Kami dapat memeriksa, apakah ini benar dengan Maxima. Maxima akan memfaktorkan ini hanya jika kita kuadratkan.

```
>$c^2/sin(computeAngle(A,B,C))^2  | factor
```

#### Contoh 4: Garis Euler dan Parabola

---

Garis Euler adalah garis yang ditentukan dari sembarang segitiga yang tidak sama sisi. Ini adalah garis tengah segitiga, dan melewati beberapa titik penting yang ditentukan dari segitiga, termasuk orthocenter, circumcenter, centroid, titik Exeter dan pusat lingkaran sembilan titik segitiga.

Untuk demonstrasi, kami menghitung dan memplot garis Euler dalam sebuah segitiga.

Pertama, kita mendefinisikan sudut-sudut segitiga di Euler. Kami menggunakan definisi, yang terlihat dalam ekspresi simbolis.

```
>A:=[-1,-1]; B:=[2,0]; C:=[1,2];
```

Untuk memplot objek geometris, kami menyiapkan area plot, dan menambahkan titik ke sana. Semua plot objek geometris ditambahkan ke plot saat ini.

```
>setPlotRange(3); plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C");
```

Kita juga bisa menambahkan sisi segitiga.

```
>plotSegment(A,B,""); plotSegment(B,C,""); plotSegment(C,A,"");
```

Berikut adalah luas segitiga, menggunakan rumus determinan. Tentu saja, kita harus mengambil nilai absolut dari hasil ini.

```
>$areaTriangle(A,B,C)
```

Kita dapat menghitung koefisien sisi c.

```
>c &= lineThrough(A,B)
```

[ - 1, 3, - 2 ]

Dan juga dapatkan rumus untuk baris ini.

```
>$getLineEquation(c,x,y)
```

Untuk bentuk Hesse, kita perlu menentukan sebuah titik, sehingga titik tersebut berada di sisi positif dari bentuk Hesse. Memasukkan titik menghasilkan jarak positif ke garis.

```
>$getHesseForm(c,x,y,C), $at(%,[x=C[1],y=C[2]])
```

Sekarang kita hitung lingkaran luar ABC.

```
>LL &= circleThrough(A,B,C); $getCircleEquation(LL,x,y)
```

Maxima said:

```
rat: replaced -5.049958083474387e-5 by -102157/2022927682 = -5.049958083474385e-5
```

```
rat: replaced -2.039932534230044e-4 by -284619/1395237319 = -2.039932534230043e-4
```

```
rat: replaced -4.634656435254722e-4 by -573493/1237401322 = -4.634656435254721e-4
```

```
rat: replaced -8.31890779119604e-4 by -332331/399488741 = -8.318907791196046e-4
```

```
rat: replaced -0.001312231792998733 by -448125/341498356 = -0.001312231792998734
```

rat: replaced -0.001907440626462018 by -276030/144712237 = -0.001907440626462018

rat: replaced -0.002620457734122131 by -2586613/987084419 = -0.002620457734122131

rat: replaced -0.00345421178986248 by -3402379/984994322 = -0.00345421178986248

rat: replaced -0.004411619393972596 by -966955/219183686 = -0.004411619393972597

rat: replaced -0.005495584781489732 by -2798484/509224061 = -0.005495584781489734

rat: replaced -0.006708999531778753 by -6054060/902378957 = -0.006708999531778753

rat: replaced -0.008054742279375651 by -806546/100133061 = -0.00805474227937564

rat: replaced -0.009535678426127348 by -4115324/431571181 = -0.009535678426127346

rat: replaced -0.01115465985465333 by -2266398/203179481 = -0.01115465985465334

rat: replaced -0.01291452464316009 by -2106925/163143829 = -0.01291452464316012

rat: replaced -0.01481809678163515 by -2779203/187554653 = -0.01481809678163516

rat: replaced -0.01686818588945119 by -7427428/440321683 = -0.01686818588945119

rat: replaced -0.01906758693440599 by -2278085/119474216 = -0.019067586934406

rat: replaced -0.02141907995322798 by -2316386/108145915 = -0.02141907995322801

rat: replaced -0.02392542977357476 by -1665518/69612877 = -0.02392542977357479

rat: replaced -0.02658938573755304 by -3678645/138350131 = -0.02658938573755308

rat: replaced -0.02941368142678652 by -4053557/137811957 = -0.0294136814267865

rat: replaced -0.03240103438906003 by -2629160/81144323 = -0.03240103438906009

rat: replaced  $-0.03555414586656669$  by  $-1834427/51595305 = -0.03555414586656674$   
rat: replaced  $-0.03887570052578646$  by  $-1643964/42287701 = -0.03887570052578645$   
rat: replaced  $-0.04236836618902146$  by  $-3055464/72116635 = -0.04236836618902143$   
rat: replaced  $-0.04603479356761608$  by  $-3139251/68193007 = -0.0460347935676161$   
rat: replaced  $-0.04987761599688789$  by  $-5203437/104324092 = -0.04987761599688785$   
rat: replaced  $-0.05389944917279615$  by  $-4533622/84112585 = -0.0538994491727962$   
rat: replaced  $-0.05810289089037535$  by  $-11687290/201148167 = -0.05810289089037535$   
rat: replaced  $-0.06249052078395612$  by  $-3949243/63197473 = -0.06249052078395603$   
rat: replaced  $-0.0670649000692059$  by  $-3281728/48933615 = -0.067064900069206$   
rat: replaced  $-0.07182857128700804$  by  $-4146139/57722699 = -0.07182857128700791$   
rat: replaced  $-0.07678405804921068$  by  $-1198255/15605518 = -0.07678405804921054$   
rat: replaced  $-0.08193386478626702$  by  $-5956639/72700574 = -0.08193386478626702$   
rat: replaced  $-0.08728047649679532$  by  $-2799808/32078285 = -0.08728047649679527$   
rat: replaced  $-0.09282635849907966$  by  $-10292829/110882611 = -0.09282635849907972$   
rat: replaced  $-0.09857395618454184$  by  $-4198057/42587892 = -0.09857395618454184$   
rat: replaced  $-0.1045256947732028$  by  $-30563827/292404916 = -0.1045256947732028$   
rat: replaced  $-0.1106839790711635$  by  $-8949559/80856860 = -0.1106839790711635$

rat: replaced  $-0.1170511932301264$  by  $-9911603/84677505 = -0.1170511932301265$   
rat: replaced  $-0.1236297005089814$  by  $-19561703/158228184 = -0.1236297005089814$   
rat: replaced  $-0.1304218430374826$  by  $-4975231/38147222 = -0.1304218430374825$   
rat: replaced  $-0.137429941582038$  by  $-3502939/25488907 = -0.137429941582038$   
rat: replaced  $-0.1446562953136327$  by  $-15521432/107298697 = -0.1446562953136327$   
rat: replaced  $-0.1521031815779155$  by  $-18080502/118869979 = -0.1521031815779155$   
rat: replaced  $-0.1597728556674664$  by  $-37419026/234201397 = -0.1597728556674664$   
rat: replaced  $-0.1676675505962674$  by  $-22585897/134706429 = -0.1676675505962674$   
rat: replaced  $-0.1757894768764047$  by  $-15940893/90681725 = -0.1757894768764048$   
rat: replaced  $-0.1841408222970185$  by  $-7944795/43145213 = -0.1841408222970182$   
rat: replaced  $-0.1927237517055264$  by  $-1392861/7227241 = -0.1927237517055264$   
rat: replaced  $-0.2015404067911402$  by  $-1735485/8611102 = -0.2015404067911401$   
rat: replaced  $-0.2105929058706983$  by  $-10627754/50465869 = -0.2105929058706985$   
rat: replaced  $-0.2198833436768368$  by  $-9372347/42624179 = -0.2198833436768366$   
rat: replaced  $-0.2294137911485169$  by  $-7405273/32279110 = -0.2294137911485168$   
rat: replaced  $-0.239186295223934$  by  $-27692337/115777273 = -0.239186295223934$   
rat: replaced  $-0.2492028786358237$  by  $-8925310/35815437 = -0.249202878635824$   
rat: replaced  $-0.2594655397091927$  by  $-11150701/42975653 = -0.259465539709193$

rat: replaced  $-0.2699762521614856$  by  $-11249087/41666950 = -0.2699762521614853$

rat: replaced  $-0.280736964905216$  by  $-12097010/43090193 = -0.2807369649052164$

rat: replaced  $-0.2917496018530771$  by  $-14831788/50837389 = -0.2917496018530771$

rat: replaced  $-0.3030160617255513$  by  $-18597622/61375037 = -0.3030160617255514$

rat: replaced  $-0.3145382178610399$  by  $-11102944/35299189 = -0.3145382178610392$

rat: replaced  $-0.3263179180285316$  by  $-13053510/40002431 = -0.3263179180285318$

rat: replaced  $-0.3383569842428258$  by  $-13796661/40775458 = -0.3383569842428257$

rat: replaced  $-0.3506572125823338$  by  $-33496033/95523582 = -0.3506572125823338$

rat: replaced  $-0.3632203730094723$  by  $-23086207/63559780 = -0.3632203730094724$

rat: replaced  $-0.376048209193667$  by  $-22674222/60296051 = -0.3760482091936668$

rat: replaced  $-0.3891424383369902$  by  $-33246815/85436107 = -0.3891424383369902$

rat: replaced  $-0.402504751002439$  by  $-7793813/19363282 = -0.4025047510024385$

rat: replaced  $-0.4161368109448825$  by  $-10481453/25187517 = -0.4161368109448819$

rat: replaced  $-0.4300402549446862$  by  $-19565443/45496771 = -0.4300402549446861$

rat: replaced  $-0.4442166926440365$  by  $-16102633/36249500 = -0.4442166926440365$

rat: replaced  $-0.4586677063859775$  by  $-19404529/42306290 = -0.4586677063859771$

rat: replaced  $-0.4733948510561774$  by  $-10262860/21679281 = -0.4733948510561766$

rat: replaced  $-0.4883996539274416$  by  $-4159841/8517289 = -0.488399653927441$   
rat: replaced  $-0.5036836145069872$  by  $-13202363/26211619 = -0.5036836145069864$   
rat: replaced  $-0.5192482043864929$  by  $-12221370/23536663 = -0.5192482043864927$   
rat: replaced  $-0.5350948670949413$  by  $-52965833/98984005 = -0.5350948670949413$   
rat: replaced  $-0.551225017954267$  by  $-14288533/25921416 = -0.551225017954266$   
rat: replaced  $-0.5676400439378262$  by  $-25565995/45039097 = -0.5676400439378259$   
rat: replaced  $-0.5843413035316997$  by  $-18888222/32323955 = -0.5843413035316997$   
rat: replaced  $-0.6013301265988455$  by  $-5789399/9627655 = -0.6013301265988447$   
rat: replaced  $-0.6186078142461149$  by  $-4803773/7765458 = -0.618607814246114$   
rat: replaced  $-0.6361756386941407$  by  $-13914515/21872128 = -0.6361756386941407$   
rat: replaced  $-0.6540348431501183$  by  $-48160581/73636109 = -0.6540348431501181$   
rat: replaced  $-0.6721866416834846$  by  $-6617334/9844489 = -0.6721866416834841$   
rat: replaced  $-0.690632219104513$  by  $-16840135/24383651 = -0.6906322191045139$   
rat: replaced  $-0.7093727308458327$  by  $-29189494/41148317 = -0.7093727308458326$   
rat: replaced  $-0.7284093028468864$  by  $-13153959/18058472 = -0.7284093028468854$   
rat: replaced  $-0.7477430314413382$  by  $-12236470/16364539 = -0.7477430314413379$   
rat: replaced  $-0.7673749832474404$  by  $-39576757/51574208 = -0.7673749832474402$   
rat: replaced  $-0.7873061950613714$  by  $-6818881/8661028 = -0.7873061950613714$

rat: replaced  $-0.8075376737535601$  by  $-20498953/25384516 = -0.807537673753559$

rat: replaced  $-0.8280703961679966$  by  $-6989671/8440914 = -0.8280703961679979$

rat: replaced  $-0.84890530902455$  by  $-25231431/29722315 = -0.8489053090245494$

rat: replaced  $-0.8700433288242969$  by  $-9721738/11173855 = -0.8700433288242957$

rat: replaced  $-0.8914853417578728$  by  $-33469619/37543656 = -0.8914853417578725$

rat: replaced  $-0.9132322036168524$  by  $-21961040/24047597 = -0.913232203616852$

rat: replaced  $-1.503320816708814e-4$  by  $-144269/959668744 = -1.503320816708812e-4$

rat: replaced  $-6.026466136005715e-4$  by  $-554629/920322105 = -6.026466136005719e-4$

rat: replaced  $-0.001358898348046048$  by  $-845667/622318072 = -0.001358898348046045$

rat: replaced  $-0.002421011643797932$  by  $-1882206/777446075 = -0.002421011643797931$

rat: replaced  $-0.003790880273744496$  by  $-1271426/335390703 = -0.003790880273744499$

rat: replaced  $-0.005470367235498236$  by  $-1827828/334132595 = -0.005470367235498231$

rat: replaced  $-0.007461304565095722$  by  $-763704/102355291 = -0.007461304565095712$

rat: replaced  $-0.009765493153796295$  by  $-1498127/153410276 = -0.009765493153796295$

rat: replaced  $-0.01238470256799509$  by  $-4127047/333237474 = -0.0123847025679951$

rat: replaced  $-0.01532067087226624$  by  $-2377723/155197055 = -0.01532067087226623$

rat: replaced  $-0.01857510445555993$  by  $-4650073/250338996 = -0.01857510445555993$

rat: replaced -0.02214967786056252 by -1400096/63210671 = -0.0221496778605625  
rat: replaced -0.02604603361624602 by -2361294/90658487 = -0.02604603361624599  
rat: replaced -0.03026578207361535 by -1660222/54854753 = -0.03026578207361539  
rat: replaced -0.03481050124467483 by -4348324/124914145 = -0.03481050124467489  
rat: replaced -0.03968173664462726 by -9988401/251712799 = -0.03968173664462728  
rat: replaced -0.04488100113732568 by -9262688/206383275 = -0.04488100113732569  
rat: replaced -0.05040977478398728 by -8488548/168390913 = -0.0504097747839873  
rat: replaced -0.05626950469518793 by -3521587/62584290 = -0.05626950469518788  
rat: replaced -0.06246160488615271 by -1724725/27612563 = -0.06246160488615272  
rat: replaced -0.06898745613535606 by -699061/10133161 = -0.06898745613535599  
rat: replaced -0.07584840584644481 by -7595322/100138189 = -0.07584840584644485  
rat: replaced -0.08304576791349888 by -2278089/27431729 = -0.083045767913499  
rat: replaced -0.09058082258964217 by -5411518/59742425 = -0.09058082258964212  
rat: replaced -0.09845481635901993 by -9650917/98023818 = -0.09845481635902001  
rat: replaced -0.1066689618121501 by -6098878/57175751 = -0.10666896181215  
rat: replaced -0.1152244375246662 by -22751561/197454303 = -0.1152244375246662  
rat: replaced -0.1241223879394598 by -36591823/294804375 = -0.1241223879394599  
rat: replaced -0.1333639232522373 by -2621363/19655713 = -0.1333639232522371

rat: replaced -0.1429501193005029 by -14344978/100349535 = -0.142950119300503  
rat: replaced -0.1528820174559729 by -7599779/49710091 = -0.1528820174559729  
rat: replaced -0.163160624520442 by -10213526/62597983 = -0.1631606245204418  
rat: replaced -0.1737869126251026 by -16586501/95441600 = -0.1737869126251027  
rat: replaced -0.1847618191333327 by -5716910/30942053 = -0.1847618191333329  
rat: replaced -0.1960862465469606 by -4485287/22874052 = -0.1960862465469607  
rat: replaced -0.2077610624160157 by -6172898/29711525 = -0.2077610624160153  
rat: replaced -0.2197870992519729 by -9779676/44496133 = -0.2197870992519732  
rat: replaced -0.2321651544445043 by -12225287/52657717 = -0.2321651544445043  
rat: replaced -0.2448959901817382 by -6355852/25953271 = -0.2448959901817385  
rat: replaced -0.257980333374044 by -7376119/28591788 = -0.2579803333740443  
rat: replaced -0.2714188755813393 by -7521403/27711422 = -0.2714188755813397  
rat: replaced -0.2852122729439353 by -33856936/118707851 = -0.2852122729439353  
rat: replaced -0.2993611461169232 by -11605920/38768959 = -0.2993611461169231  
rat: replaced -0.3138660802081108 by -12935015/41211892 = -0.3138660802081108  
rat: replaced -0.3287276247195093 by -4721149/14361887 = -0.3287276247195093  
rat: replaced -0.3439462934923849 by -29044045/84443547 = -0.3439462934923849

rat: replaced  $-0.359522564655877$  by  $-32233940/89657627 = -0.359522564655877$   
rat: replaced  $-0.3754568805791821$  by  $-37000079/98546813 = -0.3754568805791822$   
rat: replaced  $-0.39174964782732$  by  $-19372366/49450883 = -0.3917496478273199$   
rat: replaced  $-0.4084012371204762$  by  $-18093351/44302880 = -0.4084012371204762$   
rat: replaced  $-0.4254119832969315$  by  $-17508065/41155552 = -0.4254119832969316$   
rat: replaced  $-0.4427821852795774$  by  $-7324893/16542881 = -0.4427821852795774$   
rat: replaced  $-0.4605121060460234$  by  $-10003471/21722493 = -0.4605121060460233$   
rat: replaced  $-0.4786019726023018$  by  $-60236280/125858821 = -0.4786019726023018$   
rat: replaced  $-0.4970519759601649$  by  $-15490512/31164773 = -0.497051975960165$   
rat: replaced  $-0.5158622711179853$  by  $-5589426/10835113 = -0.5158622711179847$   
rat: replaced  $-0.5350329770452558$  by  $-7868837/14707200 = -0.5350329770452568$   
rat: replaced  $-0.5545641766706926$  by  $-12815301/23108779 = -0.554564176670693$   
rat: replaced  $-0.5744559168739426$  by  $-13141311/22876100 = -0.5744559168739427$   
rat: replaced  $-0.5947082084808951$  by  $-14675891/24677465 = -0.5947082084808955$   
rat: replaced  $-0.6153210262625995$  by  $-19715525/32041039 = -0.6153210262626003$   
rat: replaced  $-0.6362943089377887$  by  $-24594815/38653206 = -0.636294308937789$   
rat: replaced  $-0.6576279591790061$  by  $-15639887/23782272 = -0.6576279591790053$   
rat: replaced  $-0.6793218436223387$  by  $-17047367/25094684 = -0.6793218436223385$

rat: replaced  $-0.701375792880754$  by  $-24937969/35555788 = -0.701375792880754$

rat: replaced  $-0.7237896015610379$  by  $-12792911/17674903 = -0.7237896015610382$

rat: replaced  $-0.7465630282843339$  by  $-24894563/33345561 = -0.7465630282843344$

rat: replaced  $-0.76969579571028$  by  $-11030405/14330863 = -0.7696957957102792$

rat: replaced  $-0.7931875905647454$  by  $-17983947/22673006 = -0.7931875905647447$

rat: replaced  $-0.8170380636711536$  by  $-35512807/43465303 = -0.817038063671154$

rat: replaced  $-0.8412468299854033$  by  $-9419201/11196715 = -0.841246829985402$

rat: replaced  $-0.8658134686343698$  by  $-33824443/39066663 = -0.8658134686343699$

rat: replaced  $-0.8907375229579941$  by  $-1798799/2019449 = -0.890737522957995$

rat: replaced  $-0.9160185005549473$  by  $-21232969/23179629 = -0.9160185005549485$

rat: replaced  $-0.9416558733318703$  by  $-11919739/12658275 = -0.9416558733318718$

rat: replaced  $-0.967649077556183$  by  $-17561986/18149127 = -0.9676490775561821$

rat: replaced  $-0.9939975139124576$  by  $-37819354/38047735 = -0.9939975139124576$

rat: replaced  $-1.020700547562349$  by  $-18875311/18492506 = -1.020700547562348$

rat: replaced  $-1.047757508208077$  by  $-16616467/15859077 = -1.047757508208075$

rat: replaced  $-1.07516769015946$  by  $-21797467/20273551 = -1.07516769015946$

rat: replaced  $-1.102930352404475$  by  $-30072842/27266311 = -1.102930352404474$

rat: replaced  $-1.131044718683369$  by  $-7906291/6990255 = -1.131044718683367$   
rat: replaced  $-1.159509977566275$  by  $-26421893/22787120 = -1.159509977566274$   
rat: replaced  $-1.188325282534358$  by  $-19245269/16195287 = -1.188325282534357$   
rat: replaced  $-1.21748975206447$  by  $-18221771/14966673 = -1.21748975206447$   
rat: replaced  $-1.247002469717292$  by  $-21540268/17273637 = -1.247002469717292$   
rat: replaced  $-1.276862484228988$  by  $-17179259/13454275 = -1.27686248422899$   
rat: replaced  $-1.307068809606323$  by  $-22169845/16961498 = -1.307068809606321$   
rat: replaced  $-1.337620425225263$  by  $-48765573/36456959 = -1.337620425225263$   
rat: replaced  $-1.368516275933041$  by  $-117856634/86120009 = -1.368516275933041$   
rat: replaced  $-1.399755272153666$  by  $-29694085/21213769 = -1.399755272153666$   
rat: replaced  $-1.431336289996881$  by  $-23110861/16146353 = -1.43133628999688$   
rat: replaced  $-1.463258171370553$  by  $-19245288/13152353 = -1.463258171370553$   
rat: replaced  $-1.495519724096479$  by  $-43164951/28862843 = -1.495519724096479$   
rat: replaced  $-1.528119722029604$  by  $-31224680/20433399 = -1.528119722029605$   
rat: replaced  $-1.561056905180636$  by  $-23576644/15103001 = -1.561056905180633$   
rat: replaced  $-1.594329979842039$  by  $-26705354/16750205 = -1.594329979842038$   
rat: replaced  $-1.627937618717409$  by  $-52734804/32393627 = -1.62793761871741$   
rat: replaced  $-1.661878461054199$  by  $-42611978/25640851 = -1.661878461054198$

```
part: invalid index of list or matrix.  
#0: lineIntersection(g=[-3,-1,-1],h=[-2,-3,-3/2])  
#1: circleThrough(a=[-1,-1],b=[2,0],c=[1,2])  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
LL &= circleThrough(A,B,C); $getCircleEquation(LL,x,y) ...  
^
```

```
>0 &= getCircleCenter(LL); $0
```

Gambarkan lingkaran dan pusatnya. Cu dan U adalah simbolis. Kami mengevaluasi ekspresi ini untuk Euler.

```
>plotCircle(LL()); plotPoint(0(),"O"):
```

```
Function LL not found.  
Try list ... to find functions!  
Error in:  
plotCircle(LL()); plotPoint(0(),"O"): ...  
^
```

Kita dapat menghitung perpotongan ketinggian di ABC (orthocenter) secara numerik dengan perintah berikut.

```
>H &= lineIntersection(perpendicular(A,lineThrough(C,B)),...  
> perpendicular(B,lineThrough(A,C))); $H
```

```
Maxima said:  
rat: replaced -9.983250083613754e-5 by -612914/6139423483 = -9.983250083613756e-5  
  
rat: replaced -3.986533601775671e-4 by -220554/553247563 = -3.986533601775666e-4  
  
rat: replaced -8.954327045205754e-4 by -584699/652979277 = -8.954327045205756e-4  
  
rat: replaced -0.001589120864678328 by -740868/466212493 = -0.00158912086467833  
  
rat: replaced -0.002478648480745763 by -878917/354595259 = -0.002478648480745762  
  
rat: replaced -0.003562926609036218 by -2735717/767828614 = -0.003562926609036219  
  
rat: replaced -0.004840846830973591 by -1164348/240525685 = -0.004840846830973582  
  
rat: replaced -0.006311281363933816 by -16515210/2616776063 = -0.006311281363933816  
  
rat: replaced -0.007973083174022497 by -2414321/302808957 = -0.007973083174022491  
  
rat: replaced -0.009825086090776508 by -1144049/116441626 = -0.009825086090776506  
  
rat: replaced -0.01186610492378118 by -1659683/139867548 = -0.01186610492378118  
  
rat: replaced -0.01409493558118687 by -986877/70016425 = -0.01409493558118684  
  
rat: replaced -0.01651035519011868 by -1738361/105289134 = -0.01651035519011867  
  
rat: replaced -0.01911112221896202 by -1475047/77182647 = -0.01911112221896199  
  
rat: replaced -0.02189597660151474 by -7711274/352177669 = -0.02189597660151473
```

rat: replaced -0.02486363986299212 by -3887839/156366446 = -0.02486363986299209  
rat: replaced -0.0280128152478745 by -2263313/80795628 = -0.02801281524787455  
rat: replaced -0.03134218784958129 by -1116362/35618509 = -0.03134218784958124  
rat: replaced -0.03485042474195996 by -3920507/112495243 = -0.03485042474195998  
rat: replaced -0.03853617511257795 by -5379408/139593719 = -0.03853617511257795  
rat: replaced -0.04239807039780302 by -3385918/79860191 = -0.04239807039780308  
rat: replaced -0.04643472441965829 by -10918553/235137672 = -0.04643472441965828  
rat: replaced -0.05064473352443885 by -5036501/99447675 = -0.05064473352443886  
rat: replaced -0.05502667672307548 by -2932521/53292715 = -0.05502667672307557  
rat: replaced -0.05957911583323347 by -6320819/106091185 = -0.05957911583323346  
rat: replaced -0.06430059562312868 by -9893260/153859539 = -0.0643005956231287  
rat: replaced -0.06918964395705007 by -6012189/86894348 = -0.06918964395705  
rat: replaced -0.07424477194257195 by -6096479/82113243 = -0.07424477194257204  
rat: replaced -0.07946447407944118 by -5389689/67825139 = -0.07946447407944125  
rat: replaced -0.0848472284101276 by -9595393/113090235 = -0.08484722841012754  
rat: replaced -0.09039149667201674 by -3773144/41742245 = -0.09039149667201657  
rat: replaced -0.0960957244512361 by -5162056/53717853 = -0.09609572445123597

rat: replaced  $-0.1019583413380946$  by  $-1082663/10618680 = -0.1019583413380948$   
rat: replaced  $-0.107977761084122$  by  $-1922059/17800508 = -0.1079777610841219$   
rat: replaced  $-0.1141523817606936$  by  $-5923297/51889386 = -0.1141523817606938$   
rat: replaced  $-0.1204805859192203$  by  $-17634703/146369665 = -0.1204805859192204$   
rat: replaced  $-0.1269607407528933$  by  $-11368220/89541223 = -0.1269607407528932$   
rat: replaced  $-0.1335911982599624$  by  $-4657902/34866833 = -0.1335911982599624$   
rat: replaced  $-0.1403702954085355$  by  $-8528456/60756843 = -0.1403702954085353$   
rat: replaced  $-0.1472963543028805$  by  $-11128453/75551449 = -0.1472963543028804$   
rat: replaced  $-0.1543676823512128$  by  $-8170760/52930509 = -0.1543676823512126$   
rat: replaced  $-0.1615825724349539$  by  $-188109817/1164171446 = -0.1615825724349539$   
rat: replaced  $-0.1689393030794406$  by  $-5046974/29874481 = -0.1689393030794409$   
rat: replaced  $-0.1764361386260728$  by  $-6530305/37012287 = -0.176436138626073$   
rat: replaced  $-0.1840713294058766$  by  $-25189859/136848357 = -0.1840713294058766$   
rat: replaced  $-0.1918431119144694$  by  $-24326967/126806570 = -0.1918431119144694$   
rat: replaced  $-0.1997497089884105$  by  $-14902039/74603558 = -0.1997497089884104$   
rat: replaced  $-0.2077893299829148$  by  $-7281351/35041987 = -0.2077893299829145$   
rat: replaced  $-0.2159601709509153$  by  $-11348921/52550991 = -0.2159601709509151$   
rat: replaced  $-0.2242604148234577$  by  $-22385730/99820247 = -0.2242604148234576$

```
rat: replaced -0.2326882315914051 by -25615030/110083049 = -0.2326882315914051
rat: replaced -0.2412417784884371 by -14523232/60201977 = -0.2412417784884373
rat: replaced -0.2499192001753251 by -11309023/45250717 = -0.2499192001753254
rat: replaced -0.2587186289254649 by -7582961/29309683 = -0.2587186289254647
rat: replaced -0.267638184811648 by -17912865/66929407 = -0.2676381848116479
rat: replaced -0.2766759758940514 by -27538925/99534934 = -0.2766759758940514
rat: replaced -0.2858300984094321 by -29258587/102363562 = -0.2858300984094321
rat: replaced -0.2950986369614998 by -7877677/26695064 = -0.2950986369614997
rat: replaced -0.304479664712457 by -14469542/47522195 = -0.304479664712457
rat: replaced -0.3139712435756791 by -8375733/26676752 = -0.3139712435756797
rat: replaced -0.3235714244095225 by -178371467/551258404 = -0.3235714244095225
rat: replaced -0.3332782472122374 by -5743591/17233621 = -0.333278247212237
rat: replaced -0.3430897413179662 by -15588245/45434891 = -0.3430897413179664
rat: replaced -0.3530039255938071 by -6523425/18479752 = -0.3530039255938067
rat: replaced -0.3630188086379282 by -51253958/141188161 = -0.3630188086379282
rat: replaced -0.373132388978704 by -9370061/25111894 = -0.3731323889787047
rat: replaced -0.3833426552748616 by -11820697/30835851 = -0.3833426552748617
```

rat: replaced  $-0.393647586516613$  by  $-9153768/23253713 = -0.3936475865166135$   
rat: replaced  $-0.4040451522277552$  by  $-16634707/41170416 = -0.404045152227755$   
rat: replaced  $-0.4145333126687146$  by  $-2088920/5039209 = -0.4145333126687145$   
rat: replaced  $-0.4251100190405208$  by  $-24667763/58026774 = -0.4251100190405209$   
rat: replaced  $-0.4357732136896836$  by  $-10448574/23977091 = -0.435773213689684$   
rat: replaced  $-0.4465208303139576$  by  $-8346266/18691773 = -0.4465208303139568$   
rat: replaced  $-0.4573507941689697$  by  $-20158688/44077081 = -0.4573507941689696$   
rat: replaced  $-0.4682610222756929$  by  $-12818601/27374905 = -0.4682610222756937$   
rat: replaced  $-0.4792494236287415$  by  $-13652513/28487281 = -0.4792494236287416$   
rat: replaced  $-0.4903138994054704$  by  $-35114711/71616797 = -0.4903138994054705$   
rat: replaced  $-0.5014523431758559$  by  $-15102855/30118226 = -0.5014523431758564$   
rat: replaced  $-0.5126626411131362$  by  $-31697340/61828847 = -0.5126626411131361$   
rat: replaced  $-0.5239426722051925$  by  $-27432767/52358337 = -0.5239426722051924$   
rat: replaced  $-0.5352903084666492$  by  $-6124470/11441399 = -0.5352903084666482$   
rat: replaced  $-0.5467034151516694$  by  $-41717397/76307182 = -0.5467034151516694$   
rat: replaced  $-0.5581798509674292$  by  $-7494380/13426461 = -0.5581798509674292$   
rat: replaced  $-0.5697174682882435$  by  $-14609183/25642856 = -0.5697174682882438$   
rat: replaced  $-0.581314113370329$  by  $-14367580/24715691 = -0.5813141133703282$

rat: replaced  $-0.5929676265671738$  by  $-9820294/16561265 = -0.5929676265671735$   
rat: replaced  $-0.6046758425455033$  by  $-23593213/39017952 = -0.6046758425455031$   
rat: replaced  $-0.6164365905018095$  by  $-15720181/25501700 = -0.6164365905018097$   
rat: replaced  $-0.6282476943794307$  by  $-53974636/85912987 = -0.6282476943794306$   
rat: replaced  $-0.640106973086155$  by  $-20459615/31962806 = -0.6401069730861552$   
rat: replaced  $-0.652012240712328$  by  $-51645100/79208789 = -0.652012240712328$   
rat: replaced  $-0.6639613067494411$  by  $-12215999/18398661 = -0.6639613067494422$   
rat: replaced  $-0.6759519763091814$  by  $-18558734/27455699 = -0.6759519763091808$   
rat: replaced  $-0.6879820503429186$  by  $-23500536/34158647 = -0.687982050342919$   
rat: replaced  $-0.7000493258616074$  by  $-29992669/42843651 = -0.7000493258616078$   
rat: replaced  $-0.7121515961560857$  by  $-10685401/15004391 = -0.7121515961560853$   
rat: replaced  $-0.7242866510177421$  by  $-11795807/16286103 = -0.7242866510177419$   
rat: replaced  $-0.7364522769595366$  by  $-14940657/20287339 = -0.7364522769595362$   
rat: replaced  $-0.7486462574373463$  by  $-42508133/56779998 = -0.7486462574373461$   
rat: replaced  $1.503320816708814e-4$  by  $144269/959668744 = 1.503320816708812e-4$   
rat: replaced  $6.026466136005715e-4$  by  $554629/920322105 = 6.026466136005719e-4$   
rat: replaced  $0.001358898348046048$  by  $845667/622318072 = 0.001358898348046045$

rat: replaced 0.002421011643797932 by 1882206/777446075 = 0.002421011643797931  
rat: replaced 0.003790880273744496 by 1271426/335390703 = 0.003790880273744499  
rat: replaced 0.005470367235498236 by 1827828/334132595 = 0.005470367235498231  
rat: replaced 0.007461304565095722 by 763704/102355291 = 0.007461304565095712  
rat: replaced 0.009765493153796295 by 1498127/153410276 = 0.009765493153796295  
rat: replaced 0.01238470256799509 by 4127047/333237474 = 0.0123847025679951  
rat: replaced 0.01532067087226624 by 2377723/155197055 = 0.01532067087226623  
rat: replaced 0.01857510445555993 by 4650073/250338996 = 0.01857510445555993  
rat: replaced 0.02214967786056252 by 1400096/63210671 = 0.0221496778605625  
rat: replaced 0.02604603361624602 by 2361294/90658487 = 0.02604603361624599  
rat: replaced 0.03026578207361535 by 1660222/54854753 = 0.03026578207361539  
rat: replaced 0.03481050124467483 by 4348324/124914145 = 0.03481050124467489  
rat: replaced 0.03968173664462726 by 9988401/251712799 = 0.03968173664462728  
rat: replaced 0.04488100113732568 by 9262688/206383275 = 0.04488100113732569  
rat: replaced 0.05040977478398728 by 8488548/168390913 = 0.0504097747839873  
rat: replaced 0.05626950469518793 by 3521587/62584290 = 0.05626950469518788  
rat: replaced 0.06246160488615271 by 1724725/27612563 = 0.06246160488615272  
rat: replaced 0.06898745613535606 by 699061/10133161 = 0.06898745613535599

rat: replaced 0.07584840584644481 by 7595322/100138189 = 0.07584840584644485  
rat: replaced 0.08304576791349888 by 2278089/27431729 = 0.083045767913499  
rat: replaced 0.09058082258964217 by 5411518/59742425 = 0.09058082258964212  
rat: replaced 0.09845481635901993 by 9650917/98023818 = 0.09845481635902001  
rat: replaced 0.1066689618121501 by 6098878/57175751 = 0.10666896181215  
rat: replaced 0.1152244375246662 by 22751561/197454303 = 0.1152244375246662  
rat: replaced 0.1241223879394598 by 36591823/294804375 = 0.1241223879394599  
rat: replaced 0.1333639232522373 by 2621363/19655713 = 0.1333639232522371  
rat: replaced 0.1429501193005029 by 14344978/100349535 = 0.142950119300503  
rat: replaced 0.1528820174559729 by 7599779/49710091 = 0.1528820174559729  
rat: replaced 0.163160624520442 by 10213526/62597983 = 0.1631606245204418  
rat: replaced 0.1737869126251026 by 16586501/95441600 = 0.1737869126251027  
rat: replaced 0.1847618191333327 by 5716910/30942053 = 0.1847618191333329  
rat: replaced 0.1960862465469606 by 4485287/22874052 = 0.1960862465469607  
rat: replaced 0.2077610624160157 by 6172898/29711525 = 0.2077610624160153  
rat: replaced 0.2197870992519729 by 9779676/44496133 = 0.2197870992519732  
rat: replaced 0.2321651544445043 by 12225287/52657717 = 0.2321651544445043

rat: replaced 0.2448959901817382 by  $6355852/25953271 = 0.2448959901817385$   
rat: replaced 0.257980333374044 by  $7376119/28591788 = 0.2579803333740443$   
rat: replaced 0.2714188755813393 by  $7521403/27711422 = 0.2714188755813397$   
rat: replaced 0.2852122729439353 by  $33856936/118707851 = 0.2852122729439353$   
rat: replaced 0.2993611461169232 by  $11605920/38768959 = 0.2993611461169231$   
rat: replaced 0.3138660802081108 by  $12935015/41211892 = 0.3138660802081108$   
rat: replaced 0.3287276247195093 by  $4721149/14361887 = 0.3287276247195093$   
rat: replaced 0.3439462934923849 by  $29044045/84443547 = 0.3439462934923849$   
rat: replaced 0.359522564655877 by  $32233940/89657627 = 0.359522564655877$   
rat: replaced 0.3754568805791821 by  $37000079/98546813 = 0.3754568805791822$   
rat: replaced 0.39174964782732 by  $19372366/49450883 = 0.3917496478273199$   
rat: replaced 0.4084012371204762 by  $18093351/44302880 = 0.4084012371204762$   
rat: replaced 0.4254119832969315 by  $17508065/41155552 = 0.4254119832969316$   
rat: replaced 0.4427821852795774 by  $7324893/16542881 = 0.4427821852795774$   
rat: replaced 0.4605121060460234 by  $10003471/21722493 = 0.4605121060460233$   
rat: replaced 0.4786019726023018 by  $60236280/125858821 = 0.4786019726023018$   
rat: replaced 0.4970519759601649 by  $15490512/31164773 = 0.497051975960165$   
rat: replaced 0.5158622711179853 by  $5589426/10835113 = 0.5158622711179847$

rat: replaced 0.5350329770452558 by 7868837/14707200 = 0.5350329770452568

rat: replaced 0.5545641766706926 by 12815301/23108779 = 0.554564176670693

rat: replaced 0.5744559168739426 by 13141311/22876100 = 0.5744559168739427

rat: replaced 0.5947082084808951 by 14675891/24677465 = 0.5947082084808955

rat: replaced 0.6153210262625995 by 19715525/32041039 = 0.6153210262626003

rat: replaced 0.6362943089377887 by 24594815/38653206 = 0.636294308937789

rat: replaced 0.6576279591790061 by 15639887/23782272 = 0.6576279591790053

rat: replaced 0.6793218436223387 by 17047367/25094684 = 0.6793218436223385

rat: replaced 0.701375792880754 by 24937969/35555788 = 0.701375792880754

rat: replaced 0.7237896015610379 by 12792911/17674903 = 0.7237896015610382

rat: replaced 0.7465630282843339 by 24894563/33345561 = 0.7465630282843344

rat: replaced 0.76969579571028 by 11030405/14330863 = 0.7696957957102792

rat: replaced 0.7931875905647454 by 17983947/22673006 = 0.7931875905647447

rat: replaced 0.8170380636711536 by 35512807/43465303 = 0.817038063671154

rat: replaced 0.8412468299854033 by 9419201/11196715 = 0.841246829985402

rat: replaced 0.8658134686343698 by 33824443/39066663 = 0.8658134686343699

rat: replaced 0.8907375229579941 by 1798799/2019449 = 0.890737522957995

rat: replaced 0.9160185005549473 by  $21232969/23179629 = 0.9160185005549485$   
rat: replaced 0.9416558733318703 by  $11919739/12658275 = 0.9416558733318718$   
rat: replaced 0.967649077556183 by  $17561986/18149127 = 0.9676490775561821$   
rat: replaced 0.9939975139124576 by  $37819354/38047735 = 0.9939975139124576$   
rat: replaced 1.020700547562349 by  $18875311/18492506 = 1.020700547562348$   
rat: replaced 1.047757508208077 by  $16616467/15859077 = 1.047757508208075$   
rat: replaced 1.07516769015946 by  $21797467/20273551 = 1.07516769015946$   
rat: replaced 1.102930352404475 by  $30072842/27266311 = 1.102930352404474$   
rat: replaced 1.131044718683369 by  $7906291/6990255 = 1.131044718683367$   
rat: replaced 1.159509977566275 by  $26421893/22787120 = 1.159509977566274$   
rat: replaced 1.188325282534358 by  $19245269/16195287 = 1.188325282534357$   
rat: replaced 1.21748975206447 by  $18221771/14966673 = 1.21748975206447$   
rat: replaced 1.247002469717292 by  $21540268/17273637 = 1.247002469717292$   
rat: replaced 1.276862484228988 by  $17179259/13454275 = 1.27686248422899$   
rat: replaced 1.307068809606323 by  $22169845/16961498 = 1.307068809606321$   
rat: replaced 1.337620425225263 by  $48765573/36456959 = 1.337620425225263$   
rat: replaced 1.368516275933041 by  $117856634/86120009 = 1.368516275933041$   
rat: replaced 1.399755272153666 by  $29694085/21213769 = 1.399755272153666$

```

rat: replaced 1.431336289996881 by 23110861/16146353 = 1.43133628999688

rat: replaced 1.463258171370553 by 19245288/13152353 = 1.463258171370553

rat: replaced 1.495519724096479 by 43164951/28862843 = 1.495519724096479

rat: replaced 1.528119722029604 by 31224680/20433399 = 1.528119722029605

rat: replaced 1.561056905180636 by 23576644/15103001 = 1.561056905180633

rat: replaced 1.594329979842039 by 26705354/16750205 = 1.594329979842038

rat: replaced 1.627937618717409 by 52734804/32393627 = 1.62793761871741

rat: replaced 1.661878461054199 by 42611978/25640851 = 1.661878461054198
part: invalid index of list or matrix.
#0: lineIntersection(g=[1,-2,1],h=[2,3,4])
-- an error. To debug this try: debugmode(true);

Error in:
perpendicular(B,lineThrough(A,C))); $H ...

```

Sekarang kita dapat menghitung garis Euler dari segitiga.

```
>el &= lineThrough(H,O); $getLineEquation(el,x,y)
```

Tambahkan ke plot kami.

```
>plotPoint(H(),"H"); plotLine(el(),"Garis Euler"):
```

```
Function H needs at least 3 arguments!
Use: H (a, b, c)
Error in:
plotPoint(H(),"H"); plotLine(el(),"Garis Euler"): ...
^
```

Pusat gravitasi harus berada di garis ini.

```
>M &= (A+B+C)/3; $getLineEquation(el,x,y) with [x=M[1],y=M[2]]
>plotPoint(M(),"M"): // titik berat
```

Teorinya memberitahu kita  $MH=2*MO$ . Kita perlu menyederhanakan dengan radcan untuk mencapai ini.

```
>$distance(M,H)/distance(M,O)|radcan
```

Fungsi termasuk fungsi untuk sudut juga.

```
>$computeAngle(A,C,B), degprint(%())
```

60°15'18.43'',

Persamaan untuk pusat incircle tidak terlalu bagus.

```
>Q &= lineIntersection(angleBisector(A,C,B),angleBisector(C,B,A))|radcan; $Q
```

```
Maxima said:  
rat: replaced 1.66665833335744e-7 by 15819/94914474571 = 1.66665833335744e-7  
rat: replaced 4.999958333473664e-5 by 201389/4027813565 = 4.99995833347366e-5  
rat: replaced 1.33330666692022e-6 by 31771/23828726570 = 1.333306666920221e-6  
rat: replaced 1.999933334222437e-4 by 200030/1000183339 = 1.999933334222437e-4  
rat: replaced 4.499797504338432e-6 by 24036/5341573699 = 4.499797504338431e-6  
rat: replaced 4.499662510124569e-4 by 1162901/2584418270 = 4.499662510124571e-4  
rat: replaced 1.066581336583994e-5 by 58861/5518660226 = 1.066581336583993e-5  
rat: replaced 7.998933390220841e-4 by 1137431/1421978337 = 7.998933390220838e-4  
rat: replaced 2.083072932167196e-5 by 35635/1710693824 = 2.0830729321672e-5  
rat: replaced 0.001249739605033717 by 567943/454449069 = 0.001249739605033716
```

```
rat: replaced 3.599352055540239e-5 by 98277/2730408098 = 3.599352055540234e-5
rat: replaced 0.00179946006479581 by 479561/266502719 = 0.001799460064795812
rat: replaced 5.71526624672386e-5 by 51154/895041417 = 5.715266246723866e-5
rat: replaced 0.002448999746720415 by 1946227/794702818 = 0.002448999746720415
rat: replaced 8.530603082730626e-5 by 121691/1426522824 = 8.530603082730627e-5
rat: replaced 0.003198293697380561 by 2986741/933854512 = 0.003198293697380562
rat: replaced 1.214508019889565e-4 by 158455/1304684674 = 1.214508019889563e-4
rat: replaced 0.004047266988005727 by 2125334/525128193 = 0.004047266988005727
rat: replaced 1.665833531718508e-4 by 142521/855553675 = 1.66583353171851e-4
rat: replaced 0.004995834721974179 by 1957223/391770967 = 0.004995834721974179
rat: replaced 2.216991628251896e-4 by 179571/809975995 = 2.216991628251896e-4
rat: replaced 0.006043902043303184 by 1800665/297930871 = 0.006043902043303193
rat: replaced 2.877927110806339e-4 by 1167733/4057548906 = 2.877927110806339e-4
rat: replaced 0.00719136414613375 by 2476362/344352191 = 0.007191364146133747
rat: replaced 3.658573803051457e-4 by 386279/1055818526 = 3.658573803051454e-4
rat: replaced 0.00843810628521191 by 2079855/246483622 = 0.008438106285211924
rat: replaced 4.5688535576352e-4 by 262978/575588595 = 4.568853557635206e-4
rat: replaced 0.009784003787362772 by 1752551/179124113 = 0.009784003787362787
```

```
rat: replaced 5.618675264007778e-4 by 150595/268025812 = 5.618675264007782e-4
rat: replaced 0.01122892206395776 by 5450241/485375263 = 0.01122892206395776
rat: replaced 6.817933857540259e-4 by 192316/282073725 = 6.817933857540258e-4
rat: replaced 0.01277271662437307 by 3258991/255152533 = 0.01277271662437308
rat: replaced 8.176509330039827e-4 by 105841/129445214 = 8.176509330039812e-4
rat: replaced 0.01441523309043924 by 2330472/161667313 = 0.01441523309043925
rat: replaced 9.704265741758145e-4 by 651321/671169790 = 9.704265741758132e-4
rat: replaced 0.01615630721187855 by 19391318/1200232067 = 0.01615630721187855
rat: replaced 0.001141105023499428 by 1259907/1104111343 = 0.001141105023499428
rat: replaced 0.01799576488272969 by 4765614/264818641 = 0.01799576488272969
rat: replaced 0.001330669204938795 by 1231154/925214167 = 0.001330669204938796
rat: replaced 0.01993342215875837 by 2504519/125644206 = 0.01993342215875836
rat: replaced 0.001540100153900437 by 276884/179783113 = 0.001540100153900439
rat: replaced 0.02196908527585173 by 1298306/59096953 = 0.0219690852758517
rat: replaced 0.001770376919130678 by 644389/363984072 = 0.001770376919130681
rat: replaced 0.02410255066939448 by 2001286/83032125 = 0.02410255066939453
rat: replaced 0.002022476464811601 by 1271955/628909667 = 0.002022476464811599
```

rat: replaced 0.02633360499462523 by 2978115/113091808 = 0.02633360499462525  
rat: replaced 0.002297373572865413 by 1020913/444382669 = 0.002297373572865417  
rat: replaced 0.02866202514797045 by 1770713/61779061 = 0.02866202514797044  
rat: replaced 0.002596040745477063 by 1097643/422814242 = 0.002596040745477065  
rat: replaced 0.03108757828935527 by 5034207/161936287 = 0.03108757828935525  
rat: replaced 0.002919448107844891 by 906221/310408326 = 0.002919448107844891  
rat: replaced 0.03361002186548678 by 4553215/135471944 = 0.03361002186548678  
rat: replaced 0.003268563311168871 by 1379071/421919623 = 0.003268563311168867  
rat: replaced 0.03622910363410947 by 3082649/85087642 = 0.0362291036341094  
rat: replaced 0.003644351435886262 by 5966577/1637212301 = 0.003644351435886261  
rat: replaced 0.03894456168922911 by 4913415/126164342 = 0.03894456168922911  
rat: replaced 0.004047774895164447 by 572425/141417202 = 0.004047774895164451  
rat: replaced 0.04175612448730281 by 1734727/41544253 = 0.04175612448730273  
rat: replaced 0.004479793338660443 by 2952779/659132861 = 0.004479793338660444  
rat: replaced 0.04466351087439402 by 4691119/105032473 = 0.04466351087439405  
rat: replaced 0.0049413635565565 by 2524919/510976165 = 0.004941363556556498  
rat: replaced 0.04766643011428662 by 3536207/74186529 = 0.04766643011428665  
rat: replaced 0.005433439383882244 by 1361584/250593391 = 0.005433439383882235

rat: replaced 0.05076458191755917 by 7710025/151878036 = 0.05076458191755916  
rat: replaced 0.005956971605131645 by 1447422/242979503 = 0.005956971605131648  
rat: replaced 0.0539576564716131 by 3377975/62604183 = 0.05395765647161309  
rat: replaced 0.006512907859185624 by 3695063/567344584 = 0.006512907859185626  
rat: replaced 0.05724533447165381 by 2560865/44734912 = 0.05724533447165382  
rat: replaced 0.007102192544548636 by 1363981/192050693 = 0.007102192544548642  
rat: replaced 0.06062728715262111 by 8274761/136485754 = 0.06062728715262107  
rat: replaced 0.007725766724910044 by 1464384/189545459 = 0.007725766724910038  
rat: replaced 0.06410317632206519 by 5287663/82486755 = 0.06410317632206528  
rat: replaced 0.00838456803503801 by 1113589/132814117 = 0.008384568035038023  
rat: replaced 0.06767265439396564 by 2921400/43169579 = 0.06767265439396572  
rat: replaced 0.009079530587017326 by 433906/47789475 = 0.00907953058701733  
rat: replaced 0.07133536442348987 by 7236103/101437808 = 0.07133536442348991  
rat: replaced 0.009811584876838586 by 1363090/138926587 = 0.009811584876838586  
rat: replaced 0.07509094014268702 by 9209133/122639735 = 0.07509094014268704  
rat: replaced 0.0105816576913495 by 1163729/109976058 = 0.01058165769134951  
rat: replaced 0.07893900599711501 by 5197067/65836489 = 0.07893900599711506

rat: replaced 0.01139067201557714 by  $13426050/1178688139 = 0.01139067201557714$   
rat: replaced 0.08287917718339499 by  $11217158/135343501 = 0.082879177183395$   
rat: replaced 0.01223954694042984 by  $2283101/186534764 = 0.01223954694042983$   
rat: replaced 0.08691105968769186 by  $5213115/59982182 = 0.08691105968769192$   
rat: replaced 0.01312919757078923 by  $3499615/266552086 = 0.01312919757078922$   
rat: replaced 0.09103425032511492 by  $5893225/64736349 = 0.09103425032511488$   
rat: replaced 0.01406053493400045 by  $2280713/162206702 = 0.01406053493400045$   
rat: replaced 0.09524833678003664 by  $9601787/100807923 = 0.09524833678003662$   
rat: replaced 0.01503446588876983 by  $200490/13335359 = 0.01503446588876985$   
rat: replaced 0.09955289764732322 by  $5687088/57126293 = 0.09955289764732328$   
rat: replaced 0.01605189303448024 by  $951971/59305840 = 0.01605189303448025$   
rat: replaced 0.1039475024744748 by  $10260011/98703776 = 0.1039475024744747$   
rat: replaced 0.01711371462093175 by  $9432386/551159477 = 0.01711371462093176$   
rat: replaced 0.1084317118046711 by  $14939691/137779721 = 0.1084317118046712$   
rat: replaced 0.01822082445851714 by  $2559788/140486947 = 0.01822082445851713$   
rat: replaced 0.113005077220716 by  $8478529/75027859 = 0.1130050772207161$   
rat: replaced 0.01937411182884202 by  $2983799/154009589 = 0.01937411182884203$   
rat: replaced 0.1176671413898787 by  $7123715/60541243 = 0.1176671413898786$

rat: replaced 0.02057446139579705 by 7167743/348380590 = 0.02057446139579705

rat: replaced 0.1224174381096274 by 12172179/99431741 = 0.1224174381096274

rat: replaced 0.02182275311709253 by 7415562/339808729 = 0.02182275311709253

rat: replaced 0.1272554923542488 by 7277933/57191504 = 0.127255492354249

rat: replaced 0.02311986215626333 by 2988661/129268115 = 0.02311986215626336

rat: replaced 0.1321808203223502 by 3633064/27485561 = 0.1321808203223503

rat: replaced 0.02446665879515308 by 1991976/81415939 = 0.02446665879515312

rat: replaced 0.1371929294852391 by 56235017/409897341 = 0.1371929294852391

rat: replaced 0.02586400834688696 by 5000736/193347293 = 0.02586400834688697

rat: replaced 0.1422913186361759 by 9349741/65708443 = 0.1422913186361759

rat: replaced 0.02731277106934082 by 858413/31428997 = 0.02731277106934084

rat: replaced 0.1474754779404944 by 1549881/10509415 = 0.1474754779404943

rat: replaced 0.02881380207911666 by 3754753/130310918 = 0.02881380207911666

rat: replaced 0.152744888986584 by 5264425/34465474 = 0.1527448889865841

rat: replaced 0.03036795126603076 by 4118329/135614318 = 0.03036795126603077

rat: replaced 0.1580990248377314 by 5442776/34426373 = 0.1580990248377312

rat: replaced 0.03197606320812652 by 3497683/109384416 = 0.03197606320812647

rat: replaced 0.1635373500848132 by 12328488/75386375 = 0.1635373500848131  
rat: replaced 0.0336389770872163 by 3971799/118071337 = 0.03363897708721635  
rat: replaced 0.1690593208998367 by 20896917/123607009 = 0.1690593208998367  
rat: replaced 0.03535752660496472 by 1815732/51353479 = 0.03535752660496478  
rat: replaced 0.1746643850903219 by 2841592/16268869 = 0.1746643850903219  
rat: replaced 0.03713253989951881 by 3333721/89778965 = 0.03713253989951878  
rat: replaced 0.1803519821545206 by 4461007/24735004 = 0.1803519821545208  
rat: replaced 0.03896483946269502 by 8785771/225479461 = 0.03896483946269501  
rat: replaced 0.1861215433374662 by 4381209/23539505 = 0.1861215433374661  
rat: replaced 0.0408552420577305 by 3189084/78058135 = 0.04085524205773043  
rat: replaced 0.1919724916878484 by 72809759/379271834 = 0.1919724916878484  
rat: replaced 0.04280455863760801 by 7646593/178639688 = 0.04280455863760801  
rat: replaced 0.1979042421157076 by 26318167/132984350 = 0.1979042421157076  
rat: replaced 0.04481359426396048 by 20610430/459914683 = 0.04481359426396048  
rat: replaced 0.2039162014509444 by 8519416/41779005 = 0.2039162014509441  
rat: replaced 0.04688314802656623 by 3439140/73355569 = 0.04688314802656633  
rat: replaced 0.2100077685026351 by 50962787/242670961 = 0.2100077685026351  
rat: replaced 0.04901401296344043 by 4006732/81746663 = 0.04901401296344048

```
rat: replaced 0.216178334119151 by 1347531/6233423 = 0.2161783341191509
rat: replaced 0.05120697598153157 by 4148974/81023609 = 0.0512069759815315
rat: replaced 0.2224272812490723 by 23234851/104460437 = 0.2224272812490723
rat: replaced 0.05346281777803219 by 11998448/224426031 = 0.05346281777803218
rat: replaced 0.2287539850028937 by 8185268/35781969 = 0.2287539850028935
rat: replaced 0.05578231276230905 by 1398019/25062048 = 0.05578231276230897
rat: replaced 0.2351578127155118 by 12642104/53760085 = 0.2351578127155119
rat: replaced 0.05816622897846346 by 4451048/76522891 = 0.05816622897846345
rat: replaced 0.2416381240094921 by 8002142/33116223 = 0.2416381240094923
rat: replaced 0.06061532802852698 by 2146337/35409146 = 0.06061532802852686
rat: replaced 0.2481942708591053 by 8882901/35790113 = 0.2481942708591057
rat: replaced 0.0631303649963022 by 14651447/232082406 = 0.06313036499630222
rat: replaced 0.2548255976551299 by 868346/3407609 = 0.25482559765513
rat: replaced 0.06571208837185505 by 4240309/64528599 = 0.06571208837185509
rat: replaced 0.2615314412704124 by 8212450/31401387 = 0.2615314412704127
rat: replaced 0.06836123997666599 by 2716643/39739522 = 0.06836123997666604
rat: replaced 0.2683111311261794 by 34459769/128432126 = 0.2683111311261794
```

rat: replaced  $0.07107855488944881$  by  $3146673/44270357 = 0.07107855488944893$   
rat: replaced  $0.2751639892590951$  by  $12552159/45617012 = 0.2751639892590949$   
rat: replaced  $0.07386476137264342$  by  $12898997/174629915 = 0.0738647613726434$   
rat: replaced  $0.2820893303890569$  by  $11134456/39471383 = 0.2820893303890568$   
rat: replaced  $0.07672058079958999$  by  $5073506/66129661 = 0.07672058079959007$   
rat: replaced  $0.2890864619877229$  by  $9583357/33150487 = 0.2890864619877228$   
rat: replaced  $0.07964672758239233$  by  $5672399/71219486 = 0.07964672758239227$   
rat: replaced  $0.2961546843477643$  by  $11052271/37319251 = 0.2961546843477647$   
rat: replaced  $0.08264390910047736$  by  $4686067/56701904 = 0.08264390910047748$   
rat: replaced  $0.3032932906528349$  by  $9918077/32701274 = 0.3032932906528351$   
rat: replaced  $0.0857128256298576$  by  $3585977/41837111 = 0.08571282562985766$   
rat: replaced  $0.3105015670482534$  by  $9320011/30015987 = 0.3105015670482533$   
rat: replaced  $0.08885417027310427$  by  $5751353/64728003 = 0.0888541702731042$   
rat: replaced  $0.3177787927123868$  by  $248395525/781661743 = 0.3177787927123868$   
rat: replaced  $0.09206862889003742$  by  $7305460/79347983 = 0.09206862889003745$   
rat: replaced  $0.3251242399287333$  by  $13842845/42577093 = 0.3251242399287335$   
rat: replaced  $0.09535688002914089$  by  $5971998/62627867 = 0.09535688002914103$   
rat: replaced  $0.3325371741586922$  by  $9318229/28021616 = 0.3325371741586923$

```
rat: replaced 0.0987195948597075 by 9821211/99485933 = 0.09871959485970745
rat: replaced 0.3400168541150183 by 13391981/39386227 = 0.3400168541150184
rat: replaced 0.1021574371047232 by 8336413/81603584 = 0.1021574371047232
rat: replaced 0.3475625318359485 by 10097818/29053241 = 0.347562531835949
rat: replaced 0.1056710629744951 by 5741011/54329074 = 0.105671062974495
rat: replaced 0.3551734527599992 by 15867851/44676343 = 0.3551734527599987
rat: replaced 0.1092611211010309 by 5551873/50812887 = 0.1092611211010309
rat: replaced 0.3628488558014202 by 6897641/19009681 = 0.3628488558014203
rat: replaced 0.1129282524731764 by 11548693/102265755 = 0.1129282524731764
rat: replaced 0.3705879734263036 by 23358661/63031352 = 0.3705879734263038
rat: replaced 0.1166730903725168 by 5656228/48479285 = 0.1166730903725168
rat: replaced 0.3783900317293359 by 14241382/37636779 = 0.3783900317293358
rat: replaced 0.1204962603100498 by 4057613/33674182 = 0.12049626031005
rat: replaced 0.3862542505111889 by 3461217/8960981 = 0.3862542505111884
rat: replaced 0.1243983799636342 by 7966447/64039797 = 0.1243983799636342
rat: replaced 0.3941798433565377 by 5314214/13481699 = 0.3941798433565384
rat: replaced 0.1283800591162231 by 796346/6203035 = 0.1283800591162229
```

```
rat: replaced 0.4021660177127022 by 11567173/28762184 = 0.4021660177127022
rat: replaced 0.1324418995948859 by 4716124/35609003 = 0.1324418995948862
rat: replaced 0.4102119749689023 by 11320633/27597032 = 0.4102119749689024
rat: replaced 0.1365844952106265 by 612971/4487852 = 0.1365844952106264
rat: replaced 0.418316910536117 by 12225195/29224721 = 0.4183169105361177
rat: replaced 0.140808431699002 by 10431632/74083859 = 0.1408084316990021
rat: replaced 0.4264800139275439 by 7978696/18708253 = 0.4264800139275431
rat: replaced 0.1451142866615502 by 3554077/24491572 = 0.1451142866615504
rat: replaced 0.4347004688396462 by 20489554/47134879 = 0.4347004688396463
rat: replaced 0.1495026295080298 by 26759297/178988805 = 0.1495026295080298
rat: replaced 0.4429774532337832 by 23449796/52936771 = 0.4429774532337834
rat: replaced 0.1539740213994798 by 16145763/104860306 = 0.1539740213994798
rat: replaced 0.451310139418413 by 8841241/19590167 = 0.4513101394184133
rat: replaced 4.999958333473664e-5 by 201389/4027813565 = 4.99995833347366e-5
rat: replaced 1.66665833335744e-7 by 15819/94914474571 = 1.66665833335744e-7
rat: replaced 1.99993334222437e-4 by 200030/1000183339 = 1.99993334222437e-4
rat: replaced 1.33330666692022e-6 by 31771/23828726570 = 1.333306666920221e-6
rat: replaced 4.499662510124569e-4 by 1162901/2584418270 = 4.499662510124571e-4
```

```
rat: replaced 4.499797504338432e-6 by 24036/5341573699 = 4.499797504338431e-6
rat: replaced 7.998933390220841e-4 by 1137431/1421978337 = 7.998933390220838e-4
rat: replaced 1.066581336583994e-5 by 58861/5518660226 = 1.066581336583993e-5
rat: replaced 0.001249739605033717 by 567943/454449069 = 0.001249739605033716
rat: replaced 2.083072932167196e-5 by 35635/1710693824 = 2.0830729321672e-5
rat: replaced 0.00179946006479581 by 479561/266502719 = 0.001799460064795812
rat: replaced 3.599352055540239e-5 by 98277/2730408098 = 3.599352055540234e-5
rat: replaced 0.002448999746720415 by 1946227/794702818 = 0.002448999746720415
rat: replaced 5.71526624672386e-5 by 51154/895041417 = 5.715266246723866e-5
rat: replaced 0.003198293697380561 by 2986741/933854512 = 0.003198293697380562
rat: replaced 8.530603082730626e-5 by 121691/1426522824 = 8.530603082730627e-5
rat: replaced 0.004047266988005727 by 2125334/525128193 = 0.004047266988005727
rat: replaced 1.214508019889565e-4 by 158455/1304684674 = 1.214508019889563e-4
rat: replaced 0.004995834721974179 by 1957223/391770967 = 0.004995834721974179
rat: replaced 1.665833531718508e-4 by 142521/855553675 = 1.66583353171851e-4
rat: replaced 0.006043902043303184 by 1800665/297930871 = 0.006043902043303193
rat: replaced 2.216991628251896e-4 by 179571/809975995 = 2.216991628251896e-4
```

```
rat: replaced 0.00719136414613375 by 2476362/344352191 = 0.007191364146133747
rat: replaced 2.877927110806339e-4 by 1167733/4057548906 = 2.877927110806339e-4
rat: replaced 0.00843810628521191 by 2079855/246483622 = 0.008438106285211924
rat: replaced 3.658573803051457e-4 by 386279/1055818526 = 3.658573803051454e-4
rat: replaced 0.009784003787362772 by 1752551/179124113 = 0.009784003787362787
rat: replaced 4.5688535576352e-4 by 262978/575588595 = 4.568853557635206e-4
rat: replaced 0.01122892206395776 by 5450241/485375263 = 0.01122892206395776
rat: replaced 5.618675264007778e-4 by 150595/268025812 = 5.618675264007782e-4
rat: replaced 0.01277271662437307 by 3258991/255152533 = 0.01277271662437308
rat: replaced 6.817933857540259e-4 by 192316/282073725 = 6.817933857540258e-4
rat: replaced 0.01441523309043924 by 2330472/161667313 = 0.01441523309043925
rat: replaced 8.176509330039827e-4 by 105841/129445214 = 8.176509330039812e-4
rat: replaced 0.01615630721187855 by 19391318/1200232067 = 0.01615630721187855
rat: replaced 9.704265741758145e-4 by 651321/671169790 = 9.704265741758132e-4
rat: replaced 0.01799576488272969 by 4765614/264818641 = 0.01799576488272969
rat: replaced 0.001141105023499428 by 1259907/1104111343 = 0.001141105023499428
rat: replaced 0.01993342215875837 by 2504519/125644206 = 0.01993342215875836
rat: replaced 0.001330669204938795 by 1231154/925214167 = 0.001330669204938796
```

rat: replaced 0.02196908527585173 by 1298306/59096953 = 0.0219690852758517

rat: replaced 0.001540100153900437 by 276884/179783113 = 0.001540100153900439

rat: replaced 0.02410255066939448 by 2001286/83032125 = 0.02410255066939453

rat: replaced 0.001770376919130678 by 644389/363984072 = 0.001770376919130681

rat: replaced 0.02633360499462523 by 2978115/113091808 = 0.02633360499462525

rat: replaced 0.002022476464811601 by 1271955/628909667 = 0.002022476464811599

rat: replaced 0.02866202514797045 by 1770713/61779061 = 0.02866202514797044

rat: replaced 0.002297373572865413 by 1020913/444382669 = 0.002297373572865417

rat: replaced 0.03108757828935527 by 5034207/161936287 = 0.03108757828935525

rat: replaced 0.002596040745477063 by 1097643/422814242 = 0.002596040745477065

rat: replaced 0.03361002186548678 by 4553215/135471944 = 0.03361002186548678

rat: replaced 0.002919448107844891 by 906221/310408326 = 0.002919448107844891

rat: replaced 0.03622910363410947 by 3082649/85087642 = 0.0362291036341094

rat: replaced 0.003268563311168871 by 1379071/421919623 = 0.003268563311168867

rat: replaced 0.03894456168922911 by 4913415/126164342 = 0.03894456168922911

rat: replaced 0.003644351435886262 by 5966577/1637212301 = 0.003644351435886261

rat: replaced 0.04175612448730281 by 1734727/41544253 = 0.04175612448730273

rat: replaced 0.004047774895164447 by 572425/141417202 = 0.004047774895164451  
rat: replaced 0.04466351087439402 by 4691119/105032473 = 0.04466351087439405  
rat: replaced 0.004479793338660443 by 2952779/659132861 = 0.004479793338660444  
rat: replaced 0.04766643011428662 by 3536207/74186529 = 0.04766643011428665  
rat: replaced 0.0049413635565565 by 2524919/510976165 = 0.004941363556556498  
rat: replaced 0.05076458191755917 by 7710025/151878036 = 0.05076458191755916  
rat: replaced 0.005433439383882244 by 1361584/250593391 = 0.005433439383882235  
rat: replaced 0.0539576564716131 by 3377975/62604183 = 0.05395765647161309  
rat: replaced 0.005956971605131645 by 1447422/242979503 = 0.005956971605131648  
rat: replaced 0.05724533447165381 by 2560865/44734912 = 0.05724533447165382  
rat: replaced 0.006512907859185624 by 3695063/567344584 = 0.006512907859185626  
rat: replaced 0.06062728715262111 by 8274761/136485754 = 0.06062728715262107  
rat: replaced 0.007102192544548636 by 1363981/192050693 = 0.007102192544548642  
rat: replaced 0.06410317632206519 by 5287663/82486755 = 0.06410317632206528  
rat: replaced 0.007725766724910044 by 1464384/189545459 = 0.007725766724910038  
rat: replaced 0.06767265439396564 by 2921400/43169579 = 0.06767265439396572  
rat: replaced 0.00838456803503801 by 1113589/132814117 = 0.008384568035038023  
rat: replaced 0.07133536442348987 by 7236103/101437808 = 0.07133536442348991

rat: replaced 0.009079530587017326 by 433906/47789475 = 0.00907953058701733  
rat: replaced 0.07509094014268702 by 9209133/122639735 = 0.07509094014268704  
rat: replaced 0.009811584876838586 by 1363090/138926587 = 0.009811584876838586  
rat: replaced 0.07893900599711501 by 5197067/65836489 = 0.07893900599711506  
rat: replaced 0.0105816576913495 by 1163729/109976058 = 0.01058165769134951  
rat: replaced 0.08287917718339499 by 11217158/135343501 = 0.082879177183395  
rat: replaced 0.01139067201557714 by 13426050/1178688139 = 0.01139067201557714  
rat: replaced 0.08691105968769186 by 5213115/59982182 = 0.08691105968769192  
rat: replaced 0.01223954694042984 by 2283101/186534764 = 0.01223954694042983  
rat: replaced 0.09103425032511492 by 5893225/64736349 = 0.09103425032511488  
rat: replaced 0.01312919757078923 by 3499615/266552086 = 0.01312919757078922  
rat: replaced 0.09524833678003664 by 9601787/100807923 = 0.09524833678003662  
rat: replaced 0.01406053493400045 by 2280713/162206702 = 0.01406053493400045  
rat: replaced 0.09955289764732322 by 5687088/57126293 = 0.09955289764732328  
rat: replaced 0.01503446588876983 by 200490/13335359 = 0.01503446588876985  
rat: replaced 0.1039475024744748 by 10260011/98703776 = 0.1039475024744747  
rat: replaced 0.01605189303448024 by 951971/59305840 = 0.01605189303448025

rat: replaced 0.1084317118046711 by 14939691/137779721 = 0.1084317118046712  
rat: replaced 0.01711371462093175 by 9432386/551159477 = 0.01711371462093176  
rat: replaced 0.113005077220716 by 8478529/75027859 = 0.1130050772207161  
rat: replaced 0.01822082445851714 by 2559788/140486947 = 0.01822082445851713  
rat: replaced 0.1176671413898787 by 7123715/60541243 = 0.1176671413898786  
rat: replaced 0.01937411182884202 by 2983799/154009589 = 0.01937411182884203  
rat: replaced 0.1224174381096274 by 12172179/99431741 = 0.1224174381096274  
rat: replaced 0.02057446139579705 by 7167743/348380590 = 0.02057446139579705  
rat: replaced 0.1272554923542488 by 7277933/57191504 = 0.127255492354249  
rat: replaced 0.02182275311709253 by 7415562/339808729 = 0.02182275311709253  
rat: replaced 0.1321808203223502 by 3633064/27485561 = 0.1321808203223503  
rat: replaced 0.02311986215626333 by 2988661/129268115 = 0.02311986215626336  
rat: replaced 0.1371929294852391 by 56235017/409897341 = 0.1371929294852391  
rat: replaced 0.02446665879515308 by 1991976/81415939 = 0.02446665879515312  
rat: replaced 0.1422913186361759 by 9349741/65708443 = 0.1422913186361759  
rat: replaced 0.02586400834688696 by 5000736/193347293 = 0.02586400834688697  
rat: replaced 0.1474754779404944 by 1549881/10509415 = 0.1474754779404943  
rat: replaced 0.02731277106934082 by 858413/31428997 = 0.02731277106934084

rat: replaced 0.152744888986584 by 5264425/34465474 = 0.1527448889865841  
rat: replaced 0.02881380207911666 by 3754753/130310918 = 0.02881380207911666  
rat: replaced 0.1580990248377314 by 5442776/34426373 = 0.1580990248377312  
rat: replaced 0.03036795126603076 by 4118329/135614318 = 0.03036795126603077  
rat: replaced 0.1635373500848132 by 12328488/75386375 = 0.1635373500848131  
rat: replaced 0.03197606320812652 by 3497683/109384416 = 0.03197606320812647  
rat: replaced 0.1690593208998367 by 20896917/123607009 = 0.1690593208998367  
rat: replaced 0.0336389770872163 by 3971799/118071337 = 0.03363897708721635  
rat: replaced 0.1746643850903219 by 2841592/16268869 = 0.1746643850903219  
rat: replaced 0.03535752660496472 by 1815732/51353479 = 0.03535752660496478  
rat: replaced 0.1803519821545206 by 4461007/24735004 = 0.1803519821545208  
rat: replaced 0.03713253989951881 by 3333721/89778965 = 0.03713253989951878  
rat: replaced 0.1861215433374662 by 4381209/23539505 = 0.1861215433374661  
rat: replaced 0.03896483946269502 by 8785771/225479461 = 0.03896483946269501  
rat: replaced 0.1919724916878484 by 72809759/379271834 = 0.1919724916878484  
rat: replaced 0.0408552420577305 by 3189084/78058135 = 0.04085524205773043  
rat: replaced 0.1979042421157076 by 26318167/132984350 = 0.1979042421157076

rat: replaced 0.04280455863760801 by 7646593/178639688 = 0.04280455863760801  
rat: replaced 0.2039162014509444 by 8519416/41779005 = 0.2039162014509441  
rat: replaced 0.04481359426396048 by 20610430/459914683 = 0.04481359426396048  
rat: replaced 0.2100077685026351 by 50962787/242670961 = 0.2100077685026351  
rat: replaced 0.04688314802656623 by 3439140/73355569 = 0.04688314802656633  
rat: replaced 0.216178334119151 by 1347531/6233423 = 0.2161783341191509  
rat: replaced 0.04901401296344043 by 4006732/81746663 = 0.04901401296344048  
rat: replaced 0.2224272812490723 by 23234851/104460437 = 0.2224272812490723  
rat: replaced 0.05120697598153157 by 4148974/81023609 = 0.0512069759815315  
rat: replaced 0.2287539850028937 by 8185268/35781969 = 0.2287539850028935  
rat: replaced 0.05346281777803219 by 11998448/224426031 = 0.05346281777803218  
rat: replaced 0.2351578127155118 by 12642104/53760085 = 0.2351578127155119  
rat: replaced 0.05578231276230905 by 1398019/25062048 = 0.05578231276230897  
rat: replaced 0.2416381240094921 by 8002142/33116223 = 0.2416381240094923  
rat: replaced 0.05816622897846346 by 4451048/76522891 = 0.05816622897846345  
rat: replaced 0.2481942708591053 by 8882901/35790113 = 0.2481942708591057  
rat: replaced 0.06061532802852698 by 2146337/35409146 = 0.06061532802852686  
rat: replaced 0.2548255976551299 by 868346/3407609 = 0.25482559765513

```
rat: replaced 0.0631303649963022 by 14651447/232082406 = 0.06313036499630222
rat: replaced 0.2615314412704124 by 8212450/31401387 = 0.2615314412704127
rat: replaced 0.06571208837185505 by 4240309/64528599 = 0.06571208837185509
rat: replaced 0.2683111311261794 by 34459769/128432126 = 0.2683111311261794
rat: replaced 0.06836123997666599 by 2716643/39739522 = 0.06836123997666604
rat: replaced 0.2751639892590951 by 12552159/45617012 = 0.2751639892590949
rat: replaced 0.07107855488944881 by 3146673/44270357 = 0.07107855488944893
rat: replaced 0.2820893303890569 by 11134456/39471383 = 0.2820893303890568
rat: replaced 0.07386476137264342 by 12898997/174629915 = 0.0738647613726434
rat: replaced 0.2890864619877229 by 9583357/33150487 = 0.2890864619877228
rat: replaced 0.07672058079958999 by 5073506/66129661 = 0.07672058079959007
rat: replaced 0.2961546843477643 by 11052271/37319251 = 0.2961546843477647
rat: replaced 0.07964672758239233 by 5672399/71219486 = 0.07964672758239227
rat: replaced 0.3032932906528349 by 9918077/32701274 = 0.3032932906528351
rat: replaced 0.08264390910047736 by 4686067/56701904 = 0.08264390910047748
rat: replaced 0.3105015670482534 by 9320011/30015987 = 0.3105015670482533
rat: replaced 0.0857128256298576 by 3585977/41837111 = 0.08571282562985766
```

```
rat: replaced 0.3177787927123868 by 248395525/781661743 = 0.3177787927123868
rat: replaced 0.08885417027310427 by 5751353/64728003 = 0.0888541702731042
rat: replaced 0.3251242399287333 by 13842845/42577093 = 0.3251242399287335
rat: replaced 0.09206862889003742 by 7305460/79347983 = 0.09206862889003745
rat: replaced 0.3325371741586922 by 9318229/28021616 = 0.3325371741586923
rat: replaced 0.09535688002914089 by 5971998/62627867 = 0.09535688002914103
rat: replaced 0.3400168541150183 by 13391981/39386227 = 0.3400168541150184
rat: replaced 0.0987195948597075 by 9821211/99485933 = 0.09871959485970745
rat: replaced 0.3475625318359485 by 10097818/29053241 = 0.347562531835949
rat: replaced 0.1021574371047232 by 8336413/81603584 = 0.1021574371047232
rat: replaced 0.3551734527599992 by 15867851/44676343 = 0.3551734527599987
rat: replaced 0.1056710629744951 by 5741011/54329074 = 0.105671062974495
rat: replaced 0.3628488558014202 by 6897641/19009681 = 0.3628488558014203
rat: replaced 0.1092611211010309 by 5551873/50812887 = 0.1092611211010309
rat: replaced 0.3705879734263036 by 23358661/63031352 = 0.3705879734263038
rat: replaced 0.1129282524731764 by 11548693/102265755 = 0.1129282524731764
rat: replaced 0.3783900317293359 by 14241382/37636779 = 0.3783900317293358
rat: replaced 0.1166730903725168 by 5656228/48479285 = 0.1166730903725168
```

rat: replaced 0.3862542505111889 by 3461217/8960981 = 0.3862542505111884  
rat: replaced 0.1204962603100498 by 4057613/33674182 = 0.12049626031005  
rat: replaced 0.3941798433565377 by 5314214/13481699 = 0.3941798433565384  
rat: replaced 0.1243983799636342 by 7966447/64039797 = 0.1243983799636342  
rat: replaced 0.4021660177127022 by 11567173/28762184 = 0.4021660177127022  
rat: replaced 0.1283800591162231 by 796346/6203035 = 0.1283800591162229  
rat: replaced 0.4102119749689023 by 11320633/27597032 = 0.4102119749689024  
rat: replaced 0.1324418995948859 by 4716124/35609003 = 0.1324418995948862  
rat: replaced 0.418316910536117 by 12225195/29224721 = 0.4183169105361177  
rat: replaced 0.1365844952106265 by 612971/4487852 = 0.1365844952106264  
rat: replaced 0.4264800139275439 by 7978696/18708253 = 0.4264800139275431  
rat: replaced 0.140808431699002 by 10431632/74083859 = 0.1408084316990021  
rat: replaced 0.4347004688396462 by 20489554/47134879 = 0.4347004688396463  
rat: replaced 0.1451142866615502 by 3554077/24491572 = 0.1451142866615504  
rat: replaced 0.4429774532337832 by 23449796/52936771 = 0.4429774532337834  
rat: replaced 0.1495026295080298 by 26759297/178988805 = 0.1495026295080298  
rat: replaced 0.451310139418413 by 8841241/19590167 = 0.4513101394184133

```

rat: replaced 0.1539740213994798 by 16145763/104860306 = 0.1539740213994798
part: invalid index of list or matrix.
#0: lineIntersection(g=[-2-sqrt(13)/sqrt(5),-3-sqrt(13)/(2*sqrt(5))+sqrt(5)*sqrt(13)/2,sqrt(13)*(-
-- an error. To debug this try: debugmode(true);

Error in:
... angleBisector(A,C,B),angleBisector(C,B,A))|radcan; $Q ...
^
```

Mari kita hitung juga ekspresi untuk jari-jari lingkaran yang tertulis.

```
>r &= distance(Q,projectToLine(Q,lineThrough(A,B)))|ratsimp; $r
```

```

Maxima said:
rat: replaced 1.498320841708742e-4 by 1329822/8875415485 = 1.498320841708742e-4

rat: replaced 5.986466935998108e-4 by 398723/666040595 = 5.986466935998098e-4

rat: replaced 0.001345398955533032 by 4525441/3363642421 = 0.001345398955533032

rat: replaced 0.002389014203700413 by 1071627/448564516 = 0.00238901420370041

rat: replaced 0.00372838808577948 by 661903/177530607 = 0.003728388085779485

rat: replaced 0.005362386673832029 by 5230891/975478144 = 0.005362386673832028

rat: replaced 0.007289846577694006 by 32241346/4422774287 = 0.007289846577694006

rat: replaced 0.009509575061314376 by 2146493/225719129 = 0.009509575061314364

rat: replaced 0.01202035016202822 by 1789188/148846579 = 0.01202035016202825
```

rat: replaced 0.01482092081275069 by 2581665/174190594 = 0.01482092081275066  
rat: replaced 0.01791000696708436 by 5107285/285163764 = 0.01791000696708436  
rat: replaced 0.02128629972732062 by 3323295/156123659 = 0.02128629972732064  
rat: replaced 0.02494846147533059 by 4548287/182307314 = 0.02494846147533061  
rat: replaced 0.02889512600632479 by 3147802/108938857 = 0.02889512600632481  
rat: replaced 0.0331248986654725 by 5858625/176864692 = 0.03312489866547248  
rat: replaced 0.03763635648736519 by 10043830/266865099 = 0.03763635648736518  
rat: replaced 0.04242804833831373 by 4635713/109260576 = 0.04242804833831372  
rat: replaced 0.04749849506145984 by 5610259/118114458 = 0.04749849506145979  
rat: replaced 0.05284618962468965 by 4237503/80185592 = 0.05284618962468968  
rat: replaced 0.05846959727133633 by 3317197/56733707 = 0.05846959727133642  
rat: replaced 0.06436715567365475 by 13427433/208606903 = 0.06436715567365477  
rat: replaced 0.07053727508905278 by 8025659/113778977 = 0.07053727508905269  
rat: replaced 0.07697833851906408 by 6306881/81930594 = 0.07697833851906408  
rat: replaced 0.08368870187104593 by 4282086/51166835 = 0.08368870187104596  
rat: replaced 0.09066669412258874 by 2175091/23989967 = 0.09066669412258883  
rat: replaced 0.09791061748861546 by 8290049/84669561 = 0.09791061748861554

```
rat: replaced 0.1054187475911595 by 8501563/80645646 = 0.1054187475911595
rat: replaced 0.1131893336318011 by 6539019/57770629 = 0.113189333631801
rat: replaced 0.121220598566744 by 5779101/47674249 = 0.1212205985667441
rat: replaced 0.1295107392845216 by 7134865/55090914 = 0.1295107392845216
rat: replaced 0.1380579267863034 by 6113057/44278928 = 0.1380579267863034
rat: replaced 0.1468603063687953 by 6311140/42973763 = 0.1468603063687953
rat: replaced 0.1559159978097077 by 4027079/25828517 = 0.1559159978097078
rat: replaced 0.1652230955557758 by 10597125/64138279 = 0.1652230955557757
rat: replaced 0.1747796689133147 by 9649007/55206690 = 0.1747796689133147
rat: replaced 0.1845837622412855 by 6871913/37229239 = 0.1845837622412857
rat: replaced 0.1946333951468589 by 39341769/202132676 = 0.1946333951468589
rat: replaced 0.2049265626834523 by 10758647/52500012 = 0.2049265626834523
rat: replaced 0.2154612355512225 by 33702610/156420759 = 0.2154612355512225
rat: replaced 0.2262353602999955 by 2338161/10335082 = 0.2262353602999957
rat: replaced 0.2372468595346078 by 7573078/31920667 = 0.2372468595346081
rat: replaced 0.2484936321226457 by 3764353/15148690 = 0.2484936321226456
rat: replaced 0.2599735534045555 by 26335713/101301508 = 0.2599735534045554
rat: replaced 0.2716844754061095 by 29831699/109802737 = 0.2716844754061094
```

rat: replaced 0.2836242270531998 by 15100773/53242183 = 0.2836242270531995  
rat: replaced 0.2957906143889442 by 2942977/9949528 = 0.2957906143889439  
rat: replaced 0.3081814207930817 by 12077608/39189929 = 0.3081814207930818  
rat: replaced 0.3207944072036307 by 9185023/28632117 = 0.3207944072036308  
rat: replaced 0.333627312340794 by 5228336/15671187 = 0.3336273123407946  
rat: replaced 0.346677852933085 by 15615111/45042136 = 0.3466778529330847  
rat: replaced 0.3599437239456539 by 7564465/21015688 = 0.3599437239456543  
rat: replaced 0.3734225988107874 by 7702871/20627758 = 0.3734225988107869  
rat: replaced 0.3871121296605642 by 97723109/252441351 = 0.3871121296605642  
rat: replaced 0.4010099475616409 by 3146543/7846546 = 0.4010099475616405  
rat: replaced 0.4151136627521425 by 6219049/14981557 = 0.4151136627521425  
rat: replaced 0.4294208648806354 by 26148647/60892819 = 0.4294208648806356  
rat: replaced 0.4439291232471635 by 19525684/43983787 = 0.4439291232471638  
rat: replaced 0.458635987046313 by 38604672/84172793 = 0.4586359870463132  
rat: replaced 0.4735389856122937 by 11146199/23538081 = 0.4735389856122935  
rat: replaced 0.488635628666001 by 13946471/28541658 = 0.4886356286660011  
rat: replaced 0.5039234065640431 by 5948069/11803518 = 0.503923406564043

rat: replaced 0.5193997905497036 by 24027011/46259185 = 0.5193997905497038  
rat: replaced 0.5350622330058146 by 7363779/13762472 = 0.5350622330058147  
rat: replaced 0.5509081677095147 by 8130825/14758948 = 0.5509081677095142  
rat: replaced 0.5669350100888726 by 10250363/18080314 = 0.5669350100888735  
rat: replaced 0.5831401574813392 by 37655026/64572857 = 0.5831401574813393  
rat: replaced 0.5995209893940125 by 30778651/51338738 = 0.5995209893940128  
rat: replaced 0.6160748677656853 by 23698401/38466755 = 0.616074867765685  
rat: replaced 0.6327991372306488 by 5052598/7984521 = 0.6327991372306492  
rat: replaced 0.6496911253842265 by 60646047/93345968 = 0.6496911253842266  
rat: replaced 0.666748143050013 by 30125566/45182827 = 0.6667481430500132  
rat: replaced 0.6839674845487889 by 8953739/13090884 = 0.6839674845487899  
rat: replaced 0.7013464279690875 by 7888577/11247761 = 0.7013464279690864  
rat: replaced 0.7188822354393821 by 16662338/23178119 = 0.7188822354393815  
rat: replaced 0.7365721534018723 by 13899283/18870226 = 0.7365721534018723  
rat: replaced 0.7544134128878366 by 16270763/21567436 = 0.754413412887837  
rat: replaced 0.7724032297945274 by 8203205/10620366 = 0.7724032297945287  
rat: replaced 0.7905388051635788 by 10794522/13654639 = 0.7905388051635784  
rat: replaced 0.8088173254609005 by 16745047/20703126 = 0.808817325460899

```
rat: replaced 0.8272359628580275 by 20291194/24528907 = 0.827235962858027
rat: replaced 0.8457918755149025 by 10996366/13001267 = 0.8457918755149018
rat: replaced 0.8644822078640563 by 9158500/10594203 = 0.8644822078640555
rat: replaced 0.8833040908961625 by 13759446/15577247 = 0.8833040908961641
rat: replaced 0.9022546424469358 by 19827819/21975857 = 0.9022546424469362
rat: replaced 0.9213309674853474 by 60458149/65620446 = 0.9213309674853475
rat: replaced 0.9405301584031224 by 11658841/12396031 = 0.9405301584031212
rat: replaced 0.9598492953055026 by 26214088/27310629 = 0.9598492953055018
rat: replaced 0.9792854463032298 by 35089005/35831233 = 0.9792854463032293
rat: replaced 0.9988356678057343 by 15735752/15754095 = 0.9988356678057356
rat: replaced 1.018497004815491 by 16202286/15908035 = 1.018497004815491
rat: replaced 1.038266491223517 by 17763365/17108676 = 1.038266491223517
rat: replaced 1.058141150105979 by 33730321/31876958 = 1.058141150105979
rat: replaced 1.078117994021884 by 51996446/48228901 = 1.078117994021883
rat: replaced 1.098194025311821 by 124719922/113568203 = 1.098194025311821
rat: replaced 1.118366236397724 by 92837336/83011569 = 1.118366236397724
rat: replaced 1.13863161008363 by 20601995/18093644 = 1.138631610083629
```

```
rat: replaced 1.158987119857388 by 20626233/17796775 = 1.15898711985739  
rat: replaced 1.17942973019332 by 4098089/3474636 = 1.179429730193321  
rat: replaced 1.199956396855759 by 17442145/14535649 = 1.199956396855758  
part: invalid index of list or matrix.  
#0: lineIntersection(g=[3,1,[3,1] . Q],h=[-1,3,-2])  
#1: projectToLine(a=Q,g=[-1,3,-2])  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
... ance(Q,projectToLine(Q,lineThrough(A,B)))|ratsimp; $r ...
```

```
>LD &= circleWithCenter(Q,r); // Lingkaran dalam
```

Mari kita tambahkan ini ke plot.

```
>color(5); plotCircle(LD()):
```

```
Q is not a variable!  
Error in expression: [Q[1],Q[2],cc[3]]  
Error in:  
color(5); plotCircle(LD()): ...
```

Selanjutnya akan dicari persamaan tempat kedudukan titik-titik yang berjarak sama ke titik C dan ke garis AB.

```
>p &= getHesseForm(lineThrough(A,B),x,y,C)-distance([x,y],C); $p='0
```

Persamaan tersebut dapat digambar menjadi satu dengan gambar sebelumnya.

```
>plot2d(p,level=0,add=1,contourcolor=6):
```

Wrong argument!

Cannot combine a symbolic expression here.

Did you want to create a symbolic expression?

Then start with &.

```
Error in expression: [-sqrt(5),-sqrt((2-4.999958333473664e-5*r)^2+(1-1.66665833335744e-7*r)^2),-sq
%ploteval2:
    if maps then return %mapexpression2(x,y,f$;args());
fcontour:
    Z=%ploteval2(f$,X,Y,maps;args());
Try "trace errors" to inspect local variables after errors.
plot2d:
    =style,=outline,=frame);
```

Ini seharusnya menjadi beberapa fungsi, tetapi pemecah default Maxima hanya dapat menemukan solusinya, jika kita kuadratkan persamaannya. Akibatnya, kami mendapatkan solusi palsu.

```
>akar &= solve(getHesseForm(lineThrough(A,B),x,y,C)^2-distance([x,y],C)^2,y)
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
... (lineThrough(A,B),x,y,C)^2-distance([x,y],C)^2,y) ...
```

Solusi pertama adalah

```
maxima: akar[1]
```

Menambahkan solusi pertama ke plot menunjukkan, bahwa itu memang jalan yang kita cari. Teorinya memberi tahu kita bahwa itu adalah parabola yang diputar.

```
>plot2d(&rhs(akar[1]),add=1);  
>function g(x) &= rhs(akar[1]); $'g(x)= g(x)// fungsi yang mendefinisikan kurva di atas  
>T &=[-1, g(-1)]; // ambil sebarang titik pada kurva tersebut  
>dTC &= distance(T,C); $fullratsimp(dTC), $float(%) // jarak T ke C  
>U &= projectToLine(T,lineThrough(A,B)); $U // proyeksi T pada garis AB
```

```
Maxima said:  
rat: replaced 5.049958083474387e-5 by 102157/2022927682 = 5.049958083474385e-5
```

rat: replaced 2.039932534230044e-4 by 284619/1395237319 = 2.039932534230043e-4  
rat: replaced 4.634656435254722e-4 by 573493/1237401322 = 4.634656435254721e-4  
rat: replaced 8.31890779119604e-4 by 332331/399488741 = 8.318907791196046e-4  
rat: replaced 0.001312231792998733 by 448125/341498356 = 0.001312231792998734  
rat: replaced 0.001907440626462018 by 276030/144712237 = 0.001907440626462018  
rat: replaced 0.002620457734122131 by 2586613/987084419 = 0.002620457734122131  
rat: replaced 0.00345421178986248 by 3402379/984994322 = 0.00345421178986248  
rat: replaced 0.004411619393972596 by 966955/219183686 = 0.004411619393972597  
rat: replaced 0.005495584781489732 by 2798484/509224061 = 0.005495584781489734  
rat: replaced 0.006708999531778753 by 6054060/902378957 = 0.006708999531778753  
rat: replaced 0.008054742279375651 by 806546/100133061 = 0.00805474227937564  
rat: replaced 0.009535678426127348 by 4115324/431571181 = 0.009535678426127346  
rat: replaced 0.01115465985465333 by 2266398/203179481 = 0.01115465985465334  
rat: replaced 0.01291452464316009 by 2106925/163143829 = 0.01291452464316012  
rat: replaced 0.01481809678163515 by 2779203/187554653 = 0.01481809678163516  
rat: replaced 0.01686818588945119 by 7427428/440321683 = 0.01686818588945119  
rat: replaced 0.01906758693440599 by 2278085/119474216 = 0.019067586934406  
rat: replaced 0.02141907995322798 by 2316386/108145915 = 0.02141907995322801

rat: replaced 0.02392542977357476 by 1665518/69612877 = 0.02392542977357479  
rat: replaced 0.02658938573755304 by 3678645/138350131 = 0.02658938573755308  
rat: replaced 0.02941368142678652 by 4053557/137811957 = 0.0294136814267865  
rat: replaced 0.03240103438906003 by 2629160/81144323 = 0.03240103438906009  
rat: replaced 0.03555414586656669 by 1834427/51595305 = 0.03555414586656674  
rat: replaced 0.03887570052578646 by 1643964/42287701 = 0.03887570052578645  
rat: replaced 0.04236836618902146 by 3055464/72116635 = 0.04236836618902143  
rat: replaced 0.04603479356761608 by 3139251/68193007 = 0.0460347935676161  
rat: replaced 0.04987761599688789 by 5203437/104324092 = 0.04987761599688785  
rat: replaced 0.05389944917279615 by 4533622/84112585 = 0.0538994491727962  
rat: replaced 0.05810289089037535 by 11687290/201148167 = 0.05810289089037535  
rat: replaced 0.06249052078395612 by 3949243/63197473 = 0.06249052078395603  
rat: replaced 0.0670649000692059 by 3281728/48933615 = 0.067064900069206  
rat: replaced 0.07182857128700804 by 4146139/57722699 = 0.07182857128700791  
rat: replaced 0.07678405804921068 by 1198255/15605518 = 0.07678405804921054  
rat: replaced 0.08193386478626702 by 5956639/72700574 = 0.08193386478626702  
rat: replaced 0.08728047649679532 by 2799808/32078285 = 0.08728047649679527

rat: replaced 0.09282635849907966 by  $10292829/110882611 = 0.09282635849907972$   
rat: replaced 0.09857395618454184 by  $4198057/42587892 = 0.09857395618454184$   
rat: replaced 0.1045256947732028 by  $30563827/292404916 = 0.1045256947732028$   
rat: replaced 0.1106839790711635 by  $8949559/80856860 = 0.1106839790711635$   
rat: replaced 0.1170511932301264 by  $9911603/84677505 = 0.1170511932301265$   
rat: replaced 0.1236297005089814 by  $19561703/158228184 = 0.1236297005089814$   
rat: replaced 0.1304218430374826 by  $4975231/38147222 = 0.1304218430374825$   
rat: replaced 0.137429941582038 by  $3502939/25488907 = 0.137429941582038$   
rat: replaced 0.1446562953136327 by  $15521432/107298697 = 0.1446562953136327$   
rat: replaced 0.1521031815779155 by  $18080502/118869979 = 0.1521031815779155$   
rat: replaced 0.1597728556674664 by  $37419026/234201397 = 0.1597728556674664$   
rat: replaced 0.1676675505962674 by  $22585897/134706429 = 0.1676675505962674$   
rat: replaced 0.1757894768764047 by  $15940893/90681725 = 0.1757894768764048$   
rat: replaced 0.1841408222970185 by  $7944795/43145213 = 0.1841408222970182$   
rat: replaced 0.1927237517055264 by  $1392861/7227241 = 0.1927237517055264$   
rat: replaced 0.2015404067911402 by  $1735485/8611102 = 0.2015404067911401$   
rat: replaced 0.2105929058706983 by  $10627754/50465869 = 0.2105929058706985$   
rat: replaced 0.2198833436768368 by  $9372347/42624179 = 0.2198833436768366$

```
rat: replaced 0.2294137911485169 by 7405273/32279110 = 0.2294137911485168
rat: replaced 0.239186295223934 by 27692337/115777273 = 0.239186295223934
rat: replaced 0.2492028786358237 by 8925310/35815437 = 0.249202878635824
rat: replaced 0.2594655397091927 by 11150701/42975653 = 0.259465539709193
rat: replaced 0.2699762521614856 by 11249087/41666950 = 0.2699762521614853
rat: replaced 0.280736964905216 by 12097010/43090193 = 0.2807369649052164
rat: replaced 0.2917496018530771 by 14831788/50837389 = 0.2917496018530771
rat: replaced 0.3030160617255513 by 18597622/61375037 = 0.3030160617255514
rat: replaced 0.3145382178610399 by 11102944/35299189 = 0.3145382178610392
rat: replaced 0.3263179180285316 by 13053510/40002431 = 0.3263179180285318
rat: replaced 0.3383569842428258 by 13796661/40775458 = 0.3383569842428257
rat: replaced 0.3506572125823338 by 33496033/95523582 = 0.3506572125823338
rat: replaced 0.3632203730094723 by 23086207/63559780 = 0.3632203730094724
rat: replaced 0.3760482091936667 by 22674222/60296051 = 0.3760482091936668
rat: replaced 0.3891424383369902 by 33246815/85436107 = 0.3891424383369902
rat: replaced 0.402504751002439 by 7793813/19363282 = 0.4025047510024385
rat: replaced 0.4161368109448825 by 10481453/25187517 = 0.4161368109448819
```

rat: replaced 0.4300402549446862 by 19565443/45496771 = 0.4300402549446861  
rat: replaced 0.4442166926440365 by 16102633/36249500 = 0.4442166926440365  
rat: replaced 0.4586677063859775 by 19404529/42306290 = 0.4586677063859771  
rat: replaced 0.4733948510561774 by 10262860/21679281 = 0.4733948510561766  
rat: replaced 0.4883996539274416 by 4159841/8517289 = 0.488399653927441  
rat: replaced 0.5036836145069872 by 13202363/26211619 = 0.5036836145069864  
rat: replaced 0.5192482043864929 by 12221370/23536663 = 0.5192482043864927  
rat: replaced 0.5350948670949413 by 52965833/98984005 = 0.5350948670949413  
rat: replaced 0.551225017954267 by 14288533/25921416 = 0.551225017954266  
rat: replaced 0.5676400439378262 by 25565995/45039097 = 0.5676400439378259  
rat: replaced 0.5843413035316997 by 18888222/32323955 = 0.5843413035316997  
rat: replaced 0.6013301265988455 by 5789399/9627655 = 0.6013301265988447  
rat: replaced 0.6186078142461149 by 4803773/7765458 = 0.618607814246114  
rat: replaced 0.6361756386941407 by 13914515/21872128 = 0.6361756386941407  
rat: replaced 0.6540348431501183 by 48160581/73636109 = 0.6540348431501181  
rat: replaced 0.6721866416834846 by 6617334/9844489 = 0.6721866416834841  
rat: replaced 0.690632219104513 by 16840135/24383651 = 0.6906322191045139  
rat: replaced 0.7093727308458327 by 29189494/41148317 = 0.7093727308458326

```
rat: replaced 0.7284093028468864 by 13153959/18058472 = 0.7284093028468854
rat: replaced 0.7477430314413382 by 12236470/16364539 = 0.7477430314413379
rat: replaced 0.7673749832474404 by 39576757/51574208 = 0.7673749832474402
rat: replaced 0.7873061950613714 by 6818881/8661028 = 0.7873061950613714
rat: replaced 0.8075376737535601 by 20498953/25384516 = 0.807537673753559
rat: replaced 0.8280703961679966 by 6989671/8440914 = 0.8280703961679979
rat: replaced 0.84890530902455 by 25231431/29722315 = 0.8489053090245494
rat: replaced 0.8700433288242969 by 9721738/11173855 = 0.8700433288242957
rat: replaced 0.8914853417578728 by 33469619/37543656 = 0.8914853417578725
rat: replaced 0.9132322036168524 by 21961040/24047597 = 0.913232203616852
rat: replaced 1.498320841708742e-4 by 1329822/8875415485 = 1.498320841708742e-4
rat: replaced 5.986466935998108e-4 by 398723/666040595 = 5.986466935998098e-4
rat: replaced 0.001345398955533032 by 4525441/3363642421 = 0.001345398955533032
rat: replaced 0.002389014203700413 by 1071627/448564516 = 0.00238901420370041
rat: replaced 0.00372838808577948 by 661903/177530607 = 0.003728388085779485
rat: replaced 0.005362386673832029 by 5230891/975478144 = 0.005362386673832028
rat: replaced 0.007289846577694006 by 32241346/4422774287 = 0.007289846577694006
```

rat: replaced 0.009509575061314376 by  $2146493/225719129 = 0.009509575061314364$   
rat: replaced 0.01202035016202822 by  $1789188/148846579 = 0.01202035016202825$   
rat: replaced 0.01482092081275069 by  $2581665/174190594 = 0.01482092081275066$   
rat: replaced 0.01791000696708436 by  $5107285/285163764 = 0.01791000696708436$   
rat: replaced 0.02128629972732062 by  $3323295/156123659 = 0.02128629972732064$   
rat: replaced 0.02494846147533059 by  $4548287/182307314 = 0.02494846147533061$   
rat: replaced 0.02889512600632479 by  $3147802/108938857 = 0.02889512600632481$   
rat: replaced 0.0331248986654725 by  $5858625/176864692 = 0.03312489866547248$   
rat: replaced 0.03763635648736519 by  $10043830/266865099 = 0.03763635648736518$   
rat: replaced 0.04242804833831373 by  $4635713/109260576 = 0.04242804833831372$   
rat: replaced 0.04749849506145984 by  $5610259/118114458 = 0.04749849506145979$   
rat: replaced 0.05284618962468965 by  $4237503/80185592 = 0.05284618962468968$   
rat: replaced 0.05846959727133633 by  $3317197/56733707 = 0.05846959727133642$   
rat: replaced 0.06436715567365475 by  $13427433/208606903 = 0.06436715567365477$   
rat: replaced 0.07053727508905278 by  $8025659/113778977 = 0.07053727508905269$   
rat: replaced 0.07697833851906408 by  $6306881/81930594 = 0.07697833851906408$   
rat: replaced 0.08368870187104593 by  $4282086/51166835 = 0.08368870187104596$   
rat: replaced 0.09066669412258874 by  $2175091/23989967 = 0.09066669412258883$

```
rat: replaced 0.09791061748861546 by 8290049/84669561 = 0.09791061748861554
rat: replaced 0.1054187475911595 by 8501563/80645646 = 0.1054187475911595
rat: replaced 0.1131893336318011 by 6539019/57770629 = 0.113189333631801
rat: replaced 0.121220598566744 by 5779101/47674249 = 0.1212205985667441
rat: replaced 0.1295107392845216 by 7134865/55090914 = 0.1295107392845216
rat: replaced 0.1380579267863034 by 6113057/44278928 = 0.1380579267863034
rat: replaced 0.1468603063687953 by 6311140/42973763 = 0.1468603063687953
rat: replaced 0.1559159978097077 by 4027079/25828517 = 0.1559159978097078
rat: replaced 0.1652230955557758 by 10597125/64138279 = 0.1652230955557757
rat: replaced 0.1747796689133147 by 9649007/55206690 = 0.1747796689133147
rat: replaced 0.1845837622412855 by 6871913/37229239 = 0.1845837622412857
rat: replaced 0.1946333951468589 by 39341769/202132676 = 0.1946333951468589
rat: replaced 0.2049265626834523 by 10758647/52500012 = 0.2049265626834523
rat: replaced 0.2154612355512225 by 33702610/156420759 = 0.2154612355512225
rat: replaced 0.2262353602999955 by 2338161/10335082 = 0.2262353602999957
rat: replaced 0.2372468595346078 by 7573078/31920667 = 0.2372468595346081
rat: replaced 0.2484936321226457 by 3764353/15148690 = 0.2484936321226456
```

rat: replaced 0.2599735534045555 by 26335713/101301508 = 0.2599735534045554  
rat: replaced 0.2716844754061095 by 29831699/109802737 = 0.2716844754061094  
rat: replaced 0.2836242270531998 by 15100773/53242183 = 0.2836242270531995  
rat: replaced 0.2957906143889442 by 2942977/9949528 = 0.2957906143889439  
rat: replaced 0.3081814207930817 by 12077608/39189929 = 0.3081814207930818  
rat: replaced 0.3207944072036307 by 9185023/28632117 = 0.3207944072036308  
rat: replaced 0.333627312340794 by 5228336/15671187 = 0.3336273123407946  
rat: replaced 0.346677852933085 by 15615111/45042136 = 0.3466778529330847  
rat: replaced 0.3599437239456539 by 7564465/21015688 = 0.3599437239456543  
rat: replaced 0.3734225988107874 by 7702871/20627758 = 0.3734225988107869  
rat: replaced 0.3871121296605642 by 97723109/252441351 = 0.3871121296605642  
rat: replaced 0.4010099475616409 by 3146543/7846546 = 0.4010099475616405  
rat: replaced 0.4151136627521425 by 6219049/14981557 = 0.4151136627521425  
rat: replaced 0.4294208648806354 by 26148647/60892819 = 0.4294208648806356  
rat: replaced 0.4439291232471635 by 19525684/43983787 = 0.4439291232471638  
rat: replaced 0.458635987046313 by 38604672/84172793 = 0.4586359870463132  
rat: replaced 0.4735389856122937 by 11146199/23538081 = 0.4735389856122935  
rat: replaced 0.488635628666001 by 13946471/28541658 = 0.4886356286660011

rat: replaced 0.5039234065640431 by 5948069/11803518 = 0.503923406564043

rat: replaced 0.5193997905497036 by 24027011/46259185 = 0.5193997905497038

rat: replaced 0.5350622330058146 by 7363779/13762472 = 0.5350622330058147

rat: replaced 0.5509081677095147 by 8130825/14758948 = 0.5509081677095142

rat: replaced 0.5669350100888726 by 10250363/18080314 = 0.5669350100888735

rat: replaced 0.5831401574813392 by 37655026/64572857 = 0.5831401574813393

rat: replaced 0.5995209893940125 by 30778651/51338738 = 0.5995209893940128

rat: replaced 0.6160748677656853 by 23698401/38466755 = 0.616074867765685

rat: replaced 0.6327991372306488 by 5052598/7984521 = 0.6327991372306492

rat: replaced 0.6496911253842265 by 60646047/93345968 = 0.6496911253842266

rat: replaced 0.666748143050013 by 30125566/45182827 = 0.6667481430500132

rat: replaced 0.6839674845487889 by 8953739/13090884 = 0.6839674845487899

rat: replaced 0.7013464279690875 by 7888577/11247761 = 0.7013464279690864

rat: replaced 0.7188822354393821 by 16662338/23178119 = 0.7188822354393815

rat: replaced 0.7365721534018723 by 13899283/18870226 = 0.7365721534018723

rat: replaced 0.7544134128878366 by 16270763/21567436 = 0.754413412887837

rat: replaced 0.7724032297945274 by 8203205/10620366 = 0.7724032297945287

rat: replaced 0.7905388051635788 by  $10794522/13654639 = 0.7905388051635784$   
rat: replaced 0.8088173254609005 by  $16745047/20703126 = 0.808817325460899$   
rat: replaced 0.8272359628580275 by  $20291194/24528907 = 0.827235962858027$   
rat: replaced 0.8457918755149025 by  $10996366/13001267 = 0.8457918755149018$   
rat: replaced 0.8644822078640563 by  $9158500/10594203 = 0.8644822078640555$   
rat: replaced 0.8833040908961625 by  $13759446/15577247 = 0.8833040908961641$   
rat: replaced 0.9022546424469358 by  $19827819/21975857 = 0.9022546424469362$   
rat: replaced 0.9213309674853474 by  $60458149/65620446 = 0.9213309674853475$   
rat: replaced 0.9405301584031224 by  $11658841/12396031 = 0.9405301584031212$   
rat: replaced 0.9598492953055026 by  $26214088/27310629 = 0.9598492953055018$   
rat: replaced 0.9792854463032298 by  $35089005/35831233 = 0.9792854463032293$   
rat: replaced 0.9988356678057343 by  $15735752/15754095 = 0.9988356678057356$   
rat: replaced 1.018497004815491 by  $16202286/15908035 = 1.018497004815491$   
rat: replaced 1.038266491223517 by  $17763365/17108676 = 1.038266491223517$   
rat: replaced 1.058141150105979 by  $33730321/31876958 = 1.058141150105979$   
rat: replaced 1.078117994021884 by  $51996446/48228901 = 1.078117994021883$   
rat: replaced 1.098194025311821 by  $124719922/113568203 = 1.098194025311821$   
rat: replaced 1.118366236397724 by  $92837336/83011569 = 1.118366236397724$

```

rat: replaced 1.13863161008363 by 20601995/18093644 = 1.138631610083629

rat: replaced 1.158987119857388 by 20626233/17796775 = 1.15898711985739

rat: replaced 1.17942973019332 by 4098089/3474636 = 1.179429730193321

rat: replaced 1.199956396855759 by 17442145/14535649 = 1.199956396855758
part: invalid index of list or matrix.
#0: lineIntersection(g=[3,1,-3],h=[-1,3,-2])
#1: projectToLine(a=[-1,0],g=[-1,3,-2])
-- an error. To debug this try: debugmode(true);

Error in:
U &= projectToLine(T,lineThrough(A,B)); $U // proyeksi T pada ...

```

```
>dU2AB &= distance(T,U); $fullratsimp(dU2AB), $float(%) // jatakan T ke AB
```

Ternyata jarak T ke C sama dengan jarak T ke AB. Coba Anda pilih titik T yang lain dan ulangi perhitungan-perhitungan di atas untuk menunjukkan bahwa hasilnya juga sama.

---

### Contoh 5: Trigonometri Rasional

Ini terinspirasi dari ceramah N.J.Wildberger. Dalam bukunya "Divine Proportions", Wildberger men-gusulkan untuk mengganti pengertian klasik tentang jarak dan sudut dengan kuadrat dan penyebaran. Dengan menggunakan ini, memang mungkin untuk menghindari fungsi trigonometri dalam banyak contoh, dan tetap "rasional".

Berikut ini, saya memperkenalkan konsep, dan memecahkan beberapa masalah. Saya menggunakan perhitungan simbolik Maxima di sini, yang menyembunyikan keuntungan utama dari trigonometri rasional bahwa perhitungan hanya dapat dilakukan dengan kertas dan pensil. Anda diundang untuk memeriksa hasil tanpa komputer.

Intinya adalah bahwa perhitungan rasional simbolis sering kali menghasilkan hasil yang sederhana. Sebaliknya, trigonometri klasik menghasilkan hasil trigonometri yang rumit, yang hanya mengevaluasi perkiraan numerik.

```
>load geometry;
```

Untuk pengenalan pertama, kami menggunakan segitiga persegi panjang dengan proporsi Mesir terkenal 3, 4 dan 5. Perintah berikut adalah perintah Euler untuk merencanakan geometri bidang yang terdapat dalam file Euler "geometry.e".

```
>C&:=[0,0]; A&:=[4,0]; B&:=[0,3]; ...
>setPlotRange(-1,5,-1,5); ...
>plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C"); ...
>plotSegment(B,A,"c"); plotSegment(A,C,"b"); plotSegment(C,B,"a"); ...
>insimg(30);
```

Tentu saja,

$$\sin(w_a) = \frac{a}{c},$$

di mana  $w_a$  adalah sudut di A. Cara yang biasa untuk menghitung sudut ini, adalah dengan mengambil invers dari fungsi sinus. Hasilnya adalah sudut yang tidak dapat dicerna, yang hanya dapat dicetak kira-kira.

```
>wa := arcsin(3/5); degprint(wa)
```

36°52'11.63''

Trigonometri rasional mencoba menghindari hal ini.

Gagasan pertama trigonometri rasional adalah kuadran, yang menggantikan jarak. Sebenarnya, itu hanya jarak kuadrat. Berikut ini, a, b, dan c menunjukkan kuadrat dari sisi-sisinya.

Teorema Pythagoras menjadi  $a+b=c$ .

```
>a &= 3^2; b &= 4^2; c &= 5^2; &a+b=c
```

25 = 25

Pengertian kedua dari trigonometri rasional adalah penyebaran. Spread mengukur pembukaan antar baris. Ini adalah 0, jika garis-garisnya sejajar, dan 1, jika garis-garisnya persegi panjang. Ini adalah kuadrat sinus sudut antara dua garis.

Penyebaran garis AB dan AC pada gambar di atas didefinisikan sebagai:

$$s_a = \sin(\alpha)^2 = \frac{a}{c},$$

di mana a dan c adalah kuadrat dari sembarang segitiga siku-siku dengan salah satu sudut di A.

```
>sa &= a/c; $sa
```

Ini lebih mudah dihitung daripada sudut, tentu saja. Tetapi Anda kehilangan properti bahwa sudut dapat ditambahkan dengan mudah.

Tentu saja, kita dapat mengonversi nilai perkiraan untuk sudut wa menjadi sprad, dan mencetaknya sebagai pecahan.

```
>fracprint(sin(wa)^2)
```

9/25

Hukum kosinus trigonometri klasik diterjemahkan menjadi "hukum silang" berikut.

$$(c + b - a)^2 = 4bc(1 - s_a)$$

Di sini  $a$ ,  $b$ , dan  $c$  adalah kuadran dari sisi-sisi segitiga, dan  $sa$  adalah penyebaran sudut  $A$ . Sisi  $a$ , seperti biasa, berhadapan dengan sudut  $A$ .

Hukum ini diimplementasikan dalam file geometri.e yang kami muat ke Euler.

```
>$crosslaw(aa,bb,cc,saa)
```

Dalam kasus kami, kami mendapatkan

```
>$crosslaw(a,b,c,sa)
```

Mari kita gunakan crosslaw ini untuk mencari spread di  $A$ . Untuk melakukan ini, kita buat crosslaw untuk kuadran  $a$ ,  $b$ , dan  $c$ , dan selesaikan untuk spread yang tidak diketahui  $sa$ .

Anda dapat melakukannya dengan tangan dengan mudah, tetapi saya menggunakan Maxima. Tentu saja, kami mendapatkan hasilnya, kami sudah memilikinya.

```
>$crosslaw(a,b,c,x), $solve(%,x)
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$crosslaw(a,b,c,x), $solve(%,x) ...
```

Kita sudah tahu ini. Definisi spread adalah kasus khusus dari crosslaw.

Kita juga dapat menyelesaikan ini untuk umum a,b,c. Hasilnya adalah rumus yang menghitung penyebaran sudut segitiga yang diberikan kuadrat dari ketiga sisinya.

```
>$solve(crosslaw(aa,bb,cc,x),x)
```

Maxima said:

```
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);
```

Error in:

```
$solve(crosslaw(aa,bb,cc,x),x) ...  
^
```

Kita bisa membuat fungsi dari hasilnya. Fungsi seperti itu sudah didefinisikan dalam file geometri.e dari Euler.

```
>$spread(a,b,c)
```

Sebagai contoh, kita dapat menggunakan untuk menghitung sudut segitiga dengan sisi

$$a, \quad a, \quad \frac{4a}{7}$$

Hasilnya rasional, yang tidak begitu mudah didapat jika kita menggunakan trigonometri klasik.

```
>$spread(a,a,4*a/7)
```

Ini adalah sudut dalam derajat.

```
>degprint(arcsin(sqrt(6/7)))
```

$67^\circ 47' 32.44''$

---

## Contoh lain

Sekarang, mari kita coba contoh yang lebih maju.

Kami mengatur tiga sudut segitiga sebagai berikut.

```
>A&:=[1,2]; B&:=[4,3]; C&:=[0,4]; ...
>setPlotRange(-1,5,1,7); ...
>plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C"); ...
>plotSegment(B,A,"c"); plotSegment(A,C,"b"); plotSegment(C,B,"a"); ...
>insimg;
```

Menggunakan Pythagoras, mudah untuk menghitung jarak antara dua titik. Saya pertama kali menggunakan jarak fungsi file Euler untuk geometri. Jarak fungsi menggunakan geometri klasik.

```
>$distance(A,B)
```

Euler juga mengandung fungsi untuk kuadran antara dua titik.

Dalam contoh berikut, karena  $c+b$  bukan  $a$ , maka segitiga itu bukan persegi panjang.

```
>c &= quad(A,B); $c, b &= quad(A,C); $b, a &= quad(B,C); $a,
```

Pertama, mari kita hitung sudut tradisional. Fungsi computeAngle menggunakan metode biasa berdasarkan hasil kali titik dua vektor. Hasilnya adalah beberapa pendekatan floating point.

$$A = \langle 1, 2 \rangle \quad B = \langle 4, 3 \rangle, \quad C = \langle 0, 4 \rangle$$

$$\mathbf{a} = C - B = \langle -4, 1 \rangle, \quad \mathbf{c} = A - B = \langle -3, -1 \rangle, \quad \beta = \angle ABC$$

$$\mathbf{a} \cdot \mathbf{c} = |\mathbf{a}| \cdot |\mathbf{c}| \cos \beta$$

$$\cos \angle ABC = \cos \beta = \frac{\mathbf{a} \cdot \mathbf{c}}{|\mathbf{a}| \cdot |\mathbf{c}|} = \frac{12 - 1}{\sqrt{17} \sqrt{10}} = \frac{11}{\sqrt{17} \sqrt{10}}$$

```
>wb &= computeAngle(A,B,C); $wb, $(wb/pi*180)()
```

32.4711922908

Dengan menggunakan pensil dan kertas, kita dapat melakukan hal yang sama dengan hukum silang. Kami memasukkan kuadran a, b, dan c ke dalam hukum silang dan menyelesaikan x.

```
>$crosslaw(a,b,c,x), $solve(% ,x) , // (b+c-a)^=4b.c(1-x)
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$crosslaw(a,b,c,x), $solve(% ,x) , // (b+c-a)^=4b.c(1-x) ...
```

Yaitu, apa yang dilakukan oleh penyebaran fungsi yang didefinisikan dalam "geometry.e".

```
>sb &= spread(b,a,c); $sb
```

Maxima mendapatkan hasil yang sama menggunakan trigonometri biasa, jika kita memaksanya. Itu menyelesaikan istilah  $\sin(\arccos(...))$  menjadi hasil pecahan. Sebagian besar siswa tidak dapat melakukan ini.

```
>$sin(computeAngle(A,B,C))^2
```

Setelah kita memiliki spread di B, kita dapat menghitung tinggi ha di sisi a. Ingat bahwa

$$s_b = \frac{h_a}{c}$$

Menurut definisi.

```
>ha &= c*sb; $ha
```

Gambar berikut telah dihasilkan dengan program geometri C.a.R., yang dapat menggambar kuadrat dan menyebar.

image: (20) Rational\_Geometry\_CaR.png

Menurut definisi, panjang ha adalah akar kuadrat dari kuadratnya.

```
>$sqrt(ha)
```

Sekarang kita dapat menghitung luas segitiga. Jangan lupa, bahwa kita berhadapan dengan kuadrat!

```
>$sqrt(ha)*sqrt(a)/2
```

Rumus determinan biasa menghasilkan hasil yang sama.

```
>$areaTriangle(B,A,C)
```

## Rumus Heron

---

Sekarang, mari kita selesaikan masalah ini secara umum!

```
>&remvalue(a,b,c,sb,ha);
```

Pertama kita hitung spread di B untuk segitiga dengan sisi a, b, dan c. Kemudian kita menghitung luas kuadrat ("quadrea"?), faktorkan dengan Maxima, dan kita mendapatkan rumus Heron yang terkenal.

Memang, ini sulit dilakukan dengan pensil dan kertas.

```
>$spread(b^2,c^2,a^2), $factor(%*c^2*a^2/4)
```

## Aturan Triple Spread

---

Kerugian dari spread adalah mereka tidak lagi hanya menambahkan sudut yang sama. Namun, tiga spread dari sebuah segitiga memenuhi aturan "triple spread" berikut.

```
>&remvalue(sa,sb,sc); $triplespread(sa,sb,sc)
```

Aturan ini berlaku untuk setiap tiga sudut yang menambah  $180^\circ$ .

$$\alpha + \beta + \gamma = \pi$$

Sejak menyebar

$$\alpha, \pi - \alpha$$

sama, aturan triple spread juga benar, jika

$$\alpha + \beta = \gamma$$

Karena penyebaran sudut negatif adalah sama, aturan penyebaran rangkap tiga juga berlaku, jika

$$\alpha + \beta + \gamma = 0$$

Misalnya, kita dapat menghitung penyebaran sudut  $60^\circ$ . Ini  $3/4$ . Persamaan memiliki solusi kedua, bagaimanapun, di mana semua spread adalah 0.

```
>$solve(triplespread(x,x,x),x)
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$solve(triplespread(x,x,x),x) ...  
^
```

Sebaran  $90^\circ$  jelas 1. Jika dua sudut dijumlahkan menjadi  $90^\circ$ , sebarannya menyelesaikan persamaan sebaran rangkap tiga dengan a,b,1. Dengan perhitungan berikut kita mendapatkan  $a+b=1$ .

```
>$triplespread(x,y,1), $solve(%,x)
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$triplespread(x,y,1), $solve(%,x) ...  
^
```

Karena sebaran  $180^\circ$ -t sama dengan sebaran t, rumus sebaran rangkap tiga juga berlaku, jika satu sudut adalah jumlah atau selisih dua sudut lainnya.

Jadi kita dapat menemukan penyebaran sudut berlipat ganda. Perhatikan bahwa ada dua solusi lagi. Kami membuat ini fungsi.

```
>$solve(triplespread(a,a,x),x), function doublespread(a) &= factor(rhs(%[1]))
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$solve(triplespread(a,a,x),x), function doublespread(a) &= fac ...  
^
```

## Pembagi Sudut

---

Ini situasinya, kita sudah tahu.

```
>C:=[0,0]; A:=[4,0]; B:=[0,3]; ...  
>setPlotRange(-1,5,-1,5); ...  
>plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C"); ...  
>plotSegment(B,A,"c"); plotSegment(A,C,"b"); plotSegment(C,B,"a"); ...  
>insimg;
```

Mari kita hitung panjang garis bagi sudut di A. Tetapi kita ingin menyelesaiakannya untuk umum a,b,c.

```
>&remvalue(a,b,c);
```

Jadi pertama-tama kita hitung penyebaran sudut yang dibagi dua di A, dengan menggunakan rumus sebaran rangkap tiga.

Masalah dengan rumus ini muncul lagi. Ini memiliki dua solusi. Kita harus memilih yang benar. Solusi lainnya mengacu pada sudut terbelah  $180^\circ$ -wa.

```
>$triplespread(x,x,a/(a+b)), $solve(% ,x), sa2 &= rhs(%[1]); $sa2
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$triplespread(x,x,a/(a+b)), $solve(% ,x), sa2 &= rhs(%[1]); $sa ...  
^
```

Mari kita periksa persegi panjang Mesir.

```
>$sa2 with [a=3^2,b=4^2]
```

Kami dapat mencetak sudut dalam Euler, setelah mentransfer penyebaran ke radian.

```
>wa2 := arcsin(sqrt(1/10)); degprint(wa2)
```

$18^\circ 26' 5.82''$

Titik P adalah perpotongan garis bagi sudut dengan sumbu y.

```
>P := [0,tan(wa2)*4]
```

[0, 1.33333]

```
>plotPoint(P,"P"); plotSegment(A,P):
```

Mari kita periksa sudut dalam contoh spesifik kita.

```
>computeAngle(C,A,P), computeAngle(P,A,B)
```

0.321750554397

0.321750554397

Sekarang kita hitung panjang garis bagi AP.

Kami menggunakan teorema sinus dalam segitiga APC. Teorema ini menyatakan bahwa

$$\frac{BC}{\sin(w_a)} = \frac{AC}{\sin(w_b)} = \frac{AB}{\sin(w_c)}$$

berlaku dalam segitiga apa pun. Kuadratkan, itu diterjemahkan ke dalam apa yang disebut "hukum penyebaran"

$$\frac{a}{s_a} = \frac{b}{s_b} = \frac{c}{s_b}$$

di mana a,b,c menunjukkan qudrances.

Karena spread CPA adalah  $1-sa^2$ , kita dapatkan darinya  $bisa/1=b/(1-sa^2)$  dan dapat menghitung bisa (kuadran dari garis-bagi sudut).

```
>&factor(ratsimp(b/(1-sa2))); bisa &= %; $bisa
```

Mari kita periksa rumus ini untuk nilai-nilai Mesir kita.

```
>sqrt(mxmeval("at(bisa,[a=3^2,b=4^2])")), distance(A,P)
```

```
Variable sa2 not found!
Use global variables or parameters for string evaluation.
Error in Evaluate, superfluous characters found.
Try "trace errors" to inspect local variables after errors.
mxmeval:
    return evaluate(mxm(s));
Error in:
sqrt(mxmeval("at(bisa,[a=3^2,b=4^2])")), distance(A,P) ...
```

Kita juga dapat menghitung P menggunakan rumus spread.

```
>py&=factor(ratsimp(sa2*bisa)); $py
```

Nilainya sama dengan yang kita dapatkan dengan rumus trigonometri.

```
>sqrt(mxmeval("at(py,[a=3^2,b=4^2])"))
```

```
Variable sa2 not found!
Use global variables or parameters for string evaluation.
Error in Evaluate, superfluous characters found.
Try "trace errors" to inspect local variables after errors.
mxmeval:
    return evaluate(mxm(s));
Error in:
sqrt(mxmeval("at(py,[a=3^2,b=4^2])"))
^
```

## Sudut Akord

---

Perhatikan situasi berikut.

```
>setPlotRange(1.2); ...
>color(1); plotCircle(circleWithCenter([0,0],1)); ...
>A:=[cos(1),sin(1)]; B:=[cos(2),sin(2)]; C:=[cos(6),sin(6)]; ...
>plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C"); ...
>color(3); plotSegment(A,B,"c"); plotSegment(A,C,"b"); plotSegment(C,B,"a"); ...
>color(1); O:=[0,0]; plotPoint(O,"O"); ...
>plotSegment(A,O); plotSegment(B,O); plotSegment(C,O,"r"); ...
>insimg;
```

Kita dapat menggunakan Maxima untuk menyelesaikan rumus penyebaran rangkap tiga untuk sudut-sudut di pusat O untuk r. Jadi kita mendapatkan rumus untuk jari-jari kuadrat dari pericircle dalam hal kuadrat dari sisi.

Kali ini, Maxima menghasilkan beberapa nol kompleks, yang kita abaikan.

```
>&remvalue(a,b,c,r); // hapus nilai-nilai sebelumnya untuk perhitungan baru  
>rabc &= rhs(solve(triplespread(spread(b,r,r),spread(a,r,r),spread(c,r,r)),r)[4]); $rabc
```

Kita dapat menjadikannya sebagai fungsi Euler.

```
>function periradius(a,b,c) &= rabc;
```

Mari kita periksa hasilnya untuk poin A,B,C.

```
>a:=quadrance(B,C); b:=quadrance(A,C); c:=quadrance(A,B);
```

Jari-jarinya memang 1.

```
>periradius(a,b,c)
```

Faktanya, spread CBA hanya bergantung pada b dan c. Ini adalah teorema sudut chord.

```
>$spread(b,a,c)*rabc | ratsimp
```

Sebenarnya spreadnya adalah  $b/(4r)$ , dan kita melihat bahwa sudut chord dari chord b adalah setengah dari sudut pusat.

```
>$doublespread(b/(4*r))-spread(b,r,r) | ratsimp
```

### **Contoh 6: Jarak Minimal pada Bidang**

---

#### **Catatan awal**

Fungsi yang, ke titik M di bidang, menetapkan jarak AM antara titik tetap A dan M, memiliki garis level yang agak sederhana: lingkaran berpusat di A.

```
>&remvalue();
>A=[-1,-1];
>function d1(x,y):=sqrt((x-A[1])^2+(y-A[2])^2)
>fcontour("d1",xmin=-2,xmax=0,ymin=-2,ymax=0,hue=1, ...
>title="If you see ellipses, please set your window square"):
```

dan grafiknya juga agak sederhana: bagian atas kerucut:

```
>plot3d("d1",xmin=-2,xmax=0,ymin=-2,ymax=0):
```

Tentu saja minimal 0 dicapai di A.

---

## Dua poin

Sekarang kita lihat fungsi MA+MB dimana A dan B adalah dua titik (tetap). Ini adalah "fakta yang diketahui" bahwa kurva level adalah elips, titik fokusnya adalah A dan B; kecuali untuk AB minimum yang konstan pada segmen [AB]:

```
>B=[1,-1];
>function d2(x,y):=d1(x,y)+sqrt((x-B[1])^2+(y-B[2])^2)
>fcontour("d2",xmin=-2,xmax=2,ymin=-3,ymax=1,hue=1):
```

Grafiknya lebih menarik:

```
>plot3d("d2",xmin=-2,xmax=2,ymin=-3,ymax=1):
```

Pembatasan garis (AB) lebih terkenal:

```
>plot2d("abs(x+1)+abs(x-1)",xmin=-3,xmax=3):
```

---

**Tiga poin**

Sekarang hal-hal yang kurang sederhana: Ini sedikit kurang terkenal bahwa  $MA+MB+MC$  mencapai minimum pada satu titik pesawat tetapi untuk menentukan itu kurang sederhana:

- 1) Jika salah satu sudut segitiga ABC lebih dari  $120^\circ$  (katakanlah di A), maka minimum dicapai pada titik ini (misalnya  $AB+AC$ ).

Contoh:

```

>C=[-4,1];
>function d3(x,y):=d2(x,y)+sqrt((x-C[1])^2+(y-C[2])^2)
>plot3d("d3",xmin=-5,xmax=3,ymin=-4,ymax=4);
>insimg;
>fcontour("d3",xmin=-4,xmax=1,ymin=-2,ymax=2,hue=1,title="The minimum is on A");
>P=(A_B_C_A)'; plot2d(P[1],P[2],add=1,color=12);
>insimg;

```

- 2) Tetapi jika semua sudut segitiga ABC kurang dari  $120^\circ$ , minimumnya adalah pada titik F di bagian dalam segitiga, yang merupakan satu-satunya titik yang melihat sisi-sisi ABC dengan sudut yang sama (maka masing-masing  $120^\circ$ ):

```

>C=[-0.5,1];
>plot3d("d3",xmin=-2,xmax=2,ymin=-2,ymax=2):
>fcontour("d3",xmin=-2,xmax=2,ymin=-2,ymax=2,hue=1,title="The Fermat point");
>P=(A_B_C_A)'; plot2d(P[1],P[2],add=1,color=12);
>insimg;

```

Merupakan kegiatan yang menarik untuk mewujudkan gambar di atas dengan perangkat lunak geometri; misalnya, saya tahu soft yang ditulis di Jawa yang memiliki instruksi "garis kontur" ...

Semua ini di atas telah ditemukan oleh seorang hakim Perancis bernama Pierre de Fermat; dia menulis surat kepada dilettants lain seperti pendeta Marin Mersenne dan Blaise Pascal yang bekerja di pajak penghasilan. Jadi titik unik F sedemikian rupa sehingga  $FA+FB+FC$  minimal, disebut titik Fermat segitiga. Tetapi tampaknya beberapa tahun sebelumnya, Torriccelli Italia telah menemukan titik ini sebelum Fermat melakukannya! Bagaimanapun tradisinya adalah mencatat poin ini F...

Langkah selanjutnya adalah menambahkan 4 titik D dan mencoba meminimalkan  $MA+MB+MC+MD$ ; katakan bahwa Anda adalah operator TV kabel dan ingin mencari di bidang mana Anda harus meletakkan antena sehingga Anda dapat memberi makan empat desa dan menggunakan panjang kabel sesedikit mungkin!

```
>D=[1,1];
>function d4(x,y):=d3(x,y)+sqrt((x-D[1])^2+(y-D[2])^2)
>plot3d("d4",xmin=-1.5,xmax=1.5,ymin=-1.5,ymax=1.5):
>fcontour("d4",xmin=-1.5,xmax=1.5,ymin=-1.5,ymax=1.5,hue=1);
>P=(A_B_C_D)';
>plot2d(P[1],P[2],points=1,add=1,color=12);
>insimg;
```

Masih ada minimum dan tidak tercapai di salah satu simpul A, B, C atau D:

```
>function f(x):=d4(x[1],x[2])
>neldermin("f", [0.2,0.2])
```

[0.142858, 0.142857]

Tampaknya dalam kasus ini, koordinat titik optimal adalah rasional atau mendekati rasional... Sekarang ABCD adalah persegi, kami berharap bahwa titik optimal akan menjadi pusat ABCD:

```
>C=[-1,1];
>plot3d("d4",xmin=-1,xmax=1,ymin=-1,ymax=1):
>fcontour("d4",xmin=-1.5,xmax=1.5,ymin=-1.5,ymax=1.5,hue=1);
>P=(A_B_C_D)'; plot2d(P[1],P[2],add=1,color=12,points=1);
>insimg;
```

### **Contoh 7: Bola Dandelin dengan Povray**

---

Anda dapat menjalankan demonstrasi ini, jika Anda telah menginstal Povray, dan povengine.exe di jalur program.

Pertama kita hitung jari-jari bola.

Jika Anda melihat gambar di bawah, Anda melihat bahwa kita membutuhkan dua lingkaran yang menyentuh dua garis yang membentuk kerucut, dan satu garis yang membentuk bidang yang memotong kerucut.

Kami menggunakan file geometri.e dari Euler untuk ini.

```
>load geometry;
```

Pertama dua garis yang membentuk kerucut.

```
>g1 &= lineThrough([0,0],[1,a])
```

$[- a, 1, 0]$

```
>g2 &= lineThrough([0,0],[-1,a])
```

$[- a, - 1, 0]$

Kemudian saya baris ketiga.

```
>g &= lineThrough([-1,0],[1,1])
```

$[- 1, 2, 1]$

Kami merencanakan semuanya sejauh ini.

```
>setPlotRange(-1,1,0,2);
>color(black); plotLine(g(), "")
>a:=2; color(blue); plotLine(g1(), ""), plotLine(g2(), ""):
```

Sekarang kita ambil titik umum pada sumbu y.

```
>P &= [0,u]
```

[0, u]

Hitung jarak ke g1.

```
>d1 &= distance(P,projectToLine(P,g1)); $d1
```

```
Maxima said:
rat: replaced 1.66665833335744e-7 by 15819/94914474571 = 1.66665833335744e-7

rat: replaced 4.999958333473664e-5 by 201389/4027813565 = 4.99995833347366e-5

rat: replaced 1.33330666692022e-6 by 31771/23828726570 = 1.333306666920221e-6
```

```
rat: replaced 1.999933334222437e-4 by 200030/1000183339 = 1.999933334222437e-4
rat: replaced 4.499797504338432e-6 by 24036/5341573699 = 4.499797504338431e-6
rat: replaced 4.499662510124569e-4 by 1162901/2584418270 = 4.499662510124571e-4
rat: replaced 1.066581336583994e-5 by 58861/5518660226 = 1.066581336583993e-5
rat: replaced 7.998933390220841e-4 by 1137431/1421978337 = 7.998933390220838e-4
rat: replaced 2.083072932167196e-5 by 35635/1710693824 = 2.0830729321672e-5
rat: replaced 0.001249739605033717 by 567943/454449069 = 0.001249739605033716
rat: replaced 3.599352055540239e-5 by 98277/2730408098 = 3.599352055540234e-5
rat: replaced 0.00179946006479581 by 479561/266502719 = 0.001799460064795812
rat: replaced 5.71526624672386e-5 by 51154/895041417 = 5.715266246723866e-5
rat: replaced 0.002448999746720415 by 1946227/794702818 = 0.002448999746720415
rat: replaced 8.530603082730626e-5 by 121691/1426522824 = 8.530603082730627e-5
rat: replaced 0.003198293697380561 by 2986741/933854512 = 0.003198293697380562
rat: replaced 1.214508019889565e-4 by 158455/1304684674 = 1.214508019889563e-4
rat: replaced 0.004047266988005727 by 2125334/525128193 = 0.004047266988005727
rat: replaced 1.665833531718508e-4 by 142521/855553675 = 1.66583353171851e-4
rat: replaced 0.004995834721974179 by 1957223/391770967 = 0.004995834721974179
```

```
rat: replaced 2.216991628251896e-4 by 179571/809975995 = 2.216991628251896e-4
rat: replaced 0.006043902043303184 by 1800665/297930871 = 0.006043902043303193
rat: replaced 2.877927110806339e-4 by 1167733/4057548906 = 2.877927110806339e-4
rat: replaced 0.00719136414613375 by 2476362/344352191 = 0.007191364146133747
rat: replaced 3.658573803051457e-4 by 386279/1055818526 = 3.658573803051454e-4
rat: replaced 0.00843810628521191 by 2079855/246483622 = 0.008438106285211924
rat: replaced 4.5688535576352e-4 by 262978/575588595 = 4.568853557635206e-4
rat: replaced 0.009784003787362772 by 1752551/179124113 = 0.009784003787362787
rat: replaced 5.618675264007778e-4 by 150595/268025812 = 5.618675264007782e-4
rat: replaced 0.01122892206395776 by 5450241/485375263 = 0.01122892206395776
rat: replaced 6.817933857540259e-4 by 192316/282073725 = 6.817933857540258e-4
rat: replaced 0.01277271662437307 by 3258991/255152533 = 0.01277271662437308
rat: replaced 8.176509330039827e-4 by 105841/129445214 = 8.176509330039812e-4
rat: replaced 0.01441523309043924 by 2330472/161667313 = 0.01441523309043925
rat: replaced 9.704265741758145e-4 by 651321/671169790 = 9.704265741758132e-4
rat: replaced 0.01615630721187855 by 19391318/1200232067 = 0.01615630721187855
rat: replaced 0.001141105023499428 by 1259907/1104111343 = 0.001141105023499428
rat: replaced 0.01799576488272969 by 4765614/264818641 = 0.01799576488272969
```

rat: replaced 0.001330669204938795 by 1231154/925214167 = 0.001330669204938796  
rat: replaced 0.01993342215875837 by 2504519/125644206 = 0.01993342215875836  
rat: replaced 0.001540100153900437 by 276884/179783113 = 0.001540100153900439  
rat: replaced 0.02196908527585173 by 1298306/59096953 = 0.0219690852758517  
rat: replaced 0.001770376919130678 by 644389/363984072 = 0.001770376919130681  
rat: replaced 0.02410255066939448 by 2001286/83032125 = 0.02410255066939453  
rat: replaced 0.002022476464811601 by 1271955/628909667 = 0.002022476464811599  
rat: replaced 0.02633360499462523 by 2978115/113091808 = 0.02633360499462525  
rat: replaced 0.002297373572865413 by 1020913/444382669 = 0.002297373572865417  
rat: replaced 0.02866202514797045 by 1770713/61779061 = 0.02866202514797044  
rat: replaced 0.002596040745477063 by 1097643/422814242 = 0.002596040745477065  
rat: replaced 0.03108757828935527 by 5034207/161936287 = 0.03108757828935525  
rat: replaced 0.002919448107844891 by 906221/310408326 = 0.002919448107844891  
rat: replaced 0.03361002186548678 by 4553215/135471944 = 0.03361002186548678  
rat: replaced 0.003268563311168871 by 1379071/421919623 = 0.003268563311168867  
rat: replaced 0.03622910363410947 by 3082649/85087642 = 0.0362291036341094  
rat: replaced 0.003644351435886262 by 5966577/1637212301 = 0.003644351435886261

rat: replaced 0.03894456168922911 by 4913415/126164342 = 0.03894456168922911  
rat: replaced 0.004047774895164447 by 572425/141417202 = 0.004047774895164451  
rat: replaced 0.04175612448730281 by 1734727/41544253 = 0.04175612448730273  
rat: replaced 0.004479793338660443 by 2952779/659132861 = 0.004479793338660444  
rat: replaced 0.04466351087439402 by 4691119/105032473 = 0.04466351087439405  
rat: replaced 0.0049413635565565 by 2524919/510976165 = 0.004941363556556498  
rat: replaced 0.04766643011428662 by 3536207/74186529 = 0.04766643011428665  
rat: replaced 0.005433439383882244 by 1361584/250593391 = 0.005433439383882235  
rat: replaced 0.05076458191755917 by 7710025/151878036 = 0.05076458191755916  
rat: replaced 0.005956971605131645 by 1447422/242979503 = 0.005956971605131648  
rat: replaced 0.0539576564716131 by 3377975/62604183 = 0.05395765647161309  
rat: replaced 0.006512907859185624 by 3695063/567344584 = 0.006512907859185626  
rat: replaced 0.05724533447165381 by 2560865/44734912 = 0.05724533447165382  
rat: replaced 0.007102192544548636 by 1363981/192050693 = 0.007102192544548642  
rat: replaced 0.06062728715262111 by 8274761/136485754 = 0.06062728715262107  
rat: replaced 0.007725766724910044 by 1464384/189545459 = 0.007725766724910038  
rat: replaced 0.06410317632206519 by 5287663/82486755 = 0.06410317632206528  
rat: replaced 0.00838456803503801 by 1113589/132814117 = 0.008384568035038023

rat: replaced 0.06767265439396564 by 2921400/43169579 = 0.06767265439396572  
rat: replaced 0.009079530587017326 by 433906/47789475 = 0.00907953058701733  
rat: replaced 0.07133536442348987 by 7236103/101437808 = 0.07133536442348991  
rat: replaced 0.009811584876838586 by 1363090/138926587 = 0.009811584876838586  
rat: replaced 0.07509094014268702 by 9209133/122639735 = 0.07509094014268704  
rat: replaced 0.0105816576913495 by 1163729/109976058 = 0.01058165769134951  
rat: replaced 0.07893900599711501 by 5197067/65836489 = 0.07893900599711506  
rat: replaced 0.01139067201557714 by 13426050/1178688139 = 0.01139067201557714  
rat: replaced 0.08287917718339499 by 11217158/135343501 = 0.082879177183395  
rat: replaced 0.01223954694042984 by 2283101/186534764 = 0.01223954694042983  
rat: replaced 0.08691105968769186 by 5213115/59982182 = 0.08691105968769192  
rat: replaced 0.01312919757078923 by 3499615/266552086 = 0.01312919757078922  
rat: replaced 0.09103425032511492 by 5893225/64736349 = 0.09103425032511488  
rat: replaced 0.01406053493400045 by 2280713/162206702 = 0.01406053493400045  
rat: replaced 0.09524833678003664 by 9601787/100807923 = 0.09524833678003662  
rat: replaced 0.01503446588876983 by 200490/13335359 = 0.01503446588876985  
rat: replaced 0.09955289764732322 by 5687088/57126293 = 0.09955289764732328

rat: replaced 0.01605189303448024 by  $951971/59305840 = 0.01605189303448025$   
rat: replaced 0.1039475024744748 by  $10260011/98703776 = 0.1039475024744747$   
rat: replaced 0.01711371462093175 by  $9432386/551159477 = 0.01711371462093176$   
rat: replaced 0.1084317118046711 by  $14939691/137779721 = 0.1084317118046712$   
rat: replaced 0.01822082445851714 by  $2559788/140486947 = 0.01822082445851713$   
rat: replaced 0.113005077220716 by  $8478529/75027859 = 0.1130050772207161$   
rat: replaced 0.01937411182884202 by  $2983799/154009589 = 0.01937411182884203$   
rat: replaced 0.1176671413898787 by  $7123715/60541243 = 0.1176671413898786$   
rat: replaced 0.02057446139579705 by  $7167743/348380590 = 0.02057446139579705$   
rat: replaced 0.1224174381096274 by  $12172179/99431741 = 0.1224174381096274$   
rat: replaced 0.02182275311709253 by  $7415562/339808729 = 0.02182275311709253$   
rat: replaced 0.1272554923542488 by  $7277933/57191504 = 0.127255492354249$   
rat: replaced 0.02311986215626333 by  $2988661/129268115 = 0.02311986215626336$   
rat: replaced 0.1321808203223502 by  $3633064/27485561 = 0.1321808203223503$   
rat: replaced 0.02446665879515308 by  $1991976/81415939 = 0.02446665879515312$   
rat: replaced 0.1371929294852391 by  $56235017/409897341 = 0.1371929294852391$   
rat: replaced 0.02586400834688696 by  $5000736/193347293 = 0.02586400834688697$   
rat: replaced 0.1422913186361759 by  $9349741/65708443 = 0.1422913186361759$

rat: replaced 0.02731277106934082 by 858413/31428997 = 0.02731277106934084

rat: replaced 0.1474754779404944 by 1549881/10509415 = 0.1474754779404943

rat: replaced 0.02881380207911666 by 3754753/130310918 = 0.02881380207911666

rat: replaced 0.152744888986584 by 5264425/34465474 = 0.1527448889865841

rat: replaced 0.03036795126603076 by 4118329/135614318 = 0.03036795126603077

rat: replaced 0.1580990248377314 by 5442776/34426373 = 0.1580990248377312

rat: replaced 0.03197606320812652 by 3497683/109384416 = 0.03197606320812647

rat: replaced 0.1635373500848132 by 12328488/75386375 = 0.1635373500848131

rat: replaced 0.0336389770872163 by 3971799/118071337 = 0.03363897708721635

rat: replaced 0.1690593208998367 by 20896917/123607009 = 0.1690593208998367

rat: replaced 0.03535752660496472 by 1815732/51353479 = 0.03535752660496478

rat: replaced 0.1746643850903219 by 2841592/16268869 = 0.1746643850903219

rat: replaced 0.03713253989951881 by 3333721/89778965 = 0.03713253989951878

rat: replaced 0.1803519821545206 by 4461007/24735004 = 0.1803519821545208

rat: replaced 0.03896483946269502 by 8785771/225479461 = 0.03896483946269501

rat: replaced 0.1861215433374662 by 4381209/23539505 = 0.1861215433374661

rat: replaced 0.0408552420577305 by 3189084/78058135 = 0.04085524205773043

rat: replaced 0.1919724916878484 by 72809759/379271834 = 0.1919724916878484  
rat: replaced 0.04280455863760801 by 7646593/178639688 = 0.04280455863760801  
rat: replaced 0.1979042421157076 by 26318167/132984350 = 0.1979042421157076  
rat: replaced 0.04481359426396048 by 20610430/459914683 = 0.04481359426396048  
rat: replaced 0.2039162014509444 by 8519416/41779005 = 0.2039162014509441  
rat: replaced 0.04688314802656623 by 3439140/73355569 = 0.04688314802656633  
rat: replaced 0.2100077685026351 by 50962787/242670961 = 0.2100077685026351  
rat: replaced 0.04901401296344043 by 4006732/81746663 = 0.04901401296344048  
rat: replaced 0.216178334119151 by 1347531/6233423 = 0.2161783341191509  
rat: replaced 0.05120697598153157 by 4148974/81023609 = 0.0512069759815315  
rat: replaced 0.2224272812490723 by 23234851/104460437 = 0.2224272812490723  
rat: replaced 0.05346281777803219 by 11998448/224426031 = 0.05346281777803218  
rat: replaced 0.2287539850028937 by 8185268/35781969 = 0.2287539850028935  
rat: replaced 0.05578231276230905 by 1398019/25062048 = 0.05578231276230897  
rat: replaced 0.2351578127155118 by 12642104/53760085 = 0.2351578127155119  
rat: replaced 0.05816622897846346 by 4451048/76522891 = 0.05816622897846345  
rat: replaced 0.2416381240094921 by 8002142/33116223 = 0.2416381240094923  
rat: replaced 0.06061532802852698 by 2146337/35409146 = 0.06061532802852686

rat: replaced 0.2481942708591053 by 8882901/35790113 = 0.2481942708591057  
rat: replaced 0.0631303649963022 by 14651447/232082406 = 0.06313036499630222  
rat: replaced 0.2548255976551299 by 868346/3407609 = 0.25482559765513  
rat: replaced 0.06571208837185505 by 4240309/64528599 = 0.06571208837185509  
rat: replaced 0.2615314412704124 by 8212450/31401387 = 0.2615314412704127  
rat: replaced 0.06836123997666599 by 2716643/39739522 = 0.06836123997666604  
rat: replaced 0.2683111311261794 by 34459769/128432126 = 0.2683111311261794  
rat: replaced 0.07107855488944881 by 3146673/44270357 = 0.07107855488944893  
rat: replaced 0.2751639892590951 by 12552159/45617012 = 0.2751639892590949  
rat: replaced 0.07386476137264342 by 12898997/174629915 = 0.0738647613726434  
rat: replaced 0.2820893303890569 by 11134456/39471383 = 0.2820893303890568  
rat: replaced 0.07672058079958999 by 5073506/66129661 = 0.07672058079959007  
rat: replaced 0.2890864619877229 by 9583357/33150487 = 0.2890864619877228  
rat: replaced 0.07964672758239233 by 5672399/71219486 = 0.07964672758239227  
rat: replaced 0.2961546843477643 by 11052271/37319251 = 0.2961546843477647  
rat: replaced 0.08264390910047736 by 4686067/56701904 = 0.08264390910047748  
rat: replaced 0.3032932906528349 by 9918077/32701274 = 0.3032932906528351

rat: replaced 0.0857128256298576 by  $3585977/41837111 = 0.08571282562985766$   
rat: replaced 0.3105015670482534 by  $9320011/30015987 = 0.3105015670482533$   
rat: replaced 0.08885417027310427 by  $5751353/64728003 = 0.0888541702731042$   
rat: replaced 0.3177787927123868 by  $248395525/781661743 = 0.3177787927123868$   
rat: replaced 0.09206862889003742 by  $7305460/79347983 = 0.09206862889003745$   
rat: replaced 0.3251242399287333 by  $13842845/42577093 = 0.3251242399287335$   
rat: replaced 0.09535688002914089 by  $5971998/62627867 = 0.09535688002914103$   
rat: replaced 0.3325371741586922 by  $9318229/28021616 = 0.3325371741586923$   
rat: replaced 0.0987195948597075 by  $9821211/99485933 = 0.09871959485970745$   
rat: replaced 0.3400168541150183 by  $13391981/39386227 = 0.3400168541150184$   
rat: replaced 0.1021574371047232 by  $8336413/81603584 = 0.1021574371047232$   
rat: replaced 0.3475625318359485 by  $10097818/29053241 = 0.347562531835949$   
rat: replaced 0.1056710629744951 by  $5741011/54329074 = 0.105671062974495$   
rat: replaced 0.3551734527599992 by  $15867851/44676343 = 0.3551734527599987$   
rat: replaced 0.1092611211010309 by  $5551873/50812887 = 0.1092611211010309$   
rat: replaced 0.3628488558014202 by  $6897641/19009681 = 0.3628488558014203$   
rat: replaced 0.1129282524731764 by  $11548693/102265755 = 0.1129282524731764$   
rat: replaced 0.3705879734263036 by  $23358661/63031352 = 0.3705879734263038$

```
rat: replaced 0.1166730903725168 by 5656228/48479285 = 0.1166730903725168
rat: replaced 0.3783900317293359 by 14241382/37636779 = 0.3783900317293358
rat: replaced 0.1204962603100498 by 4057613/33674182 = 0.12049626031005
rat: replaced 0.3862542505111889 by 3461217/8960981 = 0.3862542505111884
rat: replaced 0.1243983799636342 by 7966447/64039797 = 0.1243983799636342
rat: replaced 0.3941798433565377 by 5314214/13481699 = 0.3941798433565384
rat: replaced 0.1283800591162231 by 796346/6203035 = 0.1283800591162229
rat: replaced 0.4021660177127022 by 11567173/28762184 = 0.4021660177127022
rat: replaced 0.1324418995948859 by 4716124/35609003 = 0.1324418995948862
rat: replaced 0.4102119749689023 by 11320633/27597032 = 0.4102119749689024
rat: replaced 0.1365844952106265 by 612971/4487852 = 0.1365844952106264
rat: replaced 0.418316910536117 by 12225195/29224721 = 0.4183169105361177
rat: replaced 0.140808431699002 by 10431632/74083859 = 0.1408084316990021
rat: replaced 0.4264800139275439 by 7978696/18708253 = 0.4264800139275431
rat: replaced 0.1451142866615502 by 3554077/24491572 = 0.1451142866615504
rat: replaced 0.4347004688396462 by 20489554/47134879 = 0.4347004688396463
rat: replaced 0.1495026295080298 by 26759297/178988805 = 0.1495026295080298
```

```
rat: replaced 0.4429774532337832 by 23449796/52936771 = 0.4429774532337834
rat: replaced 0.1539740213994798 by 16145763/104860306 = 0.1539740213994798
rat: replaced 0.451310139418413 by 8841241/19590167 = 0.4513101394184133
rat: replaced 4.999958333473664e-5 by 201389/4027813565 = 4.99995833347366e-5
rat: replaced -1.66665833335744e-7 by -15819/94914474571 = -1.66665833335744e-7
rat: replaced 1.999933334222437e-4 by 200030/1000183339 = 1.999933334222437e-4
rat: replaced -1.33330666692022e-6 by -31771/23828726570 = -1.333306666920221e-6
rat: replaced 4.499662510124569e-4 by 1162901/2584418270 = 4.499662510124571e-4
rat: replaced -4.499797504338432e-6 by -24036/5341573699 = -4.499797504338431e-6
rat: replaced 7.998933390220841e-4 by 1137431/1421978337 = 7.998933390220838e-4
rat: replaced -1.066581336583994e-5 by -58861/5518660226 = -1.066581336583993e-5
rat: replaced 0.001249739605033717 by 567943/454449069 = 0.001249739605033716
rat: replaced -2.083072932167196e-5 by -35635/1710693824 = -2.0830729321672e-5
rat: replaced 0.00179946006479581 by 479561/266502719 = 0.001799460064795812
rat: replaced -3.599352055540239e-5 by -98277/2730408098 = -3.599352055540234e-5
rat: replaced 0.002448999746720415 by 1946227/794702818 = 0.002448999746720415
rat: replaced -5.71526624672386e-5 by -51154/895041417 = -5.715266246723866e-5
rat: replaced 0.003198293697380561 by 2986741/933854512 = 0.003198293697380562
```

```
rat: replaced -8.530603082730626e-5 by -121691/1426522824 = -8.530603082730627e-5
rat: replaced 0.004047266988005727 by 2125334/525128193 = 0.004047266988005727
rat: replaced -1.214508019889565e-4 by -158455/1304684674 = -1.214508019889563e-4
rat: replaced 0.004995834721974179 by 1957223/391770967 = 0.004995834721974179
rat: replaced -1.665833531718508e-4 by -142521/855553675 = -1.66583353171851e-4
rat: replaced 0.006043902043303184 by 1800665/297930871 = 0.006043902043303193
rat: replaced -2.216991628251896e-4 by -179571/809975995 = -2.216991628251896e-4
rat: replaced 0.00719136414613375 by 2476362/344352191 = 0.007191364146133747
rat: replaced -2.877927110806339e-4 by -1167733/4057548906 = -2.877927110806339e-4
rat: replaced 0.00843810628521191 by 2079855/246483622 = 0.008438106285211924
rat: replaced -3.658573803051457e-4 by -386279/1055818526 = -3.658573803051454e-4
rat: replaced 0.009784003787362772 by 1752551/179124113 = 0.009784003787362787
rat: replaced -4.5688535576352e-4 by -262978/575588595 = -4.568853557635206e-4
rat: replaced 0.01122892206395776 by 5450241/485375263 = 0.01122892206395776
rat: replaced -5.618675264007778e-4 by -150595/268025812 = -5.618675264007782e-4
rat: replaced 0.01277271662437307 by 3258991/255152533 = 0.01277271662437308
rat: replaced -6.817933857540259e-4 by -192316/282073725 = -6.817933857540258e-4
```

rat: replaced 0.01441523309043924 by 2330472/161667313 = 0.01441523309043925  
rat: replaced -8.176509330039827e-4 by -105841/129445214 = -8.176509330039812e-4  
rat: replaced 0.01615630721187855 by 19391318/1200232067 = 0.01615630721187855  
rat: replaced -9.704265741758145e-4 by -651321/671169790 = -9.704265741758132e-4  
rat: replaced 0.01799576488272969 by 4765614/264818641 = 0.01799576488272969  
rat: replaced -0.001141105023499428 by -1259907/1104111343 = -0.001141105023499428  
rat: replaced 0.01993342215875837 by 2504519/125644206 = 0.01993342215875836  
rat: replaced -0.001330669204938795 by -1231154/925214167 = -0.001330669204938796  
rat: replaced 0.02196908527585173 by 1298306/59096953 = 0.0219690852758517  
rat: replaced -0.001540100153900437 by -276884/179783113 = -0.001540100153900439  
rat: replaced 0.02410255066939448 by 2001286/83032125 = 0.02410255066939453  
rat: replaced -0.001770376919130678 by -644389/363984072 = -0.001770376919130681  
rat: replaced 0.02633360499462523 by 2978115/113091808 = 0.02633360499462525  
rat: replaced -0.002022476464811601 by -1271955/628909667 = -0.002022476464811599  
rat: replaced 0.02866202514797045 by 1770713/61779061 = 0.02866202514797044  
rat: replaced -0.002297373572865413 by -1020913/444382669 = -0.002297373572865417  
rat: replaced 0.03108757828935527 by 5034207/161936287 = 0.03108757828935525  
rat: replaced -0.002596040745477063 by -1097643/422814242 = -0.002596040745477065

rat: replaced 0.03361002186548678 by 4553215/135471944 = 0.03361002186548678  
rat: replaced -0.002919448107844891 by -906221/310408326 = -0.002919448107844891  
rat: replaced 0.03622910363410947 by 3082649/85087642 = 0.0362291036341094  
rat: replaced -0.003268563311168871 by -1379071/421919623 = -0.003268563311168867  
rat: replaced 0.03894456168922911 by 4913415/126164342 = 0.03894456168922911  
rat: replaced -0.003644351435886262 by -5966577/1637212301 = -0.003644351435886261  
rat: replaced 0.04175612448730281 by 1734727/41544253 = 0.04175612448730273  
rat: replaced -0.004047774895164447 by -572425/141417202 = -0.004047774895164451  
rat: replaced 0.04466351087439402 by 4691119/105032473 = 0.04466351087439405  
rat: replaced -0.004479793338660443 by -2952779/659132861 = -0.004479793338660444  
rat: replaced 0.04766643011428662 by 3536207/74186529 = 0.04766643011428665  
rat: replaced -0.0049413635565565 by -2524919/510976165 = -0.004941363556556498  
rat: replaced 0.05076458191755917 by 7710025/151878036 = 0.05076458191755916  
rat: replaced -0.005433439383882244 by -1361584/250593391 = -0.005433439383882235  
rat: replaced 0.0539576564716131 by 3377975/62604183 = 0.05395765647161309  
rat: replaced -0.005956971605131645 by -1447422/242979503 = -0.005956971605131648  
rat: replaced 0.05724533447165381 by 2560865/44734912 = 0.05724533447165382

rat: replaced  $-0.006512907859185624$  by  $-3695063/567344584 = -0.006512907859185626$   
rat: replaced  $0.06062728715262111$  by  $8274761/136485754 = 0.06062728715262107$   
rat: replaced  $-0.007102192544548636$  by  $-1363981/192050693 = -0.007102192544548642$   
rat: replaced  $0.06410317632206519$  by  $5287663/82486755 = 0.06410317632206528$   
rat: replaced  $-0.007725766724910044$  by  $-1464384/189545459 = -0.007725766724910038$   
rat: replaced  $0.06767265439396564$  by  $2921400/43169579 = 0.06767265439396572$   
rat: replaced  $-0.00838456803503801$  by  $-1113589/132814117 = -0.008384568035038023$   
rat: replaced  $0.07133536442348987$  by  $7236103/101437808 = 0.07133536442348991$   
rat: replaced  $-0.009079530587017326$  by  $-433906/47789475 = -0.00907953058701733$   
rat: replaced  $0.07509094014268702$  by  $9209133/122639735 = 0.07509094014268704$   
rat: replaced  $-0.009811584876838586$  by  $-1363090/138926587 = -0.009811584876838586$   
rat: replaced  $0.07893900599711501$  by  $5197067/65836489 = 0.07893900599711506$   
rat: replaced  $-0.0105816576913495$  by  $-1163729/109976058 = -0.01058165769134951$   
rat: replaced  $0.08287917718339499$  by  $11217158/135343501 = 0.082879177183395$   
rat: replaced  $-0.01139067201557714$  by  $-13426050/1178688139 = -0.01139067201557714$   
rat: replaced  $0.08691105968769186$  by  $5213115/59982182 = 0.08691105968769192$   
rat: replaced  $-0.01223954694042984$  by  $-2283101/186534764 = -0.01223954694042983$   
rat: replaced  $0.09103425032511492$  by  $5893225/64736349 = 0.09103425032511488$

rat: replaced  $-0.01312919757078923$  by  $-3499615/266552086 = -0.01312919757078922$

rat: replaced  $0.09524833678003664$  by  $9601787/100807923 = 0.09524833678003662$

rat: replaced  $-0.01406053493400045$  by  $-2280713/162206702 = -0.01406053493400045$

rat: replaced  $0.09955289764732322$  by  $5687088/57126293 = 0.09955289764732328$

rat: replaced  $-0.01503446588876983$  by  $-200490/13335359 = -0.01503446588876985$

rat: replaced  $0.1039475024744748$  by  $10260011/98703776 = 0.1039475024744747$

rat: replaced  $-0.01605189303448024$  by  $-951971/59305840 = -0.01605189303448025$

rat: replaced  $0.1084317118046711$  by  $14939691/137779721 = 0.1084317118046712$

rat: replaced  $-0.01711371462093175$  by  $-9432386/551159477 = -0.01711371462093176$

rat: replaced  $0.113005077220716$  by  $8478529/75027859 = 0.1130050772207161$

rat: replaced  $-0.01822082445851714$  by  $-2559788/140486947 = -0.01822082445851713$

rat: replaced  $0.1176671413898787$  by  $7123715/60541243 = 0.1176671413898786$

rat: replaced  $-0.01937411182884202$  by  $-2983799/154009589 = -0.01937411182884203$

rat: replaced  $0.1224174381096274$  by  $12172179/99431741 = 0.1224174381096274$

rat: replaced  $-0.02057446139579705$  by  $-7167743/348380590 = -0.02057446139579705$

rat: replaced  $0.1272554923542488$  by  $7277933/57191504 = 0.127255492354249$

rat: replaced  $-0.02182275311709253$  by  $-7415562/339808729 = -0.02182275311709253$

```
rat: replaced 0.1321808203223502 by 3633064/27485561 = 0.1321808203223503
rat: replaced -0.02311986215626333 by -2988661/129268115 = -0.02311986215626336
rat: replaced 0.1371929294852391 by 56235017/409897341 = 0.1371929294852391
rat: replaced -0.02446665879515308 by -1991976/81415939 = -0.02446665879515312
rat: replaced 0.1422913186361759 by 9349741/65708443 = 0.1422913186361759
rat: replaced -0.02586400834688696 by -5000736/193347293 = -0.02586400834688697
rat: replaced 0.1474754779404944 by 1549881/10509415 = 0.1474754779404943
rat: replaced -0.02731277106934082 by -858413/31428997 = -0.02731277106934084
rat: replaced 0.152744888986584 by 5264425/34465474 = 0.1527448889865841
rat: replaced -0.02881380207911666 by -3754753/130310918 = -0.02881380207911666
rat: replaced 0.1580990248377314 by 5442776/34426373 = 0.1580990248377312
rat: replaced -0.03036795126603076 by -4118329/135614318 = -0.03036795126603077
rat: replaced 0.1635373500848132 by 12328488/75386375 = 0.1635373500848131
rat: replaced -0.03197606320812652 by -3497683/109384416 = -0.03197606320812647
rat: replaced 0.1690593208998367 by 20896917/123607009 = 0.1690593208998367
rat: replaced -0.0336389770872163 by -3971799/118071337 = -0.03363897708721635
rat: replaced 0.1746643850903219 by 2841592/16268869 = 0.1746643850903219
rat: replaced -0.03535752660496472 by -1815732/51353479 = -0.03535752660496478
```

```
rat: replaced 0.1803519821545206 by 4461007/24735004 = 0.1803519821545208
rat: replaced -0.03713253989951881 by -3333721/89778965 = -0.03713253989951878
rat: replaced 0.1861215433374662 by 4381209/23539505 = 0.1861215433374661
rat: replaced -0.03896483946269502 by -8785771/225479461 = -0.03896483946269501
rat: replaced 0.1919724916878484 by 72809759/379271834 = 0.1919724916878484
rat: replaced -0.0408552420577305 by -3189084/78058135 = -0.04085524205773043
rat: replaced 0.1979042421157076 by 26318167/132984350 = 0.1979042421157076
rat: replaced -0.04280455863760801 by -7646593/178639688 = -0.04280455863760801
rat: replaced 0.2039162014509444 by 8519416/41779005 = 0.2039162014509441
rat: replaced -0.04481359426396048 by -20610430/459914683 = -0.04481359426396048
rat: replaced 0.2100077685026351 by 50962787/242670961 = 0.2100077685026351
rat: replaced -0.04688314802656623 by -3439140/73355569 = -0.04688314802656633
rat: replaced 0.216178334119151 by 1347531/6233423 = 0.2161783341191509
rat: replaced -0.04901401296344043 by -4006732/81746663 = -0.04901401296344048
rat: replaced 0.2224272812490723 by 23234851/104460437 = 0.2224272812490723
rat: replaced -0.05120697598153157 by -4148974/81023609 = -0.0512069759815315
rat: replaced 0.2287539850028937 by 8185268/35781969 = 0.2287539850028935
```

```
rat: replaced -0.05346281777803219 by -11998448/224426031 = -0.05346281777803218
rat: replaced 0.2351578127155118 by 12642104/53760085 = 0.2351578127155119
rat: replaced -0.05578231276230905 by -1398019/25062048 = -0.05578231276230897
rat: replaced 0.2416381240094921 by 8002142/33116223 = 0.2416381240094923
rat: replaced -0.05816622897846346 by -4451048/76522891 = -0.05816622897846345
rat: replaced 0.2481942708591053 by 8882901/35790113 = 0.2481942708591057
rat: replaced -0.06061532802852698 by -2146337/35409146 = -0.06061532802852686
rat: replaced 0.2548255976551299 by 868346/3407609 = 0.25482559765513
rat: replaced -0.0631303649963022 by -14651447/232082406 = -0.06313036499630222
rat: replaced 0.2615314412704124 by 8212450/31401387 = 0.2615314412704127
rat: replaced -0.06571208837185505 by -4240309/64528599 = -0.06571208837185509
rat: replaced 0.2683111311261794 by 34459769/128432126 = 0.2683111311261794
rat: replaced -0.06836123997666599 by -2716643/39739522 = -0.06836123997666604
rat: replaced 0.2751639892590951 by 12552159/45617012 = 0.2751639892590949
rat: replaced -0.07107855488944881 by -3146673/44270357 = -0.07107855488944893
rat: replaced 0.2820893303890569 by 11134456/39471383 = 0.2820893303890568
rat: replaced -0.07386476137264342 by -12898997/174629915 = -0.0738647613726434
rat: replaced 0.2890864619877229 by 9583357/33150487 = 0.2890864619877228
```

rat: replaced  $-0.07672058079958999$  by  $-5073506/66129661 = -0.07672058079959007$

rat: replaced  $0.2961546843477643$  by  $11052271/37319251 = 0.2961546843477647$

rat: replaced  $-0.07964672758239233$  by  $-5672399/71219486 = -0.07964672758239227$

rat: replaced  $0.3032932906528349$  by  $9918077/32701274 = 0.3032932906528351$

rat: replaced  $-0.08264390910047736$  by  $-4686067/56701904 = -0.08264390910047748$

rat: replaced  $0.3105015670482534$  by  $9320011/30015987 = 0.3105015670482533$

rat: replaced  $-0.0857128256298576$  by  $-3585977/41837111 = -0.08571282562985766$

rat: replaced  $0.3177787927123868$  by  $248395525/781661743 = 0.3177787927123868$

rat: replaced  $-0.08885417027310427$  by  $-5751353/64728003 = -0.0888541702731042$

rat: replaced  $0.3251242399287333$  by  $13842845/42577093 = 0.3251242399287335$

rat: replaced  $-0.09206862889003742$  by  $-7305460/79347983 = -0.09206862889003745$

rat: replaced  $0.3325371741586922$  by  $9318229/28021616 = 0.3325371741586923$

rat: replaced  $-0.09535688002914089$  by  $-5971998/62627867 = -0.09535688002914103$

rat: replaced  $0.3400168541150183$  by  $13391981/39386227 = 0.3400168541150184$

rat: replaced  $-0.0987195948597075$  by  $-9821211/99485933 = -0.09871959485970745$

rat: replaced  $0.3475625318359485$  by  $10097818/29053241 = 0.347562531835949$

rat: replaced  $-0.1021574371047232$  by  $-8336413/81603584 = -0.1021574371047232$

```
rat: replaced 0.3551734527599992 by 15867851/44676343 = 0.3551734527599987
rat: replaced -0.1056710629744951 by -5741011/54329074 = -0.105671062974495
rat: replaced 0.3628488558014202 by 6897641/19009681 = 0.3628488558014203
rat: replaced -0.1092611211010309 by -5551873/50812887 = -0.1092611211010309
rat: replaced 0.3705879734263036 by 23358661/63031352 = 0.3705879734263038
rat: replaced -0.1129282524731764 by -11548693/102265755 = -0.1129282524731764
rat: replaced 0.3783900317293359 by 14241382/37636779 = 0.3783900317293358
rat: replaced -0.1166730903725168 by -5656228/48479285 = -0.1166730903725168
rat: replaced 0.3862542505111889 by 3461217/8960981 = 0.3862542505111884
rat: replaced -0.1204962603100498 by -4057613/33674182 = -0.12049626031005
rat: replaced 0.3941798433565377 by 5314214/13481699 = 0.3941798433565384
rat: replaced -0.1243983799636342 by -7966447/64039797 = -0.1243983799636342
rat: replaced 0.4021660177127022 by 11567173/28762184 = 0.4021660177127022
rat: replaced -0.1283800591162231 by -796346/6203035 = -0.1283800591162229
rat: replaced 0.4102119749689023 by 11320633/27597032 = 0.4102119749689024
rat: replaced -0.1324418995948859 by -4716124/35609003 = -0.1324418995948862
rat: replaced 0.418316910536117 by 12225195/29224721 = 0.4183169105361177
rat: replaced -0.1365844952106265 by -612971/4487852 = -0.1365844952106264
```

```

rat: replaced 0.4264800139275439 by 7978696/18708253 = 0.4264800139275431

rat: replaced -0.140808431699002 by -10431632/74083859 = -0.1408084316990021

rat: replaced 0.4347004688396462 by 20489554/47134879 = 0.4347004688396463

rat: replaced -0.1451142866615502 by -3554077/24491572 = -0.1451142866615504

rat: replaced 0.4429774532337832 by 23449796/52936771 = 0.4429774532337834

rat: replaced -0.1495026295080298 by -26759297/178988805 = -0.1495026295080298

rat: replaced 0.451310139418413 by 8841241/19590167 = 0.4513101394184133

rat: replaced -0.1539740213994798 by -16145763/104860306 = -0.1539740213994798
part: invalid index of list or matrix.
#0: lineIntersection(g=[1,a,a*u],h=[-a,1,0])
#1: projectToLine(a=[0,u],g=[-a,1,0])
-- an error. To debug this try: debugmode(true);

Error in:
d1 &= distance(P,projectToLine(P,g1));
^

```

Hitung jarak ke g.

```
>d &= distance(P,projectToLine(P,g)); $d
```

Maxima said:

```
rat: replaced 5.033291500140813e-5 by 263336/5231884543 = 5.033291500140813e-5
```

rat: replaced  $2.026599467560841e-4$  by  $407727/2011877564 = 2.02659946756084e-4$   
rat: replaced  $4.589658460211338e-4$  by  $352373/767754296 = 4.589658460211339e-4$   
rat: replaced  $8.21224965753764e-4$  by  $219501/267284860 = 8.212249657537654e-4$   
rat: replaced  $0.001291401063677061$  by  $174589/135193477 = 0.001291401063677059$   
rat: replaced  $0.001871447105906615$  by  $1078337/576204904 = 0.001871447105906617$   
rat: replaced  $0.002563305071654892$  by  $1323915/516487489 = 0.002563305071654891$   
rat: replaced  $0.003368905759035173$  by  $820537/243561874 = 0.003368905759035176$   
rat: replaced  $0.00429016859198364$  by  $7572857/1765165363 = 0.00429016859198364$   
rat: replaced  $0.005329001428317881$  by  $3020890/566877311 = 0.005329001428317882$   
rat: replaced  $0.006487300368953564$  by  $2580732/397812935 = 0.006487300368953564$   
rat: replaced  $0.007766949568295017$  by  $1049181/135082762 = 0.007766949568295028$   
rat: replaced  $0.009169821045822202$  by  $2408608/262666849 = 0.009169821045822193$   
rat: replaced  $0.01069777449888981$  by  $2325322/217365023 = 0.01069777449888982$   
rat: replaced  $0.01235265711675931$  by  $7449711/603085711 = 0.01235265711675931$   
rat: replaced  $0.01413630339588112$  by  $3774568/267012379 = 0.01413630339588113$   
rat: replaced  $0.0160505349564472$  by  $2619104/163178611 = 0.0160505349564472$   
rat: replaced  $0.01809716036023018$  by  $3107690/171722521 = 0.01809716036023021$

rat: replaced 0.02027797492972855 by 6791343/334912289 = 0.02027797492972854  
rat: replaced 0.02259476056863596 by 2685790/118867823 = 0.02259476056863597  
rat: replaced 0.0250492855836526 by 2956693/118035023 = 0.02504928558365258  
rat: replaced 0.02764330450765584 by 2138111/77346433 = 0.02764330450765583  
rat: replaced 0.03037855792424843 by 1678577/55255322 = 0.03037855792424846  
rat: replaced 0.03325677229370128 by 1488397/44754704 = 0.03325677229370124  
rat: replaced 0.03627965978030939 by 3229091/89005548 = 0.03627965978030943  
rat: replaced 0.03944891808117656 by 6094420/154488901 = 0.03944891808117659  
rat: replaced 0.04276623025644721 by 206826/4836199 = 0.04276623025644726  
rat: replaced 0.04623326456100163 by 7175941/155211644 = 0.04623326456100162  
rat: replaced 0.04985167427763171 by 2856261/57295187 = 0.04985167427763173  
rat: replaced 0.0536230975517149 by 8075629/150599823 = 0.05362309755171492  
rat: replaced 0.05754915722739962 by 12314906/213989337 = 0.05754915722739961  
rat: replaced 0.06163146068532366 by 10145753/164619707 = 0.06163146068532366  
rat: replaced 0.06587159968187639 by 5154956/78257641 = 0.06587159968187643  
rat: replaced 0.07027115019002506 by 3189686/45391117 = 0.07027115019002507  
rat: replaced 0.07483167224171838 by 4757796/63579977 = 0.0748316722417185  
rat: replaced 0.07955470977188528 by 7059961/88743470 = 0.07955470977188518

rat: replaced 0.08444179046404166 by 17285418/204702173 = 0.08444179046404163  
rat: replaced 0.08949442559752452 by 6119169/68374862 = 0.08949442559752442  
rat: replaced 0.0947141098963642 by 2739857/28927654 = 0.09471410989636422  
rat: replaced 0.100102321379814 by 21380147/213582929 = 0.100102321379814  
rat: replaced 0.1056605212145493 by 8628153/81659194 = 0.1056605212145493  
rat: replaced 0.1113901535685515 by 4925969/44222661 = 0.1113901535685517  
rat: replaced 0.1172926454666934 by 7052303/60125705 = 0.1172926454666935  
rat: replaced 0.1233694066480375 by 17851649/144700777 = 0.1233694066480376  
rat: replaced 0.1296218294248629 by 13037238/100579031 = 0.1296218294248629  
rat: replaced 0.1360512885434353 by 20468361/150445918 = 0.1360512885434353  
rat: replaced 0.1426591410465347 by 8451499/59242604 = 0.1426591410465347  
rat: replaced 0.1494467261377502 by 40350618/270000013 = 0.1494467261377502  
rat: replaced 0.1564153650475627 by 30759845/196654881 = 0.1564153650475627  
rat: replaced 0.1635663609012215 by 11970848/73186491 = 0.1635663609012215  
rat: replaced 0.1709009985884339 by 3726835/21806982 = 0.1709009985884337  
rat: replaced 0.1784205446348769 by 7050541/39516419 = 0.178420544634877  
rat: replaced 0.1861262470755453 by 7913431/42516470 = 0.1861262470755451

rat: replaced 0.1940193353299499 by 15356416/79148895 = 0.19401933532995  
rat: replaced 0.2021010200791761 by 21517868/106470853 = 0.202101020079176  
rat: replaced 0.2103724931448173 by 10133132/48167571 = 0.2103724931448173  
rat: replaced 0.2188349273697929 by 14393696/65774217 = 0.2188349273697929  
rat: replaced 0.2274894765010662 by 2362445/10384854 = 0.2274894765010659  
rat: replaced 0.2363372750742693 by 14238388/60246053 = 0.2363372750742692  
rat: replaced 0.2453794383002513 by 11843947/48267887 = 0.2453794383002513  
rat: replaced 0.2546170619535583 by 10437767/40993981 = 0.2546170619535585  
rat: replaced 0.2640512222628563 by 18572095/70335198 = 0.2640512222628562  
rat: replaced 0.2736829758033094 by 25733021/94024924 = 0.2736829758033094  
rat: replaced 0.2835133593909236 by 5354031/18884581 = 0.2835133593909232  
rat: replaced 0.2935433899788653 by 33562265/114334937 = 0.2935433899788654  
rat: replaced 0.3037740645557676 by 12785981/42090430 = 0.3037740645557672  
rat: replaced 0.3142063600460319 by 13879096/44171913 = 0.314206360046032  
rat: replaced 0.3248412332121354 by 13048490/40168823 = 0.3248412332121357  
rat: replaced 0.3356796205589581 by 12520681/37299497 = 0.3356796205589582  
rat: replaced 0.3467224382401299 by 27133151/78256115 = 0.3467224382401299  
rat: replaced 0.3579705819664191 by 32019579/89447515 = 0.3579705819664191

rat: replaced 0.3694249269161592 by 12845283/34771024 = 0.3694249269161587  
rat: replaced 0.3810863276477343 by 12790304/33562747 = 0.381086327647734  
rat: replaced 0.3929556180141225 by 27557157/70127912 = 0.3929556180141225  
rat: replaced 0.4050336110795114 by 12582391/31065054 = 0.4050336110795107  
rat: replaced 0.4173210990379927 by 17616979/42214446 = 0.4173210990379928  
rat: replaced 0.4298188531343438 by 28764336/66921997 = 0.4298188531343439  
rat: replaced 0.4425276235869029 by 52612738/118891421 = 0.4425276235869029  
rat: replaced 0.4554481395125489 by 12438812/27311149 = 0.4554481395125485  
rat: replaced 0.4685811088537897 by 12910499/27552325 = 0.46858110885379  
rat: replaced 0.4819272183079686 by 11623658/24119115 = 0.4819272183079686  
rat: replaced 0.4954871332585954 by 40137729/81006602 = 0.4954871332585954  
rat: replaced 0.5092614977088081 by 27060617/53136978 = 0.5092614977088084  
rat: replaced 0.523250934216974 by 57357723/109618004 = 0.5232509342169741  
rat: replaced 0.5374560438344332 by 19984722/37183919 = 0.5374560438344328  
rat: replaced 0.551877406045395 by 10637804/19275665 = 0.5518774060453946  
rat: replaced 0.5665155787089895 by 22241852/39260795 = 0.5665155787089895  
rat: replaced 0.5813710980034821 by 10844268/18652919 = 0.5813710980034814

```
rat: replaced 0.5964444783726564 by 13079224/21928653 = 0.596444478372657
rat: replaced 0.6117362124743696 by 11199699/18308053 = 0.6117362124743685
rat: replaced 0.6272467711312885 by 11338738/18076997 = 0.6272467711312891
rat: replaced 0.6429766032838061 by 10161473/15803799 = 0.6429766032838053
rat: replaced 0.6589261359451484 by 11120191/16876233 = 0.6589261359451484
rat: replaced 0.6750957741586742 by 11234073/16640710 = 0.6750957741586747
rat: replaced 0.6914859009573701 by 9571673/13842181 = 0.6914859009573708
rat: replaced 0.7080968773255479 by 20218829/28553761 = 0.7080968773255474
rat: replaced 0.7249290421627467 by 10945526/15098755 = 0.7249290421627479
rat: replaced 0.7419827122498429 by 23520179/31699093 = 0.7419827122498426
rat: replaced 0.7592581822173726 by 16709871/22008154 = 0.7592581822173727
rat: replaced 9.983250083613754e-5 by 612914/6139423483 = 9.983250083613756e-5
rat: replaced 3.986533601775671e-4 by 220554/553247563 = 3.986533601775666e-4
rat: replaced 8.954327045205754e-4 by 584699/652979277 = 8.954327045205756e-4
rat: replaced 0.001589120864678328 by 740868/466212493 = 0.00158912086467833
rat: replaced 0.002478648480745763 by 878917/354595259 = 0.002478648480745762
rat: replaced 0.003562926609036218 by 2735717/767828614 = 0.003562926609036219
rat: replaced 0.004840846830973591 by 1164348/240525685 = 0.004840846830973582
```

rat: replaced 0.006311281363933816 by 16515210/2616776063 = 0.006311281363933816

rat: replaced 0.007973083174022497 by 2414321/302808957 = 0.007973083174022491

rat: replaced 0.009825086090776508 by 1144049/116441626 = 0.009825086090776506

rat: replaced 0.01186610492378118 by 1659683/139867548 = 0.01186610492378118

rat: replaced 0.01409493558118687 by 986877/70016425 = 0.01409493558118684

rat: replaced 0.01651035519011868 by 1738361/105289134 = 0.01651035519011867

rat: replaced 0.01911112221896202 by 1475047/77182647 = 0.01911112221896199

rat: replaced 0.02189597660151474 by 7711274/352177669 = 0.02189597660151473

rat: replaced 0.02486363986299212 by 3887839/156366446 = 0.02486363986299209

rat: replaced 0.0280128152478745 by 2263313/80795628 = 0.02801281524787455

rat: replaced 0.03134218784958129 by 1116362/35618509 = 0.03134218784958124

rat: replaced 0.03485042474195996 by 3920507/112495243 = 0.03485042474195998

rat: replaced 0.03853617511257795 by 5379408/139593719 = 0.03853617511257795

rat: replaced 0.04239807039780302 by 3385918/79860191 = 0.04239807039780308

rat: replaced 0.04643472441965829 by 10918553/235137672 = 0.04643472441965828

rat: replaced 0.05064473352443885 by 5036501/99447675 = 0.05064473352443886

rat: replaced 0.05502667672307548 by 2932521/53292715 = 0.05502667672307557

rat: replaced 0.05957911583323347 by 6320819/106091185 = 0.05957911583323346  
rat: replaced 0.06430059562312868 by 9893260/153859539 = 0.0643005956231287  
rat: replaced 0.06918964395705007 by 6012189/86894348 = 0.06918964395705  
rat: replaced 0.07424477194257195 by 6096479/82113243 = 0.07424477194257204  
rat: replaced 0.07946447407944118 by 5389689/67825139 = 0.07946447407944125  
rat: replaced 0.0848472284101276 by 9595393/113090235 = 0.08484722841012754  
rat: replaced 0.09039149667201674 by 3773144/41742245 = 0.09039149667201657  
rat: replaced 0.0960957244512361 by 5162056/53717853 = 0.09609572445123597  
rat: replaced 0.1019583413380946 by 1082663/10618680 = 0.1019583413380948  
rat: replaced 0.107977761084122 by 1922059/17800508 = 0.1079777610841219  
rat: replaced 0.1141523817606936 by 5923297/51889386 = 0.1141523817606938  
rat: replaced 0.1204805859192203 by 17634703/146369665 = 0.1204805859192204  
rat: replaced 0.1269607407528933 by 11368220/89541223 = 0.1269607407528932  
rat: replaced 0.1335911982599624 by 4657902/34866833 = 0.1335911982599624  
rat: replaced 0.1403702954085355 by 8528456/60756843 = 0.1403702954085353  
rat: replaced 0.1472963543028805 by 11128453/75551449 = 0.1472963543028804  
rat: replaced 0.1543676823512128 by 8170760/52930509 = 0.1543676823512126  
rat: replaced 0.1615825724349539 by 188109817/1164171446 = 0.1615825724349539

rat: replaced 0.1689393030794406 by 5046974/29874481 = 0.1689393030794409  
rat: replaced 0.1764361386260728 by 6530305/37012287 = 0.176436138626073  
rat: replaced 0.1840713294058766 by 25189859/136848357 = 0.1840713294058766  
rat: replaced 0.1918431119144694 by 24326967/126806570 = 0.1918431119144694  
rat: replaced 0.1997497089884105 by 14902039/74603558 = 0.1997497089884104  
rat: replaced 0.2077893299829148 by 7281351/35041987 = 0.2077893299829145  
rat: replaced 0.2159601709509153 by 11348921/52550991 = 0.2159601709509151  
rat: replaced 0.2242604148234577 by 22385730/99820247 = 0.2242604148234576  
rat: replaced 0.2326882315914051 by 25615030/110083049 = 0.2326882315914051  
rat: replaced 0.2412417784884371 by 14523232/60201977 = 0.2412417784884373  
rat: replaced 0.2499192001753251 by 11309023/45250717 = 0.2499192001753254  
rat: replaced 0.2587186289254649 by 7582961/29309683 = 0.2587186289254647  
rat: replaced 0.267638184811648 by 17912865/66929407 = 0.2676381848116479  
rat: replaced 0.2766759758940514 by 27538925/99534934 = 0.2766759758940514  
rat: replaced 0.2858300984094321 by 29258587/102363562 = 0.2858300984094321  
rat: replaced 0.2950986369614998 by 7877677/26695064 = 0.2950986369614997  
rat: replaced 0.304479664712457 by 14469542/47522195 = 0.304479664712457

rat: replaced 0.3139712435756791 by  $8375733/26676752 = 0.3139712435756797$   
rat: replaced 0.3235714244095225 by  $178371467/551258404 = 0.3235714244095225$   
rat: replaced 0.3332782472122374 by  $5743591/17233621 = 0.333278247212237$   
rat: replaced 0.3430897413179662 by  $15588245/45434891 = 0.3430897413179664$   
rat: replaced 0.3530039255938071 by  $6523425/18479752 = 0.3530039255938067$   
rat: replaced 0.3630188086379282 by  $51253958/141188161 = 0.3630188086379282$   
rat: replaced 0.373132388978704 by  $9370061/25111894 = 0.3731323889787047$   
rat: replaced 0.3833426552748616 by  $11820697/30835851 = 0.3833426552748617$   
rat: replaced 0.393647586516613 by  $9153768/23253713 = 0.3936475865166135$   
rat: replaced 0.4040451522277552 by  $16634707/41170416 = 0.404045152227755$   
rat: replaced 0.4145333126687146 by  $2088920/5039209 = 0.4145333126687145$   
rat: replaced 0.4251100190405208 by  $24667763/58026774 = 0.4251100190405209$   
rat: replaced 0.4357732136896836 by  $10448574/23977091 = 0.435773213689684$   
rat: replaced 0.4465208303139576 by  $8346266/18691773 = 0.4465208303139568$   
rat: replaced 0.4573507941689697 by  $20158688/44077081 = 0.4573507941689696$   
rat: replaced 0.4682610222756929 by  $12818601/27374905 = 0.4682610222756937$   
rat: replaced 0.4792494236287415 by  $13652513/28487281 = 0.4792494236287416$   
rat: replaced 0.4903138994054704 by  $35114711/71616797 = 0.4903138994054705$

rat: replaced 0.5014523431758559 by 15102855/30118226 = 0.5014523431758564  
rat: replaced 0.5126626411131362 by 31697340/61828847 = 0.5126626411131361  
rat: replaced 0.5239426722051925 by 27432767/52358337 = 0.5239426722051924  
rat: replaced 0.5352903084666492 by 6124470/11441399 = 0.5352903084666482  
rat: replaced 0.5467034151516694 by 41717397/76307182 = 0.5467034151516694  
rat: replaced 0.5581798509674292 by 7494380/13426461 = 0.5581798509674292  
rat: replaced 0.5697174682882435 by 14609183/25642856 = 0.5697174682882438  
rat: replaced 0.581314113370329 by 14367580/24715691 = 0.5813141133703282  
rat: replaced 0.5929676265671738 by 9820294/16561265 = 0.5929676265671735  
rat: replaced 0.6046758425455033 by 23593213/39017952 = 0.6046758425455031  
rat: replaced 0.6164365905018095 by 15720181/25501700 = 0.6164365905018097  
rat: replaced 0.6282476943794307 by 53974636/85912987 = 0.6282476943794306  
rat: replaced 0.640106973086155 by 20459615/31962806 = 0.6401069730861552  
rat: replaced 0.652012240712328 by 51645100/79208789 = 0.652012240712328  
rat: replaced 0.6639613067494411 by 12215999/18398661 = 0.6639613067494422  
rat: replaced 0.6759519763091814 by 18558734/27455699 = 0.6759519763091808  
rat: replaced 0.6879820503429186 by 23500536/34158647 = 0.687982050342919

```

rat: replaced 0.7000493258616074 by 29992669/42843651 = 0.7000493258616078

rat: replaced 0.7121515961560857 by 10685401/15004391 = 0.7121515961560853

rat: replaced 0.7242866510177421 by 11795807/16286103 = 0.7242866510177419

rat: replaced 0.7364522769595366 by 14940657/20287339 = 0.7364522769595362

rat: replaced 0.7486462574373463 by 42508133/56779998 = 0.7486462574373461
part: invalid index of list or matrix.
#0: lineIntersection(g=[2,1,u],h=[-1,2,1])
#1: projectToLine(a=[0,u],g=[-1,2,1])
-- an error. To debug this try: debugmode(true);

Error in:
d &= distance(P,projectToLine(P,g));
^

```

Dan temukan pusat kedua lingkaran yang jaraknya sama.

```
>sol &= solve(d1^2=d^2,u); $sol
```

Ada dua solusi.

Kami mengevaluasi solusi simbolis, dan menemukan kedua pusat, dan kedua jarak.

```
>u := sol()
```

```
[]
```

```
>dd := d()
```

```
Function d needs at least one argument!
Use: d (n)
Error in:
dd := d() ...  
^
```

Plot lingkaran ke dalam gambar.

```
>color(red);
>plotCircle(circleWithCenter([0,u[1]],dd[1]), "");
```

```
Index 1 out of bounds!
Error in:
plotCircle(circleWithCenter([0,u[1]],dd[1]), ""); ...  
^
```

```
>plotCircle(circleWithCenter([0,u[2]],dd[2]), "");
```

```
Index 2 out of bounds!
Error in:
plotCircle(circleWithCenter([0,u[2]],dd[2]), ""); ...  
^
```

```
>insimg;
```

## Plot dengan Povray

---

Selanjutnya kami merencanakan semuanya dengan Povray. Perhatikan bahwa Anda mengubah perintah apa pun dalam urutan perintah Povray berikut, dan menjalankan kembali semua perintah dengan Shift-Return.

Pertama kita memuat fungsi povray.

```
>load povray;
>defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe

Kami mengatur adegan dengan tepat.

```
>povstart(zoom=11,center=[0,0,0.5],height=10°,angle=140°);
```

Selanjutnya kita menulis dua bidang ke file Povray.

```
>writeln(povsphere([0,0,u[1]],dd[1],povlook(red)));
```

```
Index 1 out of bounds!
Error in:
writeln(povsphere([0,0,u[1]],dd[1],povlook(red))); ...  
^
```

```
>writeln(povsphere([0,0,u[2]],dd[2],povlook(red)));
```

```
Index 2 out of bounds!
Error in:
writeln(povsphere([0,0,u[2]],dd[2],povlook(red))); ...  
^
```

Dan kerucutnya, transparan.

```
>writeln(povcone([0,0,0],0,[0,0,a],1,povlook(lightgray,1)));
```

Kami menghasilkan bidang terbatas pada kerucut.

```
>gp=g();
>pc=povcone([0,0,0],0,[0,0,a],1,"");
>vp=[gp[1],0,gp[2]]; dp=gp[3];
>writeln(povplane(vp,dp,povlook(blue,0.5),pc));
```

Sekarang kita menghasilkan dua titik pada lingkaran, di mana bola menyentuh kerucut.

```
>function turnz(v) := return [-v[2],v[1],v[3]]
>P1=projectToLine([0,u[1]],g1()); P1=turnz([P1[1],0,P1[2]]);
```

```
Index 1 out of bounds!
Error in:
P1=projectToLine([0,u[1]],g1()); P1=turnz([P1[1],0,P1[2]]); ...
```

```
>writeln(povpoint(P1,povlook(yellow)));
```

```
Function povpoint needs a vector for P
Error in:
writeln(povpoint(P1,povlook(yellow))); ...
```

```
>P2=projectToLine([0,u[2]],g1()); P2=turnz([P2[1],0,P2[2]]);
```

```
Index 2 out of bounds!
Error in:
P2=projectToLine([0,u[2]],g1()); P2=turnz([P2[1],0,P2[2]]); ...
```

```
>writeln(povpoint(P2,povlook(yellow)));
```

```
Function povpoint needs a vector for P
Error in:
writeln(povpoint(P2,povlook(yellow))); ...
```

Kemudian kami menghasilkan dua titik di mana bola menyentuh bidang. Ini adalah fokus dari ellips.

```
>P3=projectToLine([0,u[1]],g()); P3=[P3[1],0,P3[2]];
```

```
Index 1 out of bounds!
Error in:
P3=projectToLine([0,u[1]],g()); P3=[P3[1],0,P3[2]]; ...
```

```
>writeln(povpoint(P3,povlook(yellow)));
```

```
Variable or function P3 not found.  
Error in:  
writeln(povpoint(P3,povlook(yellow))); ...  
^
```

```
>P4=projectToLine([0,u[2]],g()); P4=[P4[1],0,P4[2]];
```

```
Index 2 out of bounds!  
Error in:  
P4=projectToLine([0,u[2]],g()); P4=[P4[1],0,P4[2]]; ...  
^
```

```
>writeln(povpoint(P4,povlook(yellow)));
```

```
Variable or function P4 not found.  
Error in:  
writeln(povpoint(P4,povlook(yellow))); ...  
^
```

Selanjutnya kita hitung perpotongan P1P2 dengan bidang.

```
>t1=scalp(vp,P1)-dp; t2=scalp(vp,P2)-dp; P5=P1+t1/(t1-t2)*(P2-P1);
```

```
Matrix expected in scalp!
Error in:
t1=scalp(vp,P1)-dp; t2=scalp(vp,P2)-dp; P5=P1+t1/(t1-t2)*(P2-P ...
^
```

```
>writeln(povpoint(P5,povlook(yellow)));
```

```
Variable or function P5 not found.
Error in:
writeln(povpoint(P5,povlook(yellow))); ...
^
```

Kami menghubungkan titik-titik dengan segmen garis.

```
>writeln(povsegment(P1,P2,povlook(yellow)));
```

```
Function povsegment needs a vector for P1
Error in:
writeln(povsegment(P1,P2,povlook(yellow))); ...
^
```

```
>writeln(povsegment(P5,P3,povlook(yellow)));
```

```
Variable or function P5 not found.  
Error in:  
writeln(povsegment(P5,P3,povlook(yellow))); ...  
^
```

```
>writeln(povsegment(P5,P4,povlook(yellow)));
```

```
Variable or function P5 not found.  
Error in:  
writeln(povsegment(P5,P4,povlook(yellow))); ...  
^
```

Sekarang kita menghasilkan pita abu-abu, di mana bola menyentuh kerucut.

```
>pcw=povcone([0,0,0],0,[0,0,a],1.01);  
>pc1=povcylinder([0,0,P1[3]-defaultpointsize/2],[0,0,P1[3]+defaultpointsize/2],1);
```

```
Index 3 out of range for string (need string vector).  
Error in:  
pc1=povcylinder([0,0,P1[3]-defaultpointsize/2],[0,0,P1[3]+defa ...  
^
```

```
>writeln(povintersection([pcw,pc1],povlook(gray)));
```

```
Variable pc1 not found!
Error in:
writeln(povintersection([pcw,pc1],povlook(gray))); ...
```

```
>pc2=povcylinder([0,0,P2[3]-defaultpointsize/2],[0,0,P2[3]+defaultpointsize/2],1);
```

```
Index 3 out of range for string (need string vector).
Error in:
pc2=povcylinder([0,0,P2[3]-defaultpointsize/2],[0,0,P2[3]+defa ...
```

```
>writeln(povintersection([pcw,pc2],povlook(gray)));
```

```
Variable pc2 not found!
Error in:
writeln(povintersection([pcw,pc2],povlook(gray))); ...
```

Mulai program Povray.

```
>povend();
```

Untuk mendapatkan Anaglyph ini kita perlu memasukkan semuanya ke dalam fungsi scene. Fungsi ini akan digunakan dua kali kemudian.

```
>function scene () ...  
  
    global a,u,dd,g,g1,defaultpointsize;  
    writeln(povsphere([0,0,u[1]],dd[1],povlook(red)));  
    writeln(povsphere([0,0,u[2]],dd[2],povlook(red)));  
    writeln(povcone([0,0,0],0,[0,0,a],1,povlook(lightgray,1)));  
    gp=g();  
    pc=povcone([0,0,0],0,[0,0,a],1,"");  
    vp=[gp[1],0,gp[2]]; dp=gp[3];  
    writeln(povplane(vp,dp,povlook(blue,0.5),pc));  
    P1=projectToLine([0,u[1]],g1()); P1=turnz([P1[1],0,P1[2]]);  
    writeln(povpoint(P1,povlook(yellow)));  
    P2=projectToLine([0,u[2]],g1()); P2=turnz([P2[1],0,P2[2]]);  
    writeln(povpoint(P2,povlook(yellow)));  
    P3=projectToLine([0,u[1]],g()); P3=[P3[1],0,P3[2]];  
    writeln(povpoint(P3,povlook(yellow)));  
    P4=projectToLine([0,u[2]],g()); P4=[P4[1],0,P4[2]];  
    writeln(povpoint(P4,povlook(yellow)));  
    t1=scalp(vp,P1)-dp; t2=scalp(vp,P2)-dp; P5=P1+t1/(t1-t2)*(P2-P1);  
    writeln(povpoint(P5,povlook(yellow)));  
    writeln(povsegment(P1,P2,povlook(yellow)));  
    writeln(povsegment(P5,P3,povlook(yellow)));  
    writeln(povsegment(P5,P4,povlook(yellow)));  
    pcw=povcone([0,0,0],0,[0,0,a],1.01);  
    pc1=povcylinder([0,0,P1[3]-defaultpointsize/2],[0,0,P1[3]+defaultpointsize/2],1);  
    writeln(povintersection([pcw,pc1],povlook(gray)));  
    pc2=povcylinder([0,0,P2[3]-defaultpointsize/2],[0,0,P2[3]+defaultpointsize/2],1);  
    writeln(povintersection([pcw,pc2],povlook(gray)));  
endfunction
```

Anda membutuhkan kacamata merah/sian untuk menghargai efek berikut.

```
>povanaglyph("scene",zoom=11,center=[0,0,0.5],height=10°,angle=140°);
```

```
Global variable dd not found in "global" command.  
scene:  
    global a,u,dd,g,g1,defaultpointsize;  
Try "trace errors" to inspect local variables after errors.  
povanaglyph:  
    scene$(args());
```

### Contoh 8: Geometri Bumi

---

Dalam buku catatan ini, kami ingin melakukan beberapa perhitungan sferis. Fungsi-fungsi tersebut terdapat dalam file "spherical.e" di folder contoh. Kita perlu memuat file itu terlebih dahulu.

```
>load "spherical.e";
```

Untuk memasukkan posisi geografis, kami menggunakan vektor dengan dua koordinat dalam radian (utara dan timur, nilai negatif untuk selatan dan barat). Berikut koordinat Kampus FMIPA UNY.

```
>FMIPA=[rad(-7,-46.467),rad(110,23.05)]
```

```
[-0.13569, 1.92657]
```

Anda dapat mencetak posisi ini dengan sposprint (cetak posisi spherical).

```
>sposprint(FMIPA) // posisi garis lintang dan garis bujur FMIPA UNY
```

```
S 7°46.467' E 110°23.050'
```

Mari kita tambahkan dua kota lagi, Solo dan Semarang.

```
>Solo=[rad(-7,-34.333),rad(110,49.683)]; Semarang=[rad(-6,-59.05),rad(110,24.533)];  
>sposprint(Solo), sposprint(Semarang),
```

```
S 7°34.333' E 110°49.683'  
S 6°59.050' E 110°24.533'
```

Pertama kita menghitung vektor dari satu ke yang lain pada bola ideal. Vektor ini [pos,jarak] dalam radian. Untuk menghitung jarak di bumi, kita kalikan dengan jari-jari bumi pada garis lintang  $7^\circ$ .

```
>br=svector(FMIPA,Solo); degprint(br[1]), br[2]*rearth(7°)->km // perkiraan jarak FMIPA-Solo
```

```
65°20'26.60''  
53.8945384608
```

Ini adalah perkiraan yang baik. Rutinitas berikut menggunakan perkiraan yang lebih baik. Pada jarak yang begitu pendek hasilnya hampir sama.

```
>esdist(FMIPA, Semarang)->" km", // perkiraan jarak FMIPA-Semarang
```

```
88.0114026318 km
```

Ada fungsi untuk heading, dengan mempertimbangkan bentuk elips bumi. Sekali lagi, kami mencetak dengan cara yang canggih.

```
>sdegprint(esdir(FMIPA,Solo))
```

```
65.34°
```

Sudut segitiga melebihi  $180^\circ$  pada bola.

```
>asum=sangle(Solo,FMIPA,Semarang)+sangle(FMIPA,Solo,Semarang)+sangle(FMIPA,Semarang,Solo); degprint(
```

$180^\circ 0' 10.77''$

Ini dapat digunakan untuk menghitung luas segitiga. Catatan: Untuk segitiga kecil, ini tidak akurat karena kesalahan pengurangan dalam asum-pi.

```
>(asum-pi)*rearth( $48^\circ$ )^2->" km^2", // perkiraan luas segitiga FMIPA-Solo-Semarang
```

$2116.02948749 \text{ km}^2$

Ada fungsi untuk ini, yang menggunakan garis lintang rata-rata segitiga untuk menghitung jari-jari bumi, dan menangani kesalahan pembulatan untuk segitiga yang sangat kecil.

```
>esarea(Solo,FMIPA,Semarang)->" km^2", //perkiraan yang sama dengan fungsi esarea()
```

$2123.64310526 \text{ km}^2$

Kita juga dapat menambahkan vektor ke posisi. Sebuah vektor berisi heading dan jarak, keduanya dalam radian. Untuk mendapatkan vektor, kami menggunakan vektor. Untuk menambahkan vektor ke posisi, kami menggunakan vektor saddr.

```
>v=svector(FMIPA,Solo); sposprint(saddrvector(FMIPA,v)), sposprint(Solo),
```

```
S 7°34.333' E 110°49.683'  
S 7°34.333' E 110°49.683'
```

Fungsi-fungsi ini mengasumsikan bola yang ideal. Hal yang sama di bumi.

```
>sposprint(esaddr(FMIPA,esdir(FMIPA,Solo),esdist(FMIPA,Solo))), sposprint(Solo),
```

```
S 7°34.333' E 110°49.683'  
S 7°34.333' E 110°49.683'
```

Mari kita beralih ke contoh yang lebih besar, Tugu Jogja dan Monas Jakarta (menggunakan Google Earth untuk mencari koordinatnya).

```
>Tugu=[-7.7833°,110.3661°]; Monas=[-6.175°,106.811944°];  
>sposprint(Tugu), sposprint(Monas)
```

```
S 7°46.998' E 110°21.966'  
S 6°10.500' E 106°48.717'
```

Menurut Google Earth, jaraknya adalah 429,66 km. Kami mendapatkan pendekatan yang baik.

```
>esdist(Tugu,Monas)->" km", // perkiraan jarak Tugu Jogja - Monas Jakarta
```

431.565659488 km

Judulnya sama dengan judul yang dihitung di Google Earth.

```
>degprint(esdir(Tugu,Monas))
```

294°17'2.85'',

Namun, kita tidak lagi mendapatkan posisi target yang tepat, jika kita menambahkan heading dan jarak ke posisi semula. Hal ini terjadi, karena kita tidak menghitung fungsi invers secara tepat, tetapi mengambil perkiraan jari-jari bumi di sepanjang jalan.

```
>sposprint(esadd(Tugu,esdir(Tugu,Monas),esdist(Tugu,Monas)))
```

S 6°10.500' E 106°48.717'

Namun, kesalahannya tidak besar.

```
>sposprint(Monas),
```

S 6°10.500' E 106°48.717'

Tentu kita tidak bisa berlayar dengan tujuan yang sama dari satu tujuan ke tujuan lainnya, jika kita ingin menempuh jalur terpendek. Bayangkan, Anda terbang NE mulai dari titik mana pun di bumi. Kemudian Anda akan berputar ke kutub utara. Lingkaran besar tidak mengikuti heading yang konstan! Perhitungan berikut menunjukkan bahwa kami jauh dari tujuan yang benar, jika kami menggunakan pos yang sama selama perjalanan kami.

```
>dist=esdist(Tugu,Monas); hd=esdir(Tugu,Monas);
```

Sekarang kita tambahkan 10 kali sepersepuluh dari jarak, menggunakan pos ke Monas, kita sampai di Tugu.

```
>p=Tugu; loop 1 to 10; p=esadd(p,hd,dist/10); end;
```

Hasilnya jauh.

```
>sposprint(p), skmpprint(esdist(p,Monas))
```

S 6°11.250' E 106°48.372'  
1.529km

Sebagai contoh lain, mari kita ambil dua titik di bumi pada garis lintang yang sama.

```
>P1=[30°,10°]; P2=[30°,50°];
```

Jalur terpendek dari P1 ke P2 bukanlah lingkaran garis lintang 30°, melainkan jalur terpendek yang dimulai 10° lebih jauh ke utara di P1.

```
>sdegprint(esdir(P1,P2))
```

79.69°

Tapi, jika kita mengikuti pembacaan kompas ini, kita akan berputar ke kutub utara! Jadi kita harus menyesuaikan arah kita di sepanjang jalan. Untuk tujuan kasar, kami menyesuaikannya pada 1/10 dari total jarak.

```
>p=P1; dist=esdist(P1,P2); ...
> loop 1 to 10; dir=esdir(p,P2); sdegprint(dir), p=esadd(p,dir,dist/10); end;
```

79.69°  
81.67°  
83.71°  
85.78°  
87.89°  
90.00°  
92.12°  
94.22°  
96.29°  
98.33°

Jaraknya tidak tepat, karena kita akan menambahkan sedikit kesalahan, jika kita mengikuti heading yang sama terlalu lama.

```
>skmprint(esdist(p,P2))
```

0.203km

Kami mendapatkan perkiraan yang baik, jika kami menyesuaikan pos setelah setiap 1/100 dari total jarak dari Tugu ke Monas.

```
>p=Tugu; dist=esdist(Tugu,Monas); ...
> loop 1 to 100; p=esadd(p,esdir(p,Monas),dist/100); end;
>skmprint(esdist(p,Monas))
```

0.000km

Untuk keperluan navigasi, kita bisa mendapatkan urutan posisi GPS di sepanjang lingkaran besar menuju Monas dengan fungsi navigasi.

```
>load spherical; v=navigate(Tugu,Monas,10); ...
> loop 1 to rows(v); sposprint(v[#]), end;
```

S 7°46.998' E 110°21.966'  
S 7°37.422' E 110°0.573'  
S 7°27.829' E 109°39.196'  
S 7°18.219' E 109°17.834'  
S 7°8.592' E 108°56.488'  
S 6°58.948' E 108°35.157'  
S 6°49.289' E 108°13.841'  
S 6°39.614' E 107°52.539'  
S 6°29.924' E 107°31.251'  
S 6°20.219' E 107°9.977'  
S 6°10.500' E 106°48.717'

Kami menulis sebuah fungsi, yang memplot bumi, dua posisi, dan posisi di antaranya.

```
>function testplot ...
useglobal;
plotearth;
plotpos(Tugu,"Tugu Jogja"); plotpos(Monas,"Tugu Monas");
plotposline(v);
endfunction
```

Sekarang rencanakan semuanya.

```
>plot3d("testplot",angle=25, height=6,>own,>user,zoom=4):
```

Atau gunakan plot3d untuk mendapatkan tampilan anaglyph. Ini terlihat sangat bagus dengan kacamata merah/sian.

```
>plot3d("testplot",angle=25,height=6,distance=5,own=1,anaglyph=1,zoom=4):
```

---

**MENCOBA RUMUS-RUMUS PADA MATERI DI ATAS**

```
>A &= [2,0]; B &= [0,2]; C &= [3,3]; // menentukan tiga titik A, B, C  
>c &= lineThrough(B,C) // c=BC
```

```
[- 1, 3, 6]
```

```
>$getLineEquation(c,x,y), $solve(%,y) | expand // persamaan garis c
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);
```

```
Error in:  
$getLineEquation(c,x,y), $solve(%,y) | expand // persamaan gar ...  
^
```

```
>h &= perpendicular(A,lineThrough(B,C)) // h melalui A tegak lurus BC
```

```
[3, 1, 6]
```

```
>Q &= lineIntersection(c,h) // Q titik potong garis c=BC dan h
```

```
Maxima said:  
rat: replaced 1.498320841708742e-4 by 1329822/8875415485 = 1.498320841708742e-4  
  
rat: replaced 5.986466935998108e-4 by 398723/666040595 = 5.986466935998098e-4  
  
rat: replaced 0.001345398955533032 by 4525441/3363642421 = 0.001345398955533032  
  
rat: replaced 0.002389014203700413 by 1071627/448564516 = 0.00238901420370041  
  
rat: replaced 0.00372838808577948 by 661903/177530607 = 0.003728388085779485  
  
rat: replaced 0.005362386673832029 by 5230891/975478144 = 0.005362386673832028  
  
rat: replaced 0.007289846577694006 by 32241346/4422774287 = 0.007289846577694006  
  
rat: replaced 0.009509575061314376 by 2146493/225719129 = 0.009509575061314364  
  
rat: replaced 0.01202035016202822 by 1789188/148846579 = 0.01202035016202825  
  
rat: replaced 0.01482092081275069 by 2581665/174190594 = 0.01482092081275066  
  
rat: replaced 0.01791000696708436 by 5107285/285163764 = 0.01791000696708436  
  
rat: replaced 0.02128629972732062 by 3323295/156123659 = 0.02128629972732064  
  
rat: replaced 0.02494846147533059 by 4548287/182307314 = 0.02494846147533061  
  
rat: replaced 0.02889512600632479 by 3147802/108938857 = 0.02889512600632481  
  
rat: replaced 0.0331248986654725 by 5858625/176864692 = 0.03312489866547248
```

rat: replaced 0.03763635648736519 by  $10043830/266865099 = 0.03763635648736518$   
rat: replaced 0.04242804833831373 by  $4635713/109260576 = 0.04242804833831372$   
rat: replaced 0.04749849506145984 by  $5610259/118114458 = 0.04749849506145979$   
rat: replaced 0.05284618962468965 by  $4237503/80185592 = 0.05284618962468968$   
rat: replaced 0.05846959727133633 by  $3317197/56733707 = 0.05846959727133642$   
rat: replaced 0.06436715567365475 by  $13427433/208606903 = 0.06436715567365477$   
rat: replaced 0.07053727508905278 by  $8025659/113778977 = 0.07053727508905269$   
rat: replaced 0.07697833851906408 by  $6306881/81930594 = 0.07697833851906408$   
rat: replaced 0.08368870187104593 by  $4282086/51166835 = 0.08368870187104596$   
rat: replaced 0.09066669412258874 by  $2175091/23989967 = 0.09066669412258883$   
rat: replaced 0.09791061748861546 by  $8290049/84669561 = 0.09791061748861554$   
rat: replaced 0.1054187475911595 by  $8501563/80645646 = 0.1054187475911595$   
rat: replaced 0.1131893336318011 by  $6539019/57770629 = 0.113189333631801$   
rat: replaced 0.121220598566744 by  $5779101/47674249 = 0.1212205985667441$   
rat: replaced 0.1295107392845216 by  $7134865/55090914 = 0.1295107392845216$   
rat: replaced 0.1380579267863034 by  $6113057/44278928 = 0.1380579267863034$   
rat: replaced 0.1468603063687953 by  $6311140/42973763 = 0.1468603063687953$   
rat: replaced 0.1559159978097077 by  $4027079/25828517 = 0.1559159978097078$

```
rat: replaced 0.1652230955557758 by 10597125/64138279 = 0.1652230955557757
rat: replaced 0.1747796689133147 by 9649007/55206690 = 0.1747796689133147
rat: replaced 0.1845837622412855 by 6871913/37229239 = 0.1845837622412857
rat: replaced 0.1946333951468589 by 39341769/202132676 = 0.1946333951468589
rat: replaced 0.2049265626834523 by 10758647/52500012 = 0.2049265626834523
rat: replaced 0.2154612355512225 by 33702610/156420759 = 0.2154612355512225
rat: replaced 0.2262353602999955 by 2338161/10335082 = 0.2262353602999957
rat: replaced 0.2372468595346078 by 7573078/31920667 = 0.2372468595346081
rat: replaced 0.2484936321226457 by 3764353/15148690 = 0.2484936321226456
rat: replaced 0.2599735534045555 by 26335713/101301508 = 0.2599735534045554
rat: replaced 0.2716844754061095 by 29831699/109802737 = 0.2716844754061094
rat: replaced 0.2836242270531998 by 15100773/53242183 = 0.2836242270531995
rat: replaced 0.2957906143889442 by 2942977/9949528 = 0.2957906143889439
rat: replaced 0.3081814207930817 by 12077608/39189929 = 0.3081814207930818
rat: replaced 0.3207944072036307 by 9185023/28632117 = 0.3207944072036308
rat: replaced 0.333627312340794 by 5228336/15671187 = 0.3336273123407946
rat: replaced 0.346677852933085 by 15615111/45042136 = 0.3466778529330847
```

rat: replaced 0.3599437239456539 by  $7564465/21015688 = 0.3599437239456543$   
rat: replaced 0.3734225988107874 by  $7702871/20627758 = 0.3734225988107869$   
rat: replaced 0.3871121296605642 by  $97723109/252441351 = 0.3871121296605642$   
rat: replaced 0.4010099475616409 by  $3146543/7846546 = 0.4010099475616405$   
rat: replaced 0.4151136627521425 by  $6219049/14981557 = 0.4151136627521425$   
rat: replaced 0.4294208648806354 by  $26148647/60892819 = 0.4294208648806356$   
rat: replaced 0.4439291232471635 by  $19525684/43983787 = 0.4439291232471638$   
rat: replaced 0.458635987046313 by  $38604672/84172793 = 0.4586359870463132$   
rat: replaced 0.4735389856122937 by  $11146199/23538081 = 0.4735389856122935$   
rat: replaced 0.488635628666001 by  $13946471/28541658 = 0.4886356286660011$   
rat: replaced 0.5039234065640431 by  $5948069/11803518 = 0.503923406564043$   
rat: replaced 0.5193997905497036 by  $24027011/46259185 = 0.5193997905497038$   
rat: replaced 0.5350622330058146 by  $7363779/13762472 = 0.5350622330058147$   
rat: replaced 0.5509081677095147 by  $8130825/14758948 = 0.5509081677095142$   
rat: replaced 0.5669350100888726 by  $10250363/18080314 = 0.5669350100888735$   
rat: replaced 0.5831401574813392 by  $37655026/64572857 = 0.5831401574813393$   
rat: replaced 0.5995209893940125 by  $30778651/51338738 = 0.5995209893940128$   
rat: replaced 0.6160748677656853 by  $23698401/38466755 = 0.616074867765685$

rat: replaced 0.6327991372306488 by 5052598/7984521 = 0.6327991372306492  
rat: replaced 0.6496911253842265 by 60646047/93345968 = 0.6496911253842266  
rat: replaced 0.666748143050013 by 30125566/45182827 = 0.6667481430500132  
rat: replaced 0.6839674845487889 by 8953739/13090884 = 0.6839674845487899  
rat: replaced 0.7013464279690875 by 7888577/11247761 = 0.7013464279690864  
rat: replaced 0.7188822354393821 by 16662338/23178119 = 0.7188822354393815  
rat: replaced 0.7365721534018723 by 13899283/18870226 = 0.7365721534018723  
rat: replaced 0.7544134128878366 by 16270763/21567436 = 0.754413412887837  
rat: replaced 0.7724032297945274 by 8203205/10620366 = 0.7724032297945287  
rat: replaced 0.7905388051635788 by 10794522/13654639 = 0.7905388051635784  
rat: replaced 0.8088173254609005 by 16745047/20703126 = 0.808817325460899  
rat: replaced 0.8272359628580275 by 20291194/24528907 = 0.827235962858027  
rat: replaced 0.8457918755149025 by 10996366/13001267 = 0.8457918755149018  
rat: replaced 0.8644822078640563 by 9158500/10594203 = 0.8644822078640555  
rat: replaced 0.8833040908961625 by 13759446/15577247 = 0.8833040908961641  
rat: replaced 0.9022546424469358 by 19827819/21975857 = 0.9022546424469362  
rat: replaced 0.9213309674853474 by 60458149/65620446 = 0.9213309674853475

```
rat: replaced 0.9405301584031224 by 11658841/12396031 = 0.9405301584031212
rat: replaced 0.9598492953055026 by 26214088/27310629 = 0.9598492953055018
rat: replaced 0.9792854463032298 by 35089005/35831233 = 0.9792854463032293
rat: replaced 0.9988356678057343 by 15735752/15754095 = 0.9988356678057356
rat: replaced 1.018497004815491 by 16202286/15908035 = 1.018497004815491
rat: replaced 1.038266491223517 by 17763365/17108676 = 1.038266491223517
rat: replaced 1.058141150105979 by 33730321/31876958 = 1.058141150105979
rat: replaced 1.078117994021884 by 51996446/48228901 = 1.078117994021883
rat: replaced 1.098194025311821 by 124719922/113568203 = 1.098194025311821
rat: replaced 1.118366236397724 by 92837336/83011569 = 1.118366236397724
rat: replaced 1.13863161008363 by 20601995/18093644 = 1.138631610083629
rat: replaced 1.158987119857388 by 20626233/17796775 = 1.15898711985739
rat: replaced 1.17942973019332 by 4098089/3474636 = 1.179429730193321
rat: replaced 1.199956396855759 by 17442145/14535649 = 1.199956396855758
rat: replaced 5.049958083474387e-5 by 102157/2022927682 = 5.049958083474385e-5
rat: replaced 2.039932534230044e-4 by 284619/1395237319 = 2.039932534230043e-4
rat: replaced 4.634656435254722e-4 by 573493/1237401322 = 4.634656435254721e-4
rat: replaced 8.31890779119604e-4 by 332331/399488741 = 8.318907791196046e-4
```

rat: replaced 0.001312231792998733 by 448125/341498356 = 0.001312231792998734  
rat: replaced 0.001907440626462018 by 276030/144712237 = 0.001907440626462018  
rat: replaced 0.002620457734122131 by 2586613/987084419 = 0.002620457734122131  
rat: replaced 0.00345421178986248 by 3402379/984994322 = 0.00345421178986248  
rat: replaced 0.004411619393972596 by 966955/219183686 = 0.004411619393972597  
rat: replaced 0.005495584781489732 by 2798484/509224061 = 0.005495584781489734  
rat: replaced 0.006708999531778753 by 6054060/902378957 = 0.006708999531778753  
rat: replaced 0.008054742279375651 by 806546/100133061 = 0.00805474227937564  
rat: replaced 0.009535678426127348 by 4115324/431571181 = 0.009535678426127346  
rat: replaced 0.01115465985465333 by 2266398/203179481 = 0.01115465985465334  
rat: replaced 0.01291452464316009 by 2106925/163143829 = 0.01291452464316012  
rat: replaced 0.01481809678163515 by 2779203/187554653 = 0.01481809678163516  
rat: replaced 0.01686818588945119 by 7427428/440321683 = 0.01686818588945119  
rat: replaced 0.01906758693440599 by 2278085/119474216 = 0.019067586934406  
rat: replaced 0.02141907995322798 by 2316386/108145915 = 0.02141907995322801  
rat: replaced 0.02392542977357476 by 1665518/69612877 = 0.02392542977357479  
rat: replaced 0.02658938573755304 by 3678645/138350131 = 0.02658938573755308

rat: replaced 0.02941368142678652 by  $4053557/137811957 = 0.0294136814267865$   
rat: replaced 0.03240103438906003 by  $2629160/81144323 = 0.03240103438906009$   
rat: replaced 0.03555414586656669 by  $1834427/51595305 = 0.03555414586656674$   
rat: replaced 0.03887570052578646 by  $1643964/42287701 = 0.03887570052578645$   
rat: replaced 0.04236836618902146 by  $3055464/72116635 = 0.04236836618902143$   
rat: replaced 0.04603479356761608 by  $3139251/68193007 = 0.0460347935676161$   
rat: replaced 0.04987761599688789 by  $5203437/104324092 = 0.04987761599688785$   
rat: replaced 0.05389944917279615 by  $4533622/84112585 = 0.0538994491727962$   
rat: replaced 0.05810289089037535 by  $11687290/201148167 = 0.05810289089037535$   
rat: replaced 0.06249052078395612 by  $3949243/63197473 = 0.06249052078395603$   
rat: replaced 0.0670649000692059 by  $3281728/48933615 = 0.067064900069206$   
rat: replaced 0.07182857128700804 by  $4146139/57722699 = 0.07182857128700791$   
rat: replaced 0.07678405804921068 by  $1198255/15605518 = 0.07678405804921054$   
rat: replaced 0.08193386478626702 by  $5956639/72700574 = 0.08193386478626702$   
rat: replaced 0.08728047649679532 by  $2799808/32078285 = 0.08728047649679527$   
rat: replaced 0.09282635849907966 by  $10292829/110882611 = 0.09282635849907972$   
rat: replaced 0.09857395618454184 by  $4198057/42587892 = 0.09857395618454184$   
rat: replaced 0.1045256947732028 by  $30563827/292404916 = 0.1045256947732028$

```
rat: replaced 0.1106839790711635 by 8949559/80856860 = 0.1106839790711635
rat: replaced 0.1170511932301264 by 9911603/84677505 = 0.1170511932301265
rat: replaced 0.1236297005089814 by 19561703/158228184 = 0.1236297005089814
rat: replaced 0.1304218430374826 by 4975231/38147222 = 0.1304218430374825
rat: replaced 0.137429941582038 by 3502939/25488907 = 0.137429941582038
rat: replaced 0.1446562953136327 by 15521432/107298697 = 0.1446562953136327
rat: replaced 0.1521031815779155 by 18080502/118869979 = 0.1521031815779155
rat: replaced 0.1597728556674664 by 37419026/234201397 = 0.1597728556674664
rat: replaced 0.1676675505962674 by 22585897/134706429 = 0.1676675505962674
rat: replaced 0.1757894768764047 by 15940893/90681725 = 0.1757894768764048
rat: replaced 0.1841408222970185 by 7944795/43145213 = 0.1841408222970182
rat: replaced 0.1927237517055264 by 1392861/7227241 = 0.1927237517055264
rat: replaced 0.2015404067911402 by 1735485/8611102 = 0.2015404067911401
rat: replaced 0.2105929058706983 by 10627754/50465869 = 0.2105929058706985
rat: replaced 0.2198833436768368 by 9372347/42624179 = 0.2198833436768366
rat: replaced 0.2294137911485169 by 7405273/32279110 = 0.2294137911485168
rat: replaced 0.239186295223934 by 27692337/115777273 = 0.239186295223934
```

rat: replaced 0.2492028786358237 by 8925310/35815437 = 0.249202878635824  
rat: replaced 0.2594655397091927 by 11150701/42975653 = 0.259465539709193  
rat: replaced 0.2699762521614856 by 11249087/41666950 = 0.2699762521614853  
rat: replaced 0.280736964905216 by 12097010/43090193 = 0.2807369649052164  
rat: replaced 0.2917496018530771 by 14831788/50837389 = 0.2917496018530771  
rat: replaced 0.3030160617255513 by 18597622/61375037 = 0.3030160617255514  
rat: replaced 0.3145382178610399 by 11102944/35299189 = 0.3145382178610392  
rat: replaced 0.3263179180285316 by 13053510/40002431 = 0.3263179180285318  
rat: replaced 0.3383569842428258 by 13796661/40775458 = 0.3383569842428257  
rat: replaced 0.3506572125823338 by 33496033/95523582 = 0.3506572125823338  
rat: replaced 0.3632203730094723 by 23086207/63559780 = 0.3632203730094724  
rat: replaced 0.376048209193667 by 22674222/60296051 = 0.3760482091936668  
rat: replaced 0.3891424383369902 by 33246815/85436107 = 0.3891424383369902  
rat: replaced 0.402504751002439 by 7793813/19363282 = 0.4025047510024385  
rat: replaced 0.4161368109448825 by 10481453/25187517 = 0.4161368109448819  
rat: replaced 0.4300402549446862 by 19565443/45496771 = 0.4300402549446861  
rat: replaced 0.4442166926440365 by 16102633/36249500 = 0.4442166926440365  
rat: replaced 0.4586677063859775 by 19404529/42306290 = 0.4586677063859771

rat: replaced 0.4733948510561774 by 10262860/21679281 = 0.4733948510561766  
rat: replaced 0.4883996539274416 by 4159841/8517289 = 0.488399653927441  
rat: replaced 0.5036836145069872 by 13202363/26211619 = 0.5036836145069864  
rat: replaced 0.5192482043864929 by 12221370/23536663 = 0.5192482043864927  
rat: replaced 0.5350948670949413 by 52965833/98984005 = 0.5350948670949413  
rat: replaced 0.551225017954267 by 14288533/25921416 = 0.551225017954266  
rat: replaced 0.5676400439378262 by 25565995/45039097 = 0.5676400439378259  
rat: replaced 0.5843413035316997 by 18888222/32323955 = 0.5843413035316997  
rat: replaced 0.6013301265988455 by 5789399/9627655 = 0.6013301265988447  
rat: replaced 0.6186078142461149 by 4803773/7765458 = 0.618607814246114  
rat: replaced 0.6361756386941407 by 13914515/21872128 = 0.6361756386941407  
rat: replaced 0.6540348431501183 by 48160581/73636109 = 0.6540348431501181  
rat: replaced 0.6721866416834846 by 6617334/9844489 = 0.6721866416834841  
rat: replaced 0.690632219104513 by 16840135/24383651 = 0.6906322191045139  
rat: replaced 0.7093727308458327 by 29189494/41148317 = 0.7093727308458326  
rat: replaced 0.7284093028468864 by 13153959/18058472 = 0.7284093028468854  
rat: replaced 0.7477430314413382 by 12236470/16364539 = 0.7477430314413379

```
rat: replaced 0.7673749832474404 by 39576757/51574208 = 0.7673749832474402  
rat: replaced 0.7873061950613714 by 6818881/8661028 = 0.7873061950613714  
rat: replaced 0.8075376737535601 by 20498953/25384516 = 0.807537673753559  
rat: replaced 0.8280703961679966 by 6989671/8440914 = 0.8280703961679979  
rat: replaced 0.84890530902455 by 25231431/29722315 = 0.8489053090245494  
rat: replaced 0.8700433288242969 by 9721738/11173855 = 0.8700433288242957  
rat: replaced 0.8914853417578728 by 33469619/37543656 = 0.8914853417578725  
rat: replaced 0.9132322036168524 by 21961040/24047597 = 0.913232203616852  
part: invalid index of list or matrix.  
#0: lineIntersection(g=[-1,3,6],h=[3,1,6])  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
... ersection(c,h) // Q titik potong garis c=BC dan h ...
```

```
>$projectToLine(A,lineThrough(B,C)) // proyeksi A pada BC
```

```
Maxima said:  
rat: replaced 5.049958083474387e-5 by 102157/2022927682 = 5.049958083474385e-5  
rat: replaced 2.039932534230044e-4 by 284619/1395237319 = 2.039932534230043e-4  
rat: replaced 4.634656435254722e-4 by 573493/1237401322 = 4.634656435254721e-4
```

rat: replaced  $8.31890779119604e-4$  by  $332331/399488741 = 8.318907791196046e-4$   
rat: replaced  $0.001312231792998733$  by  $448125/341498356 = 0.001312231792998734$   
rat: replaced  $0.001907440626462018$  by  $276030/144712237 = 0.001907440626462018$   
rat: replaced  $0.002620457734122131$  by  $2586613/987084419 = 0.002620457734122131$   
rat: replaced  $0.00345421178986248$  by  $3402379/984994322 = 0.00345421178986248$   
rat: replaced  $0.004411619393972596$  by  $966955/219183686 = 0.004411619393972597$   
rat: replaced  $0.005495584781489732$  by  $2798484/509224061 = 0.005495584781489734$   
rat: replaced  $0.006708999531778753$  by  $6054060/902378957 = 0.006708999531778753$   
rat: replaced  $0.008054742279375651$  by  $806546/100133061 = 0.00805474227937564$   
rat: replaced  $0.009535678426127348$  by  $4115324/431571181 = 0.009535678426127346$   
rat: replaced  $0.01115465985465333$  by  $2266398/203179481 = 0.01115465985465334$   
rat: replaced  $0.01291452464316009$  by  $2106925/163143829 = 0.01291452464316012$   
rat: replaced  $0.01481809678163515$  by  $2779203/187554653 = 0.01481809678163516$   
rat: replaced  $0.01686818588945119$  by  $7427428/440321683 = 0.01686818588945119$   
rat: replaced  $0.01906758693440599$  by  $2278085/119474216 = 0.019067586934406$   
rat: replaced  $0.02141907995322798$  by  $2316386/108145915 = 0.02141907995322801$   
rat: replaced  $0.02392542977357476$  by  $1665518/69612877 = 0.02392542977357479$   
rat: replaced  $0.02658938573755304$  by  $3678645/138350131 = 0.02658938573755308$

rat: replaced 0.02941368142678652 by 4053557/137811957 = 0.0294136814267865  
rat: replaced 0.03240103438906003 by 2629160/81144323 = 0.03240103438906009  
rat: replaced 0.03555414586656669 by 1834427/51595305 = 0.03555414586656674  
rat: replaced 0.03887570052578646 by 1643964/42287701 = 0.03887570052578645  
rat: replaced 0.04236836618902146 by 3055464/72116635 = 0.04236836618902143  
rat: replaced 0.04603479356761608 by 3139251/68193007 = 0.0460347935676161  
rat: replaced 0.04987761599688789 by 5203437/104324092 = 0.04987761599688785  
rat: replaced 0.05389944917279615 by 4533622/84112585 = 0.0538994491727962  
rat: replaced 0.05810289089037535 by 11687290/201148167 = 0.05810289089037535  
rat: replaced 0.06249052078395612 by 3949243/63197473 = 0.06249052078395603  
rat: replaced 0.0670649000692059 by 3281728/48933615 = 0.067064900069206  
rat: replaced 0.07182857128700804 by 4146139/57722699 = 0.07182857128700791  
rat: replaced 0.07678405804921068 by 1198255/15605518 = 0.07678405804921054  
rat: replaced 0.08193386478626702 by 5956639/72700574 = 0.08193386478626702  
rat: replaced 0.08728047649679532 by 2799808/32078285 = 0.08728047649679527  
rat: replaced 0.09282635849907966 by 10292829/110882611 = 0.09282635849907972  
rat: replaced 0.09857395618454184 by 4198057/42587892 = 0.09857395618454184

rat: replaced 0.1045256947732028 by 30563827/292404916 = 0.1045256947732028  
rat: replaced 0.1106839790711635 by 8949559/80856860 = 0.1106839790711635  
rat: replaced 0.1170511932301264 by 9911603/84677505 = 0.1170511932301265  
rat: replaced 0.1236297005089814 by 19561703/158228184 = 0.1236297005089814  
rat: replaced 0.1304218430374826 by 4975231/38147222 = 0.1304218430374825  
rat: replaced 0.137429941582038 by 3502939/25488907 = 0.137429941582038  
rat: replaced 0.1446562953136327 by 15521432/107298697 = 0.1446562953136327  
rat: replaced 0.1521031815779155 by 18080502/118869979 = 0.1521031815779155  
rat: replaced 0.1597728556674664 by 37419026/234201397 = 0.1597728556674664  
rat: replaced 0.1676675505962674 by 22585897/134706429 = 0.1676675505962674  
rat: replaced 0.1757894768764047 by 15940893/90681725 = 0.1757894768764048  
rat: replaced 0.1841408222970185 by 7944795/43145213 = 0.1841408222970182  
rat: replaced 0.1927237517055264 by 1392861/7227241 = 0.1927237517055264  
rat: replaced 0.2015404067911402 by 1735485/8611102 = 0.2015404067911401  
rat: replaced 0.2105929058706983 by 10627754/50465869 = 0.2105929058706985  
rat: replaced 0.2198833436768368 by 9372347/42624179 = 0.2198833436768366  
rat: replaced 0.2294137911485169 by 7405273/32279110 = 0.2294137911485168  
rat: replaced 0.239186295223934 by 27692337/115777273 = 0.239186295223934

rat: replaced 0.2492028786358237 by 8925310/35815437 = 0.249202878635824  
rat: replaced 0.2594655397091927 by 11150701/42975653 = 0.259465539709193  
rat: replaced 0.2699762521614856 by 11249087/41666950 = 0.2699762521614853  
rat: replaced 0.280736964905216 by 12097010/43090193 = 0.2807369649052164  
rat: replaced 0.2917496018530771 by 14831788/50837389 = 0.2917496018530771  
rat: replaced 0.3030160617255513 by 18597622/61375037 = 0.3030160617255514  
rat: replaced 0.3145382178610399 by 11102944/35299189 = 0.3145382178610392  
rat: replaced 0.3263179180285316 by 13053510/40002431 = 0.3263179180285318  
rat: replaced 0.3383569842428258 by 13796661/40775458 = 0.3383569842428257  
rat: replaced 0.3506572125823338 by 33496033/95523582 = 0.3506572125823338  
rat: replaced 0.3632203730094723 by 23086207/63559780 = 0.3632203730094724  
rat: replaced 0.376048209193667 by 22674222/60296051 = 0.3760482091936668  
rat: replaced 0.3891424383369902 by 33246815/85436107 = 0.3891424383369902  
rat: replaced 0.402504751002439 by 7793813/19363282 = 0.4025047510024385  
rat: replaced 0.4161368109448825 by 10481453/25187517 = 0.4161368109448819  
rat: replaced 0.4300402549446862 by 19565443/45496771 = 0.4300402549446861  
rat: replaced 0.4442166926440365 by 16102633/36249500 = 0.4442166926440365

rat: replaced 0.4586677063859775 by 19404529/42306290 = 0.4586677063859771  
rat: replaced 0.4733948510561774 by 10262860/21679281 = 0.4733948510561766  
rat: replaced 0.4883996539274416 by 4159841/8517289 = 0.488399653927441  
rat: replaced 0.5036836145069872 by 13202363/26211619 = 0.5036836145069864  
rat: replaced 0.5192482043864929 by 12221370/23536663 = 0.5192482043864927  
rat: replaced 0.5350948670949413 by 52965833/98984005 = 0.5350948670949413  
rat: replaced 0.551225017954267 by 14288533/25921416 = 0.551225017954266  
rat: replaced 0.5676400439378262 by 25565995/45039097 = 0.5676400439378259  
rat: replaced 0.5843413035316997 by 18888222/32323955 = 0.5843413035316997  
rat: replaced 0.6013301265988455 by 5789399/9627655 = 0.6013301265988447  
rat: replaced 0.6186078142461149 by 4803773/7765458 = 0.618607814246114  
rat: replaced 0.6361756386941407 by 13914515/21872128 = 0.6361756386941407  
rat: replaced 0.6540348431501183 by 48160581/73636109 = 0.6540348431501181  
rat: replaced 0.6721866416834846 by 6617334/9844489 = 0.6721866416834841  
rat: replaced 0.690632219104513 by 16840135/24383651 = 0.6906322191045139  
rat: replaced 0.7093727308458327 by 29189494/41148317 = 0.7093727308458326  
rat: replaced 0.7284093028468864 by 13153959/18058472 = 0.7284093028468854  
rat: replaced 0.7477430314413382 by 12236470/16364539 = 0.7477430314413379

```
rat: replaced 0.7673749832474404 by 39576757/51574208 = 0.7673749832474402
rat: replaced 0.7873061950613714 by 6818881/8661028 = 0.7873061950613714
rat: replaced 0.8075376737535601 by 20498953/25384516 = 0.807537673753559
rat: replaced 0.8280703961679966 by 6989671/8440914 = 0.8280703961679979
rat: replaced 0.84890530902455 by 25231431/29722315 = 0.8489053090245494
rat: replaced 0.8700433288242969 by 9721738/11173855 = 0.8700433288242957
rat: replaced 0.8914853417578728 by 33469619/37543656 = 0.8914853417578725
rat: replaced 0.9132322036168524 by 21961040/24047597 = 0.913232203616852
rat: replaced 1.498320841708742e-4 by 1329822/8875415485 = 1.498320841708742e-4
rat: replaced 5.986466935998108e-4 by 398723/666040595 = 5.986466935998098e-4
rat: replaced 0.001345398955533032 by 4525441/3363642421 = 0.001345398955533032
rat: replaced 0.002389014203700413 by 1071627/448564516 = 0.00238901420370041
rat: replaced 0.00372838808577948 by 661903/177530607 = 0.003728388085779485
rat: replaced 0.005362386673832029 by 5230891/975478144 = 0.005362386673832028
rat: replaced 0.007289846577694006 by 32241346/4422774287 = 0.007289846577694006
rat: replaced 0.009509575061314376 by 2146493/225719129 = 0.009509575061314364
rat: replaced 0.01202035016202822 by 1789188/148846579 = 0.01202035016202825
```

rat: replaced 0.01482092081275069 by 2581665/174190594 = 0.01482092081275066  
rat: replaced 0.01791000696708436 by 5107285/285163764 = 0.01791000696708436  
rat: replaced 0.02128629972732062 by 3323295/156123659 = 0.02128629972732064  
rat: replaced 0.02494846147533059 by 4548287/182307314 = 0.02494846147533061  
rat: replaced 0.02889512600632479 by 3147802/108938857 = 0.02889512600632481  
rat: replaced 0.0331248986654725 by 5858625/176864692 = 0.03312489866547248  
rat: replaced 0.03763635648736519 by 10043830/266865099 = 0.03763635648736518  
rat: replaced 0.04242804833831373 by 4635713/109260576 = 0.04242804833831372  
rat: replaced 0.04749849506145984 by 5610259/118114458 = 0.04749849506145979  
rat: replaced 0.05284618962468965 by 4237503/80185592 = 0.05284618962468968  
rat: replaced 0.05846959727133633 by 3317197/56733707 = 0.05846959727133642  
rat: replaced 0.06436715567365475 by 13427433/208606903 = 0.06436715567365477  
rat: replaced 0.07053727508905278 by 8025659/113778977 = 0.07053727508905269  
rat: replaced 0.07697833851906408 by 6306881/81930594 = 0.07697833851906408  
rat: replaced 0.08368870187104593 by 4282086/51166835 = 0.08368870187104596  
rat: replaced 0.09066669412258874 by 2175091/23989967 = 0.09066669412258883  
rat: replaced 0.09791061748861546 by 8290049/84669561 = 0.09791061748861554  
rat: replaced 0.1054187475911595 by 8501563/80645646 = 0.1054187475911595

```
rat: replaced 0.1131893336318011 by 6539019/57770629 = 0.113189333631801
rat: replaced 0.121220598566744 by 5779101/47674249 = 0.1212205985667441
rat: replaced 0.1295107392845216 by 7134865/55090914 = 0.1295107392845216
rat: replaced 0.1380579267863034 by 6113057/44278928 = 0.1380579267863034
rat: replaced 0.1468603063687953 by 6311140/42973763 = 0.1468603063687953
rat: replaced 0.1559159978097077 by 4027079/25828517 = 0.1559159978097078
rat: replaced 0.1652230955557758 by 10597125/64138279 = 0.1652230955557757
rat: replaced 0.1747796689133147 by 9649007/55206690 = 0.1747796689133147
rat: replaced 0.1845837622412855 by 6871913/37229239 = 0.1845837622412857
rat: replaced 0.1946333951468589 by 39341769/202132676 = 0.1946333951468589
rat: replaced 0.2049265626834523 by 10758647/52500012 = 0.2049265626834523
rat: replaced 0.2154612355512225 by 33702610/156420759 = 0.2154612355512225
rat: replaced 0.2262353602999955 by 2338161/10335082 = 0.2262353602999957
rat: replaced 0.2372468595346078 by 7573078/31920667 = 0.2372468595346081
rat: replaced 0.2484936321226457 by 3764353/15148690 = 0.2484936321226456
rat: replaced 0.2599735534045555 by 26335713/101301508 = 0.2599735534045554
rat: replaced 0.2716844754061095 by 29831699/109802737 = 0.2716844754061094
```

rat: replaced 0.2836242270531998 by  $\frac{15100773}{53242183} = 0.2836242270531995$   
rat: replaced 0.2957906143889442 by  $\frac{2942977}{9949528} = 0.2957906143889439$   
rat: replaced 0.3081814207930817 by  $\frac{12077608}{39189929} = 0.3081814207930818$   
rat: replaced 0.3207944072036307 by  $\frac{9185023}{28632117} = 0.3207944072036308$   
rat: replaced 0.333627312340794 by  $\frac{5228336}{15671187} = 0.3336273123407946$   
rat: replaced 0.346677852933085 by  $\frac{15615111}{45042136} = 0.3466778529330847$   
rat: replaced 0.3599437239456539 by  $\frac{7564465}{21015688} = 0.3599437239456543$   
rat: replaced 0.3734225988107874 by  $\frac{7702871}{20627758} = 0.3734225988107869$   
rat: replaced 0.3871121296605642 by  $\frac{97723109}{252441351} = 0.3871121296605642$   
rat: replaced 0.4010099475616409 by  $\frac{3146543}{7846546} = 0.4010099475616405$   
rat: replaced 0.4151136627521425 by  $\frac{6219049}{14981557} = 0.4151136627521425$   
rat: replaced 0.4294208648806354 by  $\frac{26148647}{60892819} = 0.4294208648806356$   
rat: replaced 0.4439291232471635 by  $\frac{19525684}{43983787} = 0.4439291232471638$   
rat: replaced 0.458635987046313 by  $\frac{38604672}{84172793} = 0.4586359870463132$   
rat: replaced 0.4735389856122937 by  $\frac{11146199}{23538081} = 0.4735389856122935$   
rat: replaced 0.488635628666001 by  $\frac{13946471}{28541658} = 0.4886356286660011$   
rat: replaced 0.5039234065640431 by  $\frac{5948069}{11803518} = 0.503923406564043$   
rat: replaced 0.5193997905497036 by  $\frac{24027011}{46259185} = 0.5193997905497038$

rat: replaced 0.5350622330058146 by  $7363779/13762472 = 0.5350622330058147$   
rat: replaced 0.5509081677095147 by  $8130825/14758948 = 0.5509081677095142$   
rat: replaced 0.5669350100888726 by  $10250363/18080314 = 0.5669350100888735$   
rat: replaced 0.5831401574813392 by  $37655026/64572857 = 0.5831401574813393$   
rat: replaced 0.5995209893940125 by  $30778651/51338738 = 0.5995209893940128$   
rat: replaced 0.6160748677656853 by  $23698401/38466755 = 0.616074867765685$   
rat: replaced 0.6327991372306488 by  $5052598/7984521 = 0.6327991372306492$   
rat: replaced 0.6496911253842265 by  $60646047/93345968 = 0.6496911253842266$   
rat: replaced 0.666748143050013 by  $30125566/45182827 = 0.6667481430500132$   
rat: replaced 0.6839674845487889 by  $8953739/13090884 = 0.6839674845487899$   
rat: replaced 0.7013464279690875 by  $7888577/11247761 = 0.7013464279690864$   
rat: replaced 0.7188822354393821 by  $16662338/23178119 = 0.7188822354393815$   
rat: replaced 0.7365721534018723 by  $13899283/18870226 = 0.7365721534018723$   
rat: replaced 0.7544134128878366 by  $16270763/21567436 = 0.754413412887837$   
rat: replaced 0.7724032297945274 by  $8203205/10620366 = 0.7724032297945287$   
rat: replaced 0.7905388051635788 by  $10794522/13654639 = 0.7905388051635784$   
rat: replaced 0.8088173254609005 by  $16745047/20703126 = 0.808817325460899$

rat: replaced 0.8272359628580275 by 20291194/24528907 = 0.827235962858027  
rat: replaced 0.8457918755149025 by 10996366/13001267 = 0.8457918755149018  
rat: replaced 0.8644822078640563 by 9158500/10594203 = 0.8644822078640555  
rat: replaced 0.8833040908961625 by 13759446/15577247 = 0.8833040908961641  
rat: replaced 0.9022546424469358 by 19827819/21975857 = 0.9022546424469362  
rat: replaced 0.9213309674853474 by 60458149/65620446 = 0.9213309674853475  
rat: replaced 0.9405301584031224 by 11658841/12396031 = 0.9405301584031212  
rat: replaced 0.9598492953055026 by 26214088/27310629 = 0.9598492953055018  
rat: replaced 0.9792854463032298 by 35089005/35831233 = 0.9792854463032293  
rat: replaced 0.9988356678057343 by 15735752/15754095 = 0.9988356678057356  
rat: replaced 1.018497004815491 by 16202286/15908035 = 1.018497004815491  
rat: replaced 1.038266491223517 by 17763365/17108676 = 1.038266491223517  
rat: replaced 1.058141150105979 by 33730321/31876958 = 1.058141150105979  
rat: replaced 1.078117994021884 by 51996446/48228901 = 1.078117994021883  
rat: replaced 1.098194025311821 by 124719922/113568203 = 1.098194025311821  
rat: replaced 1.118366236397724 by 92837336/83011569 = 1.118366236397724  
rat: replaced 1.13863161008363 by 20601995/18093644 = 1.138631610083629  
rat: replaced 1.158987119857388 by 20626233/17796775 = 1.15898711985739

```

rat: replaced 1.17942973019332 by 4098089/3474636 = 1.179429730193321

rat: replaced 1.199956396855759 by 17442145/14535649 = 1.199956396855758
part: invalid index of list or matrix.
#0: lineIntersection(g=[3,1,6],h=[-1,3,6])
#1: projectToLine(a=[2,0],g=[-1,3,6])
-- an error. To debug this try: debugmode(true);

Error in:
$projectToLine(A,lineThrough(B,C)) // proyeksi A pada BC ...

```

```

>$distance(A,Q) // jarak AQ
>cc &= circleThrough(A,B,C); $cc // (titik pusat dan jari-jari) lingkaran melalui A, B, C

```

```

Maxima said:
rat: replaced -9.96658350028018e-5 by -86001/862893488 = -9.966583500280164e-5

rat: replaced -3.973200535106469e-4 by -1080775/2720162223 = -3.973200535106468e-4

rat: replaced -8.90932907016237e-4 by -1194571/1340809157 = -8.90932907016237e-4

rat: replaced -0.001578455051312488 by -2522953/1598368606 = -0.001578455051312488

rat: replaced -0.002457817751424091 by -519814/211494119 = -0.002457817751424095

rat: replaced -0.003526933088480816 by -11813191/3349423055 = -0.003526933088480816

rat: replaced -0.004783694168506353 by -866601/181157275 = -0.004783694168506343

rat: replaced -0.006225975333106509 by -4878061/783501498 = -0.00622597533310651

```

rat: replaced -0.00785163237203354 by -1241039/158061272 = -0.007851632372033549  
rat: replaced -0.009658502737604657 by -6031380/624463249 = -0.009658502737604659  
rat: replaced -0.01164440576095599 by -2532373/217475503 = -0.01164440576095598  
rat: replaced -0.01380714287010623 by -1331361/96425525 = -0.01380714287010623  
rat: replaced -0.01614449780981353 by -7953293/492631799 = -0.01614449780981353  
rat: replaced -0.0186542368631985 by -865030/46371771 = -0.01865423686319852  
rat: replaced -0.02133410907511396 by -2814913/131944249 = -0.02133410907511399  
rat: replaced -0.02418184647723809 by -2509632/103781653 = -0.02418184647723813  
rat: replaced -0.02719516431487051 by -1827823/67211324 = -0.02719516431487051  
rat: replaced -0.03037176127540547 by -9190485/302599672 = -0.03037176127540548  
rat: replaced -0.03370931971846053 by -3905653/115862706 = -0.03370931971846057  
rat: replaced -0.03720550590763916 by -2032371/54625544 = -0.03720550590763911  
rat: replaced -0.04085797024390259 by -2827822/69211025 = -0.04085797024390261  
rat: replaced -0.04466434750052761 by -3719233/83270734 = -0.04466434750052762  
rat: replaced -0.04862225705962725 by -2754536/56651751 = -0.04862225705962733  
rat: replaced -0.05272930315021007 by -5066672/96088355 = -0.05272930315021003  
rat: replaced -0.05698307508775641 by -9699307/170213822 = -0.05698307508775639

rat: replaced  $-0.06138114751528378$  by  $-7938451/129330443 = -0.06138114751528378$

rat: replaced  $-0.0659210806458812$  by  $-2936449/44544916 = -0.06592108064588112$

rat: replaced  $-0.07060042050668569$  by  $-4716201/66801316 = -0.07060042050668583$

rat: replaced  $-0.07541669918427674$  by  $-2448749/32469586 = -0.07541669918427664$

rat: replaced  $-0.08036743507146715$  by  $-2461511/30628214 = -0.08036743507146711$

rat: replaced  $-0.08545013311546024$  by  $-13954421/163304848 = -0.08545013311546024$

rat: replaced  $-0.09066228506735385$  by  $-4103116/45257143 = -0.0906622850673539$

rat: replaced  $-0.09600136973296292$  by  $-16995415/177033047 = -0.0960013697329629$

rat: replaced  $-0.1014648532249364$  by  $-7634177/75239620 = -0.1014648532249365$

rat: replaced  $-0.107050189216145$  by  $-3894269/36377974 = -0.1070501892161449$

rat: replaced  $-0.1127548191943103$  by  $-9512927/84368252 = -0.1127548191943102$

rat: replaced  $-0.1185761727178553$  by  $-6978418/58851773 = -0.1185761727178551$

rat: replaced  $-0.1245116676729451$  by  $-3380435/27149544 = -0.1245116676729451$

rat: replaced  $-0.1305587105316969$  by  $-7571267/57991282 = -0.1305587105316968$

rat: replaced  $-0.136714696611531$  by  $-8109727/59318619 = -0.136714696611531$

rat: replaced  $-0.1429770103356357$  by  $-5513427/38561633 = -0.142977010335636$

rat: replaced  $-0.1493430254945241$  by  $-2259975/15132779 = -0.1493430254945242$

rat: replaced  $-0.1558101055086514$  by  $-1315594/8443573 = -0.1558101055086514$

rat: replaced  $-0.1623756036920724$  by  $-5159837/31777169 = -0.1623756036920721$   
rat: replaced  $-0.1690368635171068$  by  $-3076049/18197504 = -0.1690368635171065$   
rat: replaced  $-0.1757912188799892$  by  $-8356449/47536214 = -0.1757912188799891$   
rat: replaced  $-0.1826359943674788$  by  $-20067867/109879036 = -0.1826359943674788$   
rat: replaced  $-0.1895685055243976$  by  $-10432363/55032153 = -0.1895685055243977$   
rat: replaced  $-0.1965860591220733$  by  $-4406725/22416264 = -0.1965860591220732$   
rat: replaced  $-0.2036859534276606$  by  $-3912367/19207839 = -0.2036859534276604$   
rat: replaced  $-0.2108654784743126$  by  $-11495573/54516145 = -0.2108654784743125$   
rat: replaced  $-0.2181219163321738$  by  $-13126833/60181174 = -0.2181219163321739$   
rat: replaced  $-0.225452541380172$  by  $-5509494/24437489 = -0.2254525413801721$   
rat: replaced  $-0.232854620578578$  by  $-20847643/89530725 = -0.2328546205785779$   
rat: replaced  $-0.2403254137423072$  by  $-9494831/39508227 = -0.2403254137423074$   
rat: replaced  $-0.2478621738149347$  by  $-6380796/25743323 = -0.2478621738149345$   
rat: replaced  $-0.2554621471434013$  by  $-34172111/133765849 = -0.2554621471434013$   
rat: replaced  $-0.2631225737533733$  by  $-13929723/52940053 = -0.2631225737533734$   
rat: replaced  $-0.2708406876252407$  by  $-56284033/207812325 = -0.2708406876252407$   
rat: replaced  $-0.2786137169707144$  by  $-23181966/83204683 = -0.2786137169707142$

rat: replaced  $-0.2864388845100037$  by  $-11672339/40749841 = -0.2864388845100034$   
rat: replaced  $-0.2943134077495424$  by  $-9731821/33066183 = -0.2943134077495428$   
rat: replaced  $-0.3022344992602357$  by  $-9353258/30947023 = -0.3022344992602358$   
rat: replaced  $-0.3101993669561991$  by  $-31708610/102220099 = -0.3101993669561991$   
rat: replaced  $-0.3182052143739678$  by  $-9026555/28367087 = -0.318205214373968$   
rat: replaced  $-0.3262492409521378$  by  $-20870146/63969945 = -0.3262492409521378$   
rat: replaced  $-0.3343286423114211$  by  $-163875765/490163702 = -0.3343286423114211$   
rat: replaced  $-0.3424406105350815$  by  $-14035276/40986015 = -0.3424406105350813$   
rat: replaced  $-0.350582334449723$  by  $-3184915/9084642 = -0.350582334449723$   
rat: replaced  $-0.3587509999064056$  by  $-5293418/14755131 = -0.3587509999064055$   
rat: replaced  $-0.3669437900620574$  by  $-16784286/45740755 = -0.3669437900620574$   
rat: replaced  $-0.3751578856611566$  by  $-29031339/77384323 = -0.3751578856611564$   
rat: replaced  $-0.3833904653176554$  by  $-32029406/83542521 = -0.3833904653176554$   
rat: replaced  $-0.3916387057971147$  by  $-42559854/108671215 = -0.3916387057971147$   
rat: replaced  $-0.3998997822990269$  by  $-5994245/14989368 = -0.3998997822990269$   
rat: replaced  $-0.4081708687392926$  by  $-35695538/87452439 = -0.4081708687392927$   
rat: replaced  $-0.416449138032827$  by  $-10407784/24991729 = -0.4164491380328268$   
rat: replaced  $-0.4247317623762659$  by  $-20393053/48013958 = -0.4247317623762656$

rat: replaced -0.4330159135307439 by -16978376/39209589 = -0.4330159135307437  
rat: replaced -0.4412987631047152 by -13590227/30795978 = -0.4412987631047145  
rat: replaced -0.4495774828367916 by -27127361/60339679 = -0.4495774828367914  
rat: replaced -0.4578492448785652 by -14370001/31385879 = -0.4578492448785647  
rat: replaced -0.4661112220773918 by -24206411/51932693 = -0.4661112220773916  
rat: replaced -0.4743605882591027 by -11052217/23299189 = -0.474360588259102  
rat: replaced -0.4825945185106215 by -34783885/72076834 = -0.4825945185106216  
rat: replaced -0.4908101894624506 by -9304730/18957899 = -0.4908101894624505  
rat: replaced -0.4990047795710082 by -12351268/24751803 = -0.4990047795710074  
rat: replaced -0.5071754694007786 by -11299519/22279309 = -0.507175469400779  
rat: replaced -0.5153194419062543 by -6871877/13335179 = -0.5153194419062541  
rat: replaced -0.5234338827136382 by -4491460/8580759 = -0.5234338827136388  
rat: replaced -0.5315159804022782 by -13987981/26317141 = -0.5315159804022785  
rat: replaced -0.5395629267858069 by -42101104/78028163 = -0.5395629267858069  
rat: replaced -0.5475719171929583 by -3020462/5516101 = -0.5475719171929593  
rat: replaced -0.5555401507480326 by -19638186/35349715 = -0.5555401507480329  
rat: replaced -0.5634648306509809 by -21674756/38466919 = -0.563464830650981

rat: replaced  $-0.5713431644570837$  by  $-35597565/62305051 = -0.5713431644570839$   
rat: replaced  $-0.5791723643561919$  by  $-1443012/2491507 = -0.5791723643561909$   
rat: replaced  $-0.5869496474515068$  by  $-23586220/40184401 = -0.5869496474515074$   
rat: replaced  $-0.5946722360378665$  by  $-17526553/29472627 = -0.5946722360378666$   
rat: replaced  $-1.501654158375457e-4$  by  $-374996/2497219469 = -1.501654158375457e-4$   
rat: replaced  $-6.013133069336513e-4$  by  $-664019/1104281233 = -6.013133069336514e-4$   
rat: replaced  $-0.001354398550541709$  by  $-654983/483596944 = -0.001354398550541709$   
rat: replaced  $-0.002410345830432092$  by  $-1208607/501424727 = -0.002410345830432092$   
rat: replaced  $-0.003770049544422824$  by  $-471953/125184827 = -0.003770049544422824$   
rat: replaced  $-0.005434373714942833$  by  $-1223803/225196695 = -0.005434373714942841$   
rat: replaced  $-0.007404151902628484$  by  $-1775171/239753455 = -0.007404151902628473$   
rat: replaced  $-0.009680187122968989$  by  $-1826977/188733645 = -0.009680187122968986$   
rat: replaced  $-0.01226325176600614$  by  $-549289/44791464 = -0.01226325176600613$   
rat: replaced  $-0.01515408751909439$  by  $-5645196/372519691 = -0.01515408751909439$   
rat: replaced  $-0.01835340529273474$  by  $-1469077/80043838 = -0.01835340529273471$   
rat: replaced  $-0.02186188514948188$  by  $-3538485/161856353 = -0.0218618851494819$   
rat: replaced  $-0.02568017623594088$  by  $-2586227/100709083 = -0.0256801762359409$   
rat: replaced  $-0.02980889671785183$  by  $-3946092/132379673 = -0.02980889671785184$

rat: replaced -0.03424863371827405 by  $-13149221/383934177 = -0.03424863371827406$   
rat: replaced -0.03899994325887324 by  $-13884089/356002800 = -0.03899994325887324$   
rat: replaced -0.0440633502043217 by  $-1745785/39619888 = -0.04406335020432163$   
rat: replaced -0.04943934820981147 by  $-5036973/101881865 = -0.04943934820981143$   
rat: replaced -0.0551283996716885 by  $-7433459/134839013 = -0.05512839967168849$   
rat: replaced -0.06113093568121392 by  $-4757027/77817016 = -0.06113093568121399$   
rat: replaced -0.06744735598145563 by  $-3884855/57598329 = -0.06744735598145564$   
rat: replaced -0.07407802892731413 by  $-2885255/38948863 = -0.07407802892731426$   
rat: replaced -0.08102329144868728 by  $-6021225/74314742 = -0.08102329144868727$   
rat: replaced -0.08828344901677676 by  $-4377003/49578976 = -0.08828344901677679$   
rat: replaced -0.09585877561354286 by  $-7052907/73576018 = -0.09585877561354299$   
rat: replaced -0.1037495137043052 by  $-5876631/56642492 = -0.1037495137043052$   
rat: replaced -0.1119558742134973 by  $-7474079/66759150 = -0.1119558742134973$   
rat: replaced -0.1204780365035736 by  $-7358791/61079938 = -0.1204780365035734$   
rat: replaced -0.1293161483570729 by  $-5061354/39139381 = -0.1293161483570729$   
rat: replaced -0.1384703259618425 by  $-5586207/40342268 = -0.1384703259618423$   
rat: replaced -0.1479406538994164 by  $-14042248/94918115 = -0.1479406538994164$

rat: replaced  $-0.1577271851365598$  by  $-1401295/8884296 = -0.1577271851365601$   
rat: replaced  $-0.167829941019971$  by  $-12121567/72225295 = -0.1678299410199709$   
rat: replaced  $-0.178248911274147$  by  $-15269783/85665505 = -0.178248911274147$   
rat: replaced  $-0.188984054002412$  by  $-7617649/40308422 = -0.1889840540024117$   
rat: replaced  $-0.2000352956911056$  by  $-20506971/102516763 = -0.2000352956911056$   
rat: replaced  $-0.2114025312169349$  by  $-5553806/26271237 = -0.2114025312169351$   
rat: replaced  $-0.2230856238574869$  by  $-19624843/87970003 = -0.223085623857487$   
rat: replaced  $-0.2350844053048997$  by  $-15894843/67613345 = -0.2350844053048995$   
rat: replaced  $-0.2473986756826945$  by  $-10672172/43137547 = -0.2473986756826947$   
rat: replaced  $-0.2600282035657621$  by  $-13234131/50894983 = -0.2600282035657621$   
rat: replaced  $-0.2729727260035054$  by  $-10344911/37897233 = -0.2729727260035053$   
rat: replaced  $-0.286231948546134$  by  $-21050803/73544561 = -0.2862319485461338$   
rat: replaced  $-0.2998055452741104$  by  $-5811723/19384975 = -0.2998055452741105$   
rat: replaced  $-0.3136931588307395$  by  $-10410719/33187587 = -0.3136931588307399$   
rat: replaced  $-0.3278944004579047$  by  $-32013736/97634287 = -0.3278944004579047$   
rat: replaced  $-0.3424088500349452$  by  $-8881888/25939423 = -0.3424088500349449$   
rat: replaced  $-0.357236056120665$  by  $-14764623/41330159 = -0.3572360561206648$   
rat: replaced  $-0.372375535998478$  by  $-13197485/35441332 = -0.3723755359984777$

rat: replaced  $-0.3878267757246791$  by  $-14427400/37200629 = -0.3878267757246793$   
rat: replaced  $-0.403589230179839$  by  $-12432000/30803597 = -0.4035892301798391$   
rat: replaced  $-0.419662323123314$  by  $-8483178/20214295 = -0.4196623231233145$   
rat: replaced  $-0.4360454472508704$  by  $-62882267/144210351 = -0.4360454472508704$   
rat: replaced  $-0.4527379642554148$  by  $-20707559/45738508 = -0.4527379642554147$   
rat: replaced  $-0.4697392048908241$  by  $-17485109/37223014 = -0.4697392048908237$   
rat: replaced  $-0.4870484690388687$  by  $-31502783/64681002 = -0.4870484690388686$   
rat: replaced  $-0.504665025779225$  by  $-2755088/5459241 = -0.5046650257792247$   
rat: replaced  $-0.5225881134625661$  by  $-18908824/36183035 = -0.5225881134625661$   
rat: replaced  $-0.5408169397867263$  by  $-7900905/14609204 = -0.5408169397867262$   
rat: replaced  $-0.5593506818759304$  by  $-9811459/17540801 = -0.5593506818759303$   
rat: replaced  $-0.5781884863630807$  by  $-55545197/96067629 = -0.5781884863630807$   
rat: replaced  $-0.5973294694750937$  by  $-15924011/26658673 = -0.5973294694750936$   
rat: replaced  $-0.6167727171212756$  by  $-14518184/23538953 = -0.6167727171212756$   
rat: replaced  $-0.6365172849847307$  by  $-12285310/19300827 = -0.6365172849847315$   
rat: replaced  $-0.6565621986167935$  by  $-11361539/17304589 = -0.6565621986167947$   
rat: replaced  $-0.6769064535344717$  by  $-8579417/12674450 = -0.6769064535344729$

rat: replaced  $-0.6975490153208934$  by  $-31186954/44709337 = -0.6975490153208938$   
rat: replaced  $-0.7184888197287485$  by  $-37348784/51982415 = -0.7184888197287487$   
rat: replaced  $-0.7397247727867132$  by  $-12753239/17240519 = -0.7397247727867126$   
rat: replaced  $-0.7612557509088446$  by  $-14688094/19294559 = -0.7612557509088443$   
rat: replaced  $-0.7830806010069399$  by  $-11512064/14700995 = -0.7830806010069387$   
rat: replaced  $-0.8051981406058428$  by  $-15691583/19487853 = -0.805198140605843$   
rat: replaced  $-0.8276071579616919$  by  $-21531267/26016289 = -0.8276071579616908$   
rat: replaced  $-0.8503064121830922$  by  $-19412384/22829869 = -0.8503064121830922$   
rat: replaced  $-0.8732946333552043$  by  $-19396479/22210693 = -0.8732946333552042$   
rat: replaced  $-0.8965705226667342$  by  $-45044189/50240542 = -0.896570522666734$   
rat: replaced  $-0.9201327525398142$  by  $-6632592/7208299 = -0.9201327525398155$   
rat: replaced  $-0.9439799667627587$  by  $-19068047/20199631 = -0.9439799667627592$   
rat: replaced  $-0.9681107806256851$  by  $-17605779/18185707 = -0.9681107806256859$   
rat: replaced  $-0.9925237810589822$  by  $-24449409/24633575 = -0.9925237810589815$   
rat: replaced  $-1.017217526774618$  by  $-31992660/31451149 = -1.017217526774618$   
rat: replaced  $-1.042190548410265$  by  $-17752654/17033981 = -1.042190548410263$   
rat: replaced  $-1.067441348676237$  by  $-34132828/31976303 = -1.067441348676237$   
rat: replaced  $-1.092968402505218$  by  $-17060687/15609497 = -1.092968402505218$

```
rat: replaced -1.118770157204762 by -23160047/20701345 = -1.118770157204761
rat: replaced -1.144845032612569 by -13646839/11920250 = -1.144845032612571
rat: replaced -1.171191421254493 by -12158144/10381005 = -1.171191421254493
rat: replaced -1.197807688505292 by -18610041/15536752 = -1.197807688505294
rat: replaced -1.224692172752087 by -1607117/1312262 = -1.224692172752088
rat: replaced -1.251843185560525 by -23622341/18870048 = -1.251843185560524
rat: replaced -1.279259011843616 by -11838497/9254183 = -1.279259011843617
rat: replaced -1.306937910033247 by -16769881/12831429 = -1.306937910033247
rat: replaced -1.33487811225433 by -18675990/13990783 = -1.334878112254332
rat: replaced -1.363077824501593 by -60815834/44616553 = -1.363077824501592
rat: replaced -1.391535226818977 by -16386165/11775602 = -1.391535226818977
rat: replaced -1.420248473481634 by -17316077/12192287 = -1.420248473481636
rat: replaced -1.449215693180489 by -19585953/13514864 = -1.449215693180486
rat: replaced -1.478434989209379 by -157494279/106527700 = -1.478434989209379
rat: replaced -1.507904439654719 by -39585536/26252019 = -1.507904439654718
part: invalid index of list or matrix.
#0: lineIntersection(g=[2,-2,0],h=[-1,-3,-7])
#1: circleThrough(a=[2,0],b=[0,2],c=[3,3])
-- an error. To debug this try: debugmode(true);
```

```
Error in:  
cc &= circleThrough(A,B,C); $cc // (titik pusat dan jari-jari) ...  
^
```

```
>r&=getCircleRadius(cc); $r , $float(r) // tampilkan nilai jari-jari  
>$computeAngle(A,C,B) // nilai <ACB  
>$solve(getLineEquation(angleBisector(A,C,B),x,y),y)[1] // persamaan garis bagi <ACB
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);
```

```
Error in:  
... (getLineEquation(angleBisector(A,C,B),x,y),y)[1] // persamaan ...  
^
```

```
>P &= lineIntersection(angleBisector(A,C,B),angleBisector(C,B,A)); $P // titik potong 2
```

```
Maxima said:  
rat: replaced -9.96658350028018e-5 by -86001/862893488 = -9.966583500280164e-5  
  
rat: replaced -3.973200535106469e-4 by -1080775/2720162223 = -3.973200535106468e-4  
  
rat: replaced -8.90932907016237e-4 by -1194571/1340809157 = -8.90932907016237e-4  
  
rat: replaced -0.001578455051312488 by -2522953/1598368606 = -0.001578455051312488  
  
rat: replaced -0.002457817751424091 by -519814/211494119 = -0.002457817751424095
```

rat: replaced  $-0.003526933088480816$  by  $-11813191/3349423055 = -0.003526933088480816$

rat: replaced  $-0.004783694168506353$  by  $-866601/181157275 = -0.004783694168506343$

rat: replaced  $-0.006225975333106509$  by  $-4878061/783501498 = -0.00622597533310651$

rat: replaced  $-0.00785163237203354$  by  $-1241039/158061272 = -0.007851632372033549$

rat: replaced  $-0.009658502737604657$  by  $-6031380/624463249 = -0.009658502737604659$

rat: replaced  $-0.01164440576095599$  by  $-2532373/217475503 = -0.01164440576095598$

rat: replaced  $-0.01380714287010623$  by  $-1331361/96425525 = -0.01380714287010623$

rat: replaced  $-0.01614449780981353$  by  $-7953293/492631799 = -0.01614449780981353$

rat: replaced  $-0.0186542368631985$  by  $-865030/46371771 = -0.01865423686319852$

rat: replaced  $-0.02133410907511396$  by  $-2814913/131944249 = -0.02133410907511399$

rat: replaced  $-0.02418184647723809$  by  $-2509632/103781653 = -0.02418184647723813$

rat: replaced  $-0.02719516431487051$  by  $-1827823/67211324 = -0.02719516431487051$

rat: replaced  $-0.03037176127540547$  by  $-9190485/302599672 = -0.03037176127540548$

rat: replaced  $-0.03370931971846053$  by  $-3905653/115862706 = -0.03370931971846057$

rat: replaced  $-0.03720550590763916$  by  $-2032371/54625544 = -0.03720550590763911$

rat: replaced  $-0.04085797024390259$  by  $-2827822/69211025 = -0.04085797024390261$

rat: replaced  $-0.04466434750052761$  by  $-3719233/83270734 = -0.04466434750052762$

rat: replaced  $-0.04862225705962725$  by  $-2754536/56651751 = -0.04862225705962733$

rat: replaced  $-0.05272930315021007$  by  $-5066672/96088355 = -0.05272930315021003$

rat: replaced  $-0.05698307508775641$  by  $-9699307/170213822 = -0.05698307508775639$

rat: replaced  $-0.06138114751528378$  by  $-7938451/129330443 = -0.06138114751528378$

rat: replaced  $-0.0659210806458812$  by  $-2936449/44544916 = -0.06592108064588112$

rat: replaced  $-0.07060042050668569$  by  $-4716201/66801316 = -0.07060042050668583$

rat: replaced  $-0.07541669918427674$  by  $-2448749/32469586 = -0.07541669918427664$

rat: replaced  $-0.08036743507146715$  by  $-2461511/30628214 = -0.08036743507146711$

rat: replaced  $-0.08545013311546024$  by  $-13954421/163304848 = -0.08545013311546024$

rat: replaced  $-0.09066228506735385$  by  $-4103116/45257143 = -0.0906622850673539$

rat: replaced  $-0.09600136973296292$  by  $-16995415/177033047 = -0.0960013697329629$

rat: replaced  $-0.1014648532249364$  by  $-7634177/75239620 = -0.1014648532249365$

rat: replaced  $-0.107050189216145$  by  $-3894269/36377974 = -0.1070501892161449$

rat: replaced  $-0.1127548191943103$  by  $-9512927/84368252 = -0.1127548191943102$

rat: replaced  $-0.1185761727178553$  by  $-6978418/58851773 = -0.1185761727178551$

rat: replaced  $-0.1245116676729451$  by  $-3380435/27149544 = -0.1245116676729451$

rat: replaced  $-0.1305587105316969$  by  $-7571267/57991282 = -0.1305587105316968$

rat: replaced  $-0.136714696611531$  by  $-8109727/59318619 = -0.136714696611531$

rat: replaced  $-0.1429770103356357$  by  $-5513427/38561633 = -0.142977010335636$   
rat: replaced  $-0.1493430254945241$  by  $-2259975/15132779 = -0.1493430254945242$   
rat: replaced  $-0.1558101055086514$  by  $-1315594/8443573 = -0.1558101055086514$   
rat: replaced  $-0.1623756036920724$  by  $-5159837/31777169 = -0.1623756036920721$   
rat: replaced  $-0.1690368635171068$  by  $-3076049/18197504 = -0.1690368635171065$   
rat: replaced  $-0.1757912188799892$  by  $-8356449/47536214 = -0.1757912188799891$   
rat: replaced  $-0.1826359943674788$  by  $-20067867/109879036 = -0.1826359943674788$   
rat: replaced  $-0.1895685055243976$  by  $-10432363/55032153 = -0.1895685055243977$   
rat: replaced  $-0.1965860591220733$  by  $-4406725/22416264 = -0.1965860591220732$   
rat: replaced  $-0.2036859534276606$  by  $-3912367/19207839 = -0.2036859534276604$   
rat: replaced  $-0.2108654784743126$  by  $-11495573/54516145 = -0.2108654784743125$   
rat: replaced  $-0.2181219163321738$  by  $-13126833/60181174 = -0.2181219163321739$   
rat: replaced  $-0.225452541380172$  by  $-5509494/24437489 = -0.2254525413801721$   
rat: replaced  $-0.232854620578578$  by  $-20847643/89530725 = -0.2328546205785779$   
rat: replaced  $-0.2403254137423072$  by  $-9494831/39508227 = -0.2403254137423074$   
rat: replaced  $-0.2478621738149347$  by  $-6380796/25743323 = -0.2478621738149345$   
rat: replaced  $-0.2554621471434013$  by  $-34172111/133765849 = -0.2554621471434013$   
rat: replaced  $-0.2631225737533733$  by  $-13929723/52940053 = -0.2631225737533734$

rat: replaced -0.2708406876252407 by -56284033/207812325 = -0.2708406876252407

rat: replaced -0.2786137169707144 by -23181966/83204683 = -0.2786137169707142

rat: replaced -0.2864388845100037 by -11672339/40749841 = -0.2864388845100034

rat: replaced -0.2943134077495424 by -9731821/33066183 = -0.2943134077495428

rat: replaced -0.3022344992602357 by -9353258/30947023 = -0.3022344992602358

rat: replaced -0.3101993669561991 by -31708610/102220099 = -0.3101993669561991

rat: replaced -0.3182052143739678 by -9026555/28367087 = -0.318205214373968

rat: replaced -0.3262492409521378 by -20870146/63969945 = -0.3262492409521378

rat: replaced -0.3343286423114211 by -163875765/490163702 = -0.3343286423114211

rat: replaced -0.3424406105350815 by -14035276/40986015 = -0.3424406105350813

rat: replaced -0.350582334449723 by -3184915/9084642 = -0.350582334449723

rat: replaced -0.3587509999064056 by -5293418/14755131 = -0.3587509999064055

rat: replaced -0.3669437900620574 by -16784286/45740755 = -0.3669437900620574

rat: replaced -0.3751578856611566 by -29031339/77384323 = -0.3751578856611564

rat: replaced -0.3833904653176554 by -32029406/83542521 = -0.3833904653176554

rat: replaced -0.3916387057971147 by -42559854/108671215 = -0.3916387057971147

rat: replaced -0.3998997822990269 by -5994245/14989368 = -0.3998997822990269

rat: replaced  $-0.4081708687392926$  by  $-35695538/87452439 = -0.4081708687392927$   
rat: replaced  $-0.416449138032827$  by  $-10407784/24991729 = -0.4164491380328268$   
rat: replaced  $-0.4247317623762659$  by  $-20393053/48013958 = -0.4247317623762656$   
rat: replaced  $-0.4330159135307439$  by  $-16978376/39209589 = -0.4330159135307437$   
rat: replaced  $-0.4412987631047152$  by  $-13590227/30795978 = -0.4412987631047145$   
rat: replaced  $-0.4495774828367916$  by  $-27127361/60339679 = -0.4495774828367914$   
rat: replaced  $-0.4578492448785652$  by  $-14370001/31385879 = -0.4578492448785647$   
rat: replaced  $-0.4661112220773918$  by  $-24206411/51932693 = -0.4661112220773916$   
rat: replaced  $-0.4743605882591027$  by  $-11052217/23299189 = -0.474360588259102$   
rat: replaced  $-0.4825945185106215$  by  $-34783885/72076834 = -0.4825945185106216$   
rat: replaced  $-0.4908101894624506$  by  $-9304730/18957899 = -0.4908101894624505$   
rat: replaced  $-0.4990047795710082$  by  $-12351268/24751803 = -0.4990047795710074$   
rat: replaced  $-0.5071754694007786$  by  $-11299519/22279309 = -0.507175469400779$   
rat: replaced  $-0.5153194419062543$  by  $-6871877/13335179 = -0.5153194419062541$   
rat: replaced  $-0.5234338827136382$  by  $-4491460/8580759 = -0.5234338827136388$   
rat: replaced  $-0.5315159804022782$  by  $-13987981/26317141 = -0.5315159804022785$   
rat: replaced  $-0.5395629267858069$  by  $-42101104/78028163 = -0.5395629267858069$   
rat: replaced  $-0.5475719171929583$  by  $-3020462/5516101 = -0.5475719171929593$

```
rat: replaced -0.5555401507480326 by -19638186/35349715 = -0.5555401507480329
rat: replaced -0.5634648306509809 by -21674756/38466919 = -0.563464830650981
rat: replaced -0.5713431644570837 by -35597565/62305051 = -0.5713431644570839
rat: replaced -0.5791723643561919 by -1443012/2491507 = -0.5791723643561909
rat: replaced -0.5869496474515068 by -23586220/40184401 = -0.5869496474515074
rat: replaced -0.5946722360378665 by -17526553/29472627 = -0.5946722360378666
rat: replaced 1.66665833335744e-7 by 15819/94914474571 = 1.66665833335744e-7
rat: replaced 4.999958333473664e-5 by 201389/4027813565 = 4.999958333473664e-5
rat: replaced 1.33330666692022e-6 by 31771/23828726570 = 1.333306666920221e-6
rat: replaced 1.999933334222437e-4 by 200030/1000183339 = 1.999933334222437e-4
rat: replaced 4.499797504338432e-6 by 24036/5341573699 = 4.499797504338431e-6
rat: replaced 4.499662510124569e-4 by 1162901/2584418270 = 4.499662510124571e-4
rat: replaced 1.066581336583994e-5 by 58861/5518660226 = 1.066581336583993e-5
rat: replaced 7.998933390220841e-4 by 1137431/1421978337 = 7.998933390220838e-4
rat: replaced 2.083072932167196e-5 by 35635/1710693824 = 2.0830729321672e-5
rat: replaced 0.001249739605033717 by 567943/454449069 = 0.001249739605033716
rat: replaced 3.599352055540239e-5 by 98277/2730408098 = 3.599352055540234e-5
```

```
rat: replaced 0.00179946006479581 by 479561/266502719 = 0.001799460064795812
rat: replaced 5.71526624672386e-5 by 51154/895041417 = 5.715266246723866e-5
rat: replaced 0.002448999746720415 by 1946227/794702818 = 0.002448999746720415
rat: replaced 8.530603082730626e-5 by 121691/1426522824 = 8.530603082730627e-5
rat: replaced 0.003198293697380561 by 2986741/933854512 = 0.003198293697380562
rat: replaced 1.214508019889565e-4 by 158455/1304684674 = 1.214508019889563e-4
rat: replaced 0.004047266988005727 by 2125334/525128193 = 0.004047266988005727
rat: replaced 1.665833531718508e-4 by 142521/855553675 = 1.66583353171851e-4
rat: replaced 0.004995834721974179 by 1957223/391770967 = 0.004995834721974179
rat: replaced 2.216991628251896e-4 by 179571/809975995 = 2.216991628251896e-4
rat: replaced 0.006043902043303184 by 1800665/297930871 = 0.006043902043303193
rat: replaced 2.877927110806339e-4 by 1167733/4057548906 = 2.877927110806339e-4
rat: replaced 0.00719136414613375 by 2476362/344352191 = 0.007191364146133747
rat: replaced 3.658573803051457e-4 by 386279/1055818526 = 3.658573803051454e-4
rat: replaced 0.00843810628521191 by 2079855/246483622 = 0.008438106285211924
rat: replaced 4.5688535576352e-4 by 262978/575588595 = 4.568853557635206e-4
rat: replaced 0.009784003787362772 by 1752551/179124113 = 0.009784003787362787
rat: replaced 5.618675264007778e-4 by 150595/268025812 = 5.618675264007782e-4
```

rat: replaced 0.01122892206395776 by 5450241/485375263 = 0.01122892206395776

rat: replaced 6.817933857540259e-4 by 192316/282073725 = 6.817933857540258e-4

rat: replaced 0.01277271662437307 by 3258991/255152533 = 0.01277271662437308

rat: replaced 8.176509330039827e-4 by 105841/129445214 = 8.176509330039812e-4

rat: replaced 0.01441523309043924 by 2330472/161667313 = 0.01441523309043925

rat: replaced 9.704265741758145e-4 by 651321/671169790 = 9.704265741758132e-4

rat: replaced 0.01615630721187855 by 19391318/1200232067 = 0.01615630721187855

rat: replaced 0.001141105023499428 by 1259907/1104111343 = 0.001141105023499428

rat: replaced 0.01799576488272969 by 4765614/264818641 = 0.01799576488272969

rat: replaced 0.001330669204938795 by 1231154/925214167 = 0.001330669204938796

rat: replaced 0.01993342215875837 by 2504519/125644206 = 0.01993342215875836

rat: replaced 0.001540100153900437 by 276884/179783113 = 0.001540100153900439

rat: replaced 0.02196908527585173 by 1298306/59096953 = 0.0219690852758517

rat: replaced 0.001770376919130678 by 644389/363984072 = 0.001770376919130681

rat: replaced 0.02410255066939448 by 2001286/83032125 = 0.02410255066939453

rat: replaced 0.002022476464811601 by 1271955/628909667 = 0.002022476464811599

rat: replaced 0.02633360499462523 by 2978115/113091808 = 0.02633360499462525

rat: replaced 0.002297373572865413 by 1020913/444382669 = 0.002297373572865417  
rat: replaced 0.02866202514797045 by 1770713/61779061 = 0.02866202514797044  
rat: replaced 0.002596040745477063 by 1097643/422814242 = 0.002596040745477065  
rat: replaced 0.03108757828935527 by 5034207/161936287 = 0.03108757828935525  
rat: replaced 0.002919448107844891 by 906221/310408326 = 0.002919448107844891  
rat: replaced 0.03361002186548678 by 4553215/135471944 = 0.03361002186548678  
rat: replaced 0.00326856331168871 by 1379071/421919623 = 0.00326856331168867  
rat: replaced 0.03622910363410947 by 3082649/85087642 = 0.0362291036341094  
rat: replaced 0.003644351435886262 by 5966577/1637212301 = 0.003644351435886261  
rat: replaced 0.03894456168922911 by 4913415/126164342 = 0.03894456168922911  
rat: replaced 0.004047774895164447 by 572425/141417202 = 0.004047774895164451  
rat: replaced 0.04175612448730281 by 1734727/41544253 = 0.04175612448730273  
rat: replaced 0.004479793338660443 by 2952779/659132861 = 0.004479793338660444  
rat: replaced 0.04466351087439402 by 4691119/105032473 = 0.04466351087439405  
rat: replaced 0.0049413635565565 by 2524919/510976165 = 0.004941363556556498  
rat: replaced 0.04766643011428662 by 3536207/74186529 = 0.04766643011428665  
rat: replaced 0.005433439383882244 by 1361584/250593391 = 0.005433439383882235  
rat: replaced 0.05076458191755917 by 7710025/151878036 = 0.05076458191755916

rat: replaced 0.005956971605131645 by 1447422/242979503 = 0.005956971605131648  
rat: replaced 0.0539576564716131 by 3377975/62604183 = 0.05395765647161309  
rat: replaced 0.006512907859185624 by 3695063/567344584 = 0.006512907859185626  
rat: replaced 0.05724533447165381 by 2560865/44734912 = 0.05724533447165382  
rat: replaced 0.007102192544548636 by 1363981/192050693 = 0.007102192544548642  
rat: replaced 0.06062728715262111 by 8274761/136485754 = 0.06062728715262107  
rat: replaced 0.007725766724910044 by 1464384/189545459 = 0.007725766724910038  
rat: replaced 0.06410317632206519 by 5287663/82486755 = 0.06410317632206528  
rat: replaced 0.00838456803503801 by 1113589/132814117 = 0.008384568035038023  
rat: replaced 0.06767265439396564 by 2921400/43169579 = 0.06767265439396572  
rat: replaced 0.009079530587017326 by 433906/47789475 = 0.00907953058701733  
rat: replaced 0.07133536442348987 by 7236103/101437808 = 0.07133536442348991  
rat: replaced 0.009811584876838586 by 1363090/138926587 = 0.009811584876838586  
rat: replaced 0.07509094014268702 by 9209133/122639735 = 0.07509094014268704  
rat: replaced 0.0105816576913495 by 1163729/109976058 = 0.01058165769134951  
rat: replaced 0.07893900599711501 by 5197067/65836489 = 0.07893900599711506  
rat: replaced 0.01139067201557714 by 13426050/1178688139 = 0.01139067201557714

rat: replaced 0.08287917718339499 by  $11217158/135343501 = 0.082879177183395$   
rat: replaced 0.01223954694042984 by  $2283101/186534764 = 0.01223954694042983$   
rat: replaced 0.08691105968769186 by  $5213115/59982182 = 0.08691105968769192$   
rat: replaced 0.01312919757078923 by  $3499615/266552086 = 0.01312919757078922$   
rat: replaced 0.09103425032511492 by  $5893225/64736349 = 0.09103425032511488$   
rat: replaced 0.01406053493400045 by  $2280713/162206702 = 0.01406053493400045$   
rat: replaced 0.09524833678003664 by  $9601787/100807923 = 0.09524833678003662$   
rat: replaced 0.01503446588876983 by  $200490/13335359 = 0.01503446588876985$   
rat: replaced 0.09955289764732322 by  $5687088/57126293 = 0.09955289764732328$   
rat: replaced 0.01605189303448024 by  $951971/59305840 = 0.01605189303448025$   
rat: replaced 0.1039475024744748 by  $10260011/98703776 = 0.1039475024744747$   
rat: replaced 0.01711371462093175 by  $9432386/551159477 = 0.01711371462093176$   
rat: replaced 0.1084317118046711 by  $14939691/137779721 = 0.1084317118046712$   
rat: replaced 0.01822082445851714 by  $2559788/140486947 = 0.01822082445851713$   
rat: replaced 0.113005077220716 by  $8478529/75027859 = 0.1130050772207161$   
rat: replaced 0.01937411182884202 by  $2983799/154009589 = 0.01937411182884203$   
rat: replaced 0.1176671413898787 by  $7123715/60541243 = 0.1176671413898786$   
rat: replaced 0.02057446139579705 by  $7167743/348380590 = 0.02057446139579705$

```
rat: replaced 0.1224174381096274 by 12172179/99431741 = 0.1224174381096274
rat: replaced 0.02182275311709253 by 7415562/339808729 = 0.02182275311709253
rat: replaced 0.1272554923542488 by 7277933/57191504 = 0.127255492354249
rat: replaced 0.02311986215626333 by 2988661/129268115 = 0.02311986215626336
rat: replaced 0.1321808203223502 by 3633064/27485561 = 0.1321808203223503
rat: replaced 0.02446665879515308 by 1991976/81415939 = 0.02446665879515312
rat: replaced 0.1371929294852391 by 56235017/409897341 = 0.1371929294852391
rat: replaced 0.02586400834688696 by 5000736/193347293 = 0.02586400834688697
rat: replaced 0.1422913186361759 by 9349741/65708443 = 0.1422913186361759
rat: replaced 0.02731277106934082 by 858413/31428997 = 0.02731277106934084
rat: replaced 0.1474754779404944 by 1549881/10509415 = 0.1474754779404943
rat: replaced 0.02881380207911666 by 3754753/130310918 = 0.02881380207911666
rat: replaced 0.152744888986584 by 5264425/34465474 = 0.1527448889865841
rat: replaced 0.03036795126603076 by 4118329/135614318 = 0.03036795126603077
rat: replaced 0.1580990248377314 by 5442776/34426373 = 0.1580990248377312
rat: replaced 0.03197606320812652 by 3497683/109384416 = 0.03197606320812647
rat: replaced 0.1635373500848132 by 12328488/75386375 = 0.1635373500848131
```

rat: replaced 0.0336389770872163 by  $3971799/118071337 = 0.03363897708721635$   
rat: replaced 0.1690593208998367 by  $20896917/123607009 = 0.1690593208998367$   
rat: replaced 0.03535752660496472 by  $1815732/51353479 = 0.03535752660496478$   
rat: replaced 0.1746643850903219 by  $2841592/16268869 = 0.1746643850903219$   
rat: replaced 0.03713253989951881 by  $3333721/89778965 = 0.03713253989951878$   
rat: replaced 0.1803519821545206 by  $4461007/24735004 = 0.1803519821545208$   
rat: replaced 0.03896483946269502 by  $8785771/225479461 = 0.03896483946269501$   
rat: replaced 0.1861215433374662 by  $4381209/23539505 = 0.1861215433374661$   
rat: replaced 0.0408552420577305 by  $3189084/78058135 = 0.04085524205773043$   
rat: replaced 0.1919724916878484 by  $72809759/379271834 = 0.1919724916878484$   
rat: replaced 0.04280455863760801 by  $7646593/178639688 = 0.04280455863760801$   
rat: replaced 0.1979042421157076 by  $26318167/132984350 = 0.1979042421157076$   
rat: replaced 0.04481359426396048 by  $20610430/459914683 = 0.04481359426396048$   
rat: replaced 0.2039162014509444 by  $8519416/41779005 = 0.2039162014509441$   
rat: replaced 0.04688314802656623 by  $3439140/73355569 = 0.04688314802656633$   
rat: replaced 0.2100077685026351 by  $50962787/242670961 = 0.2100077685026351$   
rat: replaced 0.04901401296344043 by  $4006732/81746663 = 0.04901401296344048$   
rat: replaced 0.216178334119151 by  $1347531/6233423 = 0.2161783341191509$

rat: replaced 0.05120697598153157 by 4148974/81023609 = 0.0512069759815315

rat: replaced 0.2224272812490723 by 23234851/104460437 = 0.2224272812490723

rat: replaced 0.05346281777803219 by 11998448/224426031 = 0.05346281777803218

rat: replaced 0.2287539850028937 by 8185268/35781969 = 0.2287539850028935

rat: replaced 0.05578231276230905 by 1398019/25062048 = 0.05578231276230897

rat: replaced 0.2351578127155118 by 12642104/53760085 = 0.2351578127155119

rat: replaced 0.05816622897846346 by 4451048/76522891 = 0.05816622897846345

rat: replaced 0.2416381240094921 by 8002142/33116223 = 0.2416381240094923

rat: replaced 0.06061532802852698 by 2146337/35409146 = 0.06061532802852686

rat: replaced 0.2481942708591053 by 8882901/35790113 = 0.2481942708591057

rat: replaced 0.0631303649963022 by 14651447/232082406 = 0.06313036499630222

rat: replaced 0.2548255976551299 by 868346/3407609 = 0.25482559765513

rat: replaced 0.06571208837185505 by 4240309/64528599 = 0.06571208837185509

rat: replaced 0.2615314412704124 by 8212450/31401387 = 0.2615314412704127

rat: replaced 0.06836123997666599 by 2716643/39739522 = 0.06836123997666604

rat: replaced 0.2683111311261794 by 34459769/128432126 = 0.2683111311261794

rat: replaced 0.07107855488944881 by 3146673/44270357 = 0.07107855488944893

rat: replaced 0.2751639892590951 by  $12552159/45617012 = 0.2751639892590949$   
rat: replaced 0.07386476137264342 by  $12898997/174629915 = 0.0738647613726434$   
rat: replaced 0.2820893303890569 by  $11134456/39471383 = 0.2820893303890568$   
rat: replaced 0.07672058079958999 by  $5073506/66129661 = 0.07672058079959007$   
rat: replaced 0.2890864619877229 by  $9583357/33150487 = 0.2890864619877228$   
rat: replaced 0.07964672758239233 by  $5672399/71219486 = 0.07964672758239227$   
rat: replaced 0.2961546843477643 by  $11052271/37319251 = 0.2961546843477647$   
rat: replaced 0.08264390910047736 by  $4686067/56701904 = 0.08264390910047748$   
rat: replaced 0.3032932906528349 by  $9918077/32701274 = 0.3032932906528351$   
rat: replaced 0.0857128256298576 by  $3585977/41837111 = 0.08571282562985766$   
rat: replaced 0.3105015670482534 by  $9320011/30015987 = 0.3105015670482533$   
rat: replaced 0.08885417027310427 by  $5751353/64728003 = 0.0888541702731042$   
rat: replaced 0.3177787927123868 by  $248395525/781661743 = 0.3177787927123868$   
rat: replaced 0.09206862889003742 by  $7305460/79347983 = 0.09206862889003745$   
rat: replaced 0.3251242399287333 by  $13842845/42577093 = 0.3251242399287335$   
rat: replaced 0.09535688002914089 by  $5971998/62627867 = 0.09535688002914103$   
rat: replaced 0.3325371741586922 by  $9318229/28021616 = 0.3325371741586923$   
rat: replaced 0.0987195948597075 by  $9821211/99485933 = 0.09871959485970745$

```
rat: replaced 0.3400168541150183 by 13391981/39386227 = 0.3400168541150184
rat: replaced 0.1021574371047232 by 8336413/81603584 = 0.1021574371047232
rat: replaced 0.3475625318359485 by 10097818/29053241 = 0.347562531835949
rat: replaced 0.1056710629744951 by 5741011/54329074 = 0.105671062974495
rat: replaced 0.3551734527599992 by 15867851/44676343 = 0.3551734527599987
rat: replaced 0.1092611211010309 by 5551873/50812887 = 0.1092611211010309
rat: replaced 0.3628488558014202 by 6897641/19009681 = 0.3628488558014203
rat: replaced 0.1129282524731764 by 11548693/102265755 = 0.1129282524731764
rat: replaced 0.3705879734263036 by 23358661/63031352 = 0.3705879734263038
rat: replaced 0.1166730903725168 by 5656228/48479285 = 0.1166730903725168
rat: replaced 0.3783900317293359 by 14241382/37636779 = 0.3783900317293358
rat: replaced 0.1204962603100498 by 4057613/33674182 = 0.12049626031005
rat: replaced 0.3862542505111889 by 3461217/8960981 = 0.3862542505111884
rat: replaced 0.1243983799636342 by 7966447/64039797 = 0.1243983799636342
rat: replaced 0.3941798433565377 by 5314214/13481699 = 0.3941798433565384
rat: replaced 0.1283800591162231 by 796346/6203035 = 0.1283800591162229
rat: replaced 0.4021660177127022 by 11567173/28762184 = 0.4021660177127022
```

```
rat: replaced 0.1324418995948859 by 4716124/35609003 = 0.1324418995948862
rat: replaced 0.4102119749689023 by 11320633/27597032 = 0.4102119749689024
rat: replaced 0.1365844952106265 by 612971/4487852 = 0.1365844952106264
rat: replaced 0.418316910536117 by 12225195/29224721 = 0.4183169105361177
rat: replaced 0.140808431699002 by 10431632/74083859 = 0.1408084316990021
rat: replaced 0.4264800139275439 by 7978696/18708253 = 0.4264800139275431
rat: replaced 0.1451142866615502 by 3554077/24491572 = 0.1451142866615504
rat: replaced 0.4347004688396462 by 20489554/47134879 = 0.4347004688396463
rat: replaced 0.1495026295080298 by 26759297/178988805 = 0.1495026295080298
rat: replaced 0.4429774532337832 by 23449796/52936771 = 0.4429774532337834
rat: replaced 0.1539740213994798 by 16145763/104860306 = 0.1539740213994798
rat: replaced 0.451310139418413 by 8841241/19590167 = 0.4513101394184133
part: invalid index of list or matrix.
#0: lineIntersection(g=[2,-2,0],h=[3-sqrt(10)/sqrt(2),1+sqrt(10)/sqrt(2),(5-sqrt(10)/sqrt(2))*(1+
-- an error. To debug this try: debugmode(true);

Error in:
... ection(angleBisector(A,C,B),angleBisector(C,B,A)); $P // titik ...
```

```
>P() //
```

0

## Garis dan Lingkaran yang berpotongan

---

```
>A &:= [2,0]; c=circleWithCenter(A,4);
>B &:= [2,3]; C &:= [3,2]; l=lineThrough(B,C);
>setPlotRange(5); plotCircle(c); plotLine(l);
>{P1,P2,f}=lineCircleIntersections(l,c);
>P1, P2,
```

```
[5.89792, -0.897916]
[1.10208, 3.89792]
```

```
>plotPoint(P1); plotPoint(P2):
```

```
>c &= circleWithCenter(A,4) // lingkaran dengan pusat A jari-jari 4
```

```
[2, 0, 4]
```

```
>l &= lineThrough(B,C) // garis l melalui B dan C
```

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

```
>$lineCircleIntersections(l,c) | radcan, // titik potong lingkaran c dan garis l
```

Maxima said:

```
rat: replaced -4.98329175014009e-5 by -86001/1725786976 = -4.983291750140082e-5

rat: replaced -1.986600267553235e-4 by -1133306/5704751069 = -1.986600267553234e-4

rat: replaced -4.454664535081185e-4 by -474290/1064704191 = -4.454664535081181e-4

rat: replaced -7.892275256562442e-4 by -1190199/1508055613 = -7.892275256562439e-4

rat: replaced -0.001228908875712045 by -259907/211494119 = -0.001228908875712047

rat: replaced -0.001763466544240408 by -5854594/3319934829 = -0.001763466544240408

rat: replaced -0.002391847084253176 by -866601/362314550 = -0.002391847084253172

rat: replaced -0.003112987666553255 by -5049204/1621980085 = -0.003112987666553255

rat: replaced -0.00392581618601677 by -1241039/316122544 = -0.003925816186016774

rat: replaced -0.004829251368802329 by -3015690/624463249 = -0.00482925136880233

rat: replaced -0.005822202880477995 by -2532373/434951006 = -0.005822202880477991
```

rat: replaced -0.006903571435053116 by  $-1331361/192851050 = -0.006903571435053115$   
rat: replaced -0.008072248904906765 by  $-7953293/985263598 = -0.008072248904906766$   
rat: replaced -0.009327118431599252 by  $-432515/46371771 = -0.009327118431599259$   
rat: replaced -0.01066705453755698 by  $-2950074/276559381 = -0.01066705453755698$   
rat: replaced -0.01209092323861904 by  $-1254816/103781653 = -0.01209092323861907$   
rat: replaced -0.01359758215743526 by  $-1827823/134422648 = -0.01359758215743526$   
rat: replaced -0.01518588063770274 by  $-9199276/605778237 = -0.01518588063770274$   
rat: replaced -0.01685465985923026 by  $-2516580/149310637 = -0.01685465985923026$   
rat: replaced -0.01860275295381958 by  $-2032371/109251088 = -0.01860275295381955$   
rat: replaced -0.02042898512195129 by  $-1413911/69211025 = -0.02042898512195131$   
rat: replaced -0.02233217375026381 by  $-3647892/163346929 = -0.02233217375026377$   
rat: replaced -0.02431112852981362 by  $-1377268/56651751 = -0.02431112852981367$   
rat: replaced -0.02636465157510504 by  $-2533336/96088355 = -0.02636465157510502$   
rat: replaced -0.0284915375438782 by  $-9699307/340427644 = -0.02849153754387819$   
rat: replaced -0.03069057375764189 by  $-7938451/258660886 = -0.03069057375764189$   
rat: replaced -0.0329605403229406 by  $-2936449/89089832 = -0.03296054032294056$   
rat: replaced -0.03530021025334285 by  $-5224432/148000025 = -0.03530021025334287$

rat: replaced  $-0.03770834959213837$  by  $-2448749/64939172 = -0.03770834959213832$   
rat: replaced  $-0.04018371753573358$  by  $-2461511/61256428 = -0.04018371753573356$   
rat: replaced  $-0.04272506655773012$  by  $-13954421/326609696 = -0.04272506655773012$   
rat: replaced  $-0.04533114253367693$  by  $-2051558/45257143 = -0.04533114253367695$   
rat: replaced  $-0.04800068486648146$  by  $-16995415/354066094 = -0.04800068486648145$   
rat: replaced  $-0.05073242661246818$  by  $-3970295/78259513 = -0.05073242661246818$   
rat: replaced  $-0.05352509460807248$  by  $-3894269/72755948 = -0.05352509460807246$   
rat: replaced  $-0.05637740959715515$  by  $-11093364/196769665 = -0.05637740959715513$   
rat: replaced  $-0.05928808635892763$  by  $-3489209/58851773 = -0.05928808635892754$   
rat: replaced  $-0.06225583383647254$  by  $-3380435/54299088 = -0.06225583383647254$   
rat: replaced  $-0.06527935526584844$  by  $-7571267/115982564 = -0.06527935526584841$   
rat: replaced  $-0.06835734830576551$  by  $-8050241/117767017 = -0.06835734830576544$   
rat: replaced  $-0.07148850516781785$  by  $-5513427/77123266 = -0.07148850516781798$   
rat: replaced  $-0.07467151274726203$  by  $-2259975/30265558 = -0.07467151274726208$   
rat: replaced  $-0.07790505275432569$  by  $-657797/8443573 = -0.07790505275432569$   
rat: replaced  $-0.08118780184603619$  by  $-4832180/59518547 = -0.08118780184603633$   
rat: replaced  $-0.08451843175855339$  by  $-3076049/36395008 = -0.08451843175855327$   
rat: replaced  $-0.08789560943999458$  by  $-7150621/81353563 = -0.08789560943999465$

rat: replaced  $-0.0913179971837394$  by  $-20067867/219758072 = -0.0913179971837394$

rat: replaced  $-0.09478425276219882$  by  $-5487749/57897265 = -0.09478425276219869$

rat: replaced  $-0.09829302956103664$  by  $-4406725/44832528 = -0.09829302956103658$

rat: replaced  $-0.1018429767138303$  by  $-3912367/38415678 = -0.1018429767138302$

rat: replaced  $-0.1054327392371563$  by  $-8451941/80164293 = -0.1054327392371564$

rat: replaced  $-0.1090609581660869$  by  $-13126833/120362348 = -0.1090609581660869$

rat: replaced  $-0.112726270690086$  by  $-2754747/24437489 = -0.112726270690086$

rat: replaced  $-0.116427310289289$  by  $-22239618/191017193 = -0.116427310289289$

rat: replaced  $-0.1201627068711536$  by  $-9494831/79016454 = -0.1201627068711537$

rat: replaced  $-0.1239310869074673$  by  $-3190398/25743323 = -0.1239310869074672$

rat: replaced  $-0.1277310735717007$  by  $-15999330/125257931 = -0.1277310735717006$

rat: replaced  $-0.1315612868766867$  by  $-13929723/105880106 = -0.1315612868766867$

rat: replaced  $-0.1354203438126204$  by  $-28035370/207024803 = -0.1354203438126204$

rat: replaced  $-0.1393068584853572$  by  $-11590983/83204683 = -0.1393068584853571$

rat: replaced  $-0.1432194422550018$  by  $-12738764/88945773 = -0.1432194422550018$

rat: replaced  $-0.1471567038747712$  by  $-5246589/35653075 = -0.147156703874771$

rat: replaced  $-0.1511172496301179$  by  $-4676629/30947023 = -0.1511172496301179$

rat: replaced  $-0.1550996834780995$  by  $-15854305/102220099 = -0.1550996834780995$

rat: replaced  $-0.1591026071869839$  by  $-9026555/56734174 = -0.159102607186984$

rat: replaced  $-0.1631246204760689$  by  $-10435073/63969945 = -0.1631246204760689$

rat: replaced  $-0.1671643211557106$  by  $-164873401/986295400 = -0.1671643211557106$

rat: replaced  $-0.1712203052675407$  by  $-7017638/40986015 = -0.1712203052675406$

rat: replaced  $-0.1752911672248615$  by  $-3184915/18169284 = -0.1752911672248615$

rat: replaced  $-0.1793754999532028$  by  $-2646709/14755131 = -0.1793754999532027$

rat: replaced  $-0.1834718950310287$  by  $-8392143/45740755 = -0.1834718950310287$

rat: replaced  $-0.1875789428305783$  by  $-12888313/68708741 = -0.1875789428305781$

rat: replaced  $-0.1916952326588277$  by  $-16014703/83542521 = -0.1916952326588277$

rat: replaced  $-0.1958193528985573$  by  $-21279927/108671215 = -0.1958193528985574$

rat: replaced  $-0.1999498911495134$  by  $-5994245/29978736 = -0.1999498911495134$

rat: replaced  $-0.2040854343696463$  by  $-17847769/87452439 = -0.2040854343696464$

rat: replaced  $-0.2082245690164135$  by  $-5203892/24991729 = -0.2082245690164134$

rat: replaced  $-0.2123658811881329$  by  $-20393053/96027916 = -0.2123658811881328$

rat: replaced  $-0.2165079567653719$  by  $-8489188/39209589 = -0.2165079567653719$

rat: replaced  $-0.2206493815523576$  by  $-14881929/67446049 = -0.2206493815523575$

rat: replaced  $-0.2247887414183958$  by  $-11437558/50881365 = -0.2247887414183955$

rat: replaced  $-0.2289246224392826$  by  $-17547464/76651711 = -0.2289246224392825$   
rat: replaced  $-0.2330556110386959$  by  $-11148764/47837355 = -0.2330556110386956$   
rat: replaced  $-0.2371802941295513$  by  $-11052217/46598378 = -0.237180294129551$   
rat: replaced  $-0.2412972592553108$  by  $-36037383/149348497 = -0.2412972592553108$   
rat: replaced  $-0.2454050947312253$  by  $-4652365/18957899 = -0.2454050947312252$   
rat: replaced  $-0.2495023897855041$  by  $-6175634/24751803 = -0.2495023897855037$   
rat: replaced  $-0.2535877347003893$  by  $-11299519/44558618 = -0.2535877347003895$   
rat: replaced  $-0.2576597209531272$  by  $-6871877/26670358 = -0.2576597209531271$   
rat: replaced  $-0.2617169413568191$  by  $-2245730/8580759 = -0.2617169413568194$   
rat: replaced  $-0.2657579902011391$  by  $-10500993/39513367 = -0.2657579902011388$   
rat: replaced  $-0.2697814633929034$  by  $-21050552/78028163 = -0.2697814633929034$   
rat: replaced  $-0.2737859585964791$  by  $-1510231/5516101 = -0.2737859585964796$   
rat: replaced  $-0.2777700753740163$  by  $-9819093/35349715 = -0.2777700753740164$   
rat: replaced  $-0.2817324153254904$  by  $-10837378/38466919 = -0.2817324153254905$   
rat: replaced  $-0.2856715822285418$  by  $-17041418/59653879 = -0.2856715822285421$   
rat: replaced  $-0.289586182178096$  by  $-721506/2491507 = -0.2895861821780955$   
rat: replaced  $-0.2934748237257534$  by  $-11793110/40184401 = -0.2934748237257537$

```
rat: replaced -0.2973361180189332 by -15390047/51759763 = -0.2973361180189329
rat: replaced 5.016624916807239e-5 by 153117/3052191514 = 5.016624916807235e-5
rat: replaced 2.013266400891639e-4 by 232411/1154397649 = 2.013266400891639e-4
rat: replaced 4.544660485167953e-4 by 444871/978887205 = 4.544660485167952e-4
rat: replaced 8.105591523879241e-4 by 1425236/1758336817 = 8.105591523879239e-4
rat: replaced 0.001270570334355389 by 696221/547959433 = 0.00127057033435539
rat: replaced 0.001835453585351213 by 1018402/554850315 = 0.001835453585351213
rat: replaced 0.002506152409187654 by 484773/193433168 = 0.002506152409187653
rat: replaced 0.003283599728207867 by 1007483/306822720 = 0.003283599728207872
rat: replaced 0.004168717789994683 by 897113/215201183 = 0.004168717789994677
rat: replaced 0.00516241807514603 by 757433/146720585 = 0.005162418075146034
rat: replaced 0.006265601206128374 by 1194190/190594639 = 0.006265601206128363
rat: replaced 0.007479156857214384 by 1971251/263565939 = 0.007479156857214391
rat: replaced 0.008803963665517056 by 365844/41554465 = 0.008803963665517051
rat: replaced 0.01024088914312629 by 1345773/131411734 = 0.01024088914312629
rat: replaced 0.01179078959035854 by 1519715/128890011 = 0.01179078959035856
rat: replaced 0.0134545100101271 by 2242921/166704027 = 0.01345451001012711
rat: replaced 0.01523288402344322 by 1950407/128039247 = 0.01523288402344322
```

rat: replaced 0.01712673378605437 by 1362867/79575418 = 0.01712673378605438  
rat: replaced 0.01913686990622912 by 1694449/88543686 = 0.01913686990622911  
rat: replaced 0.02126409136369717 by 9814128/461535263 = 0.02126409136369716  
rat: replaced 0.02350918542975217 by 2315819/98506986 = 0.02350918542975216  
rat: replaced 0.02587292758852516 by 3386321/130882792 = 0.02587292758852516  
rat: replaced 0.02835608145943683 by 10230271/360778728 = 0.02835608145943682  
rat: replaced 0.03095939872083586 by 14307719/462144602 = 0.03095939872083587  
rat: replaced 0.03368361903483233 by 4712088/139892569 = 0.03368361903483236  
rat: replaced 0.03652946997333167 by 4111522/112553563 = 0.03652946997333172  
rat: replaced 0.03949766694527834 by 8626745/218411508 = 0.03949766694527836  
rat: replaced 0.04258891312511537 by 3115258/73147159 = 0.04258891312511536  
rat: replaced 0.04580389938246726 by 2358579/51492974 = 0.04580389938246721  
rat: replaced 0.04914330421305446 by 2180747/44375262 = 0.04914330421305456  
rat: replaced 0.05260779367084312 by 4975224/94571995 = 0.05260779367084304  
rat: replaced 0.05619802130144141 by 1396735/24853811 = 0.05619802130144146  
rat: replaced 0.05991462807674475 by 6603037/110207427 = 0.05991462807674477  
rat: replaced 0.06375824233083943 by 6198842/97224167 = 0.0637582423308394

rat: replaced 0.06772947969716975 by 4012504/59243095 = 0.06772947969716978  
rat: replaced 0.07182894304697524 by 5813372/80933559 = 0.07182894304697511  
rat: replaced 0.07605722242900365 by 14672328/192911699 = 0.07605722242900365  
rat: replaced 0.08041489501050719 by 3507279/43614793 = 0.0804148950105071  
rat: replaced 0.08490252501952561 by 2460362/28978667 = 0.08490252501952557  
rat: replaced 0.08952066368846451 by 4304415/48082921 = 0.08952066368846436  
rat: replaced 0.09426984919897213 by 3898288/41352437 = 0.09426984919897224  
rat: replaced 0.0991506066281217 by 11428253/115261554 = 0.09915060662812164  
rat: replaced 0.1041634478959041 by 7209817/69216382 = 0.1041634478959042  
rat: replaced 0.1093088717140371 by 3826731/35008421 = 0.109308871714037  
rat: replaced 0.1145873635360931 by 5173172/45146095 = 0.1145873635360932  
rat: replaced 0.1199993955089551 by 23218093/193485083 = 0.1199993955089551  
rat: replaced 0.1255454264256029 by 2445819/19481546 = 0.125545426425603  
rat: replaced 0.1312259016792331 by 9111136/69430927 = 0.131225901679233  
rat: replaced 0.1370412532187207 by 16597683/121114501 = 0.1370412532187207  
rat: replaced 0.1429918995054244 by 34253454/239548213 = 0.1429918995054244  
rat: replaced 0.1490782454713414 by 11997679/80479073 = 0.1490782454713414  
rat: replaced 0.1553006824786136 by 13065213/84128497 = 0.1553006824786136

```
rat: replaced 0.1616595882803922 by 12686167/78474572 = 0.1616595882803923
rat: replaced 0.1681553269830629 by 4527449/26924208 = 0.168155326983063
rat: replaced 0.1747882490098353 by 23565700/134824281 = 0.1747882490098353
rat: replaced 0.1815586910657007 by 4563713/25136296 = 0.1815586910657004
rat: replaced 0.1884669761037622 by 8213146/43578701 = 0.1884669761037623
rat: replaced 0.1955134132929397 by 7172626/36686107 = 0.1955134132929395
rat: replaced 0.202698297987053 by 17668607/87167022 = 0.2026982979870529
rat: replaced 0.2100219116952866 by 8269584/39374863 = 0.2100219116952864
rat: replaced 0.2174845220540395 by 56596301/260231397 = 0.2174845220540395
rat: replaced 0.2250863828001612 by 8187128/36373271 = 0.2250863828001611
rat: replaced 0.2328277337455789 by 10320856/44328293 = 0.2328277337455787
rat: replaced 0.2407088007533156 by 16964872/70478819 = 0.2407088007533157
rat: replaced 0.2487297957149048 by 11063220/44478869 = 0.2487297957149045
rat: replaced 0.2568909165292014 by 17200949/66958183 = 0.2568909165292015
rat: replaced 0.2651923470825914 by 8866093/33432688 = 0.2651923470825918
rat: replaced 0.2736342572306039 by 12664159/46281336 = 0.2736342572306037
rat: replaced 0.2822168027809259 by 8116045/28758192 = 0.2822168027809259
```

```
rat: replaced 0.2909401254778209 by 24764749/85119744 = 0.290940125477821
rat: replaced 0.2998043529879556 by 28498628/95057419 = 0.2998043529879556
rat: replaced 0.3088095988876323 by 13390352/43361191 = 0.308809598887632
rat: replaced 0.3179559626514321 by 26241235/82531036 = 0.3179559626514321
rat: replaced 0.3272435296422674 by 8247573/25203166 = 0.3272435296422679
rat: replaced 0.3366723711028454 by 10805861/32096073 = 0.3366723711028449
rat: replaced 0.3462425441485439 by 20967050/60555961 = 0.3462425441485438
rat: replaced 0.3559540917617003 by 19053013/53526602 = 0.3559540917617001
rat: replaced 0.3658070427873129 by 10401097/28433288 = 0.3658070427873132
rat: replaced 0.3758014119301566 by 5923743/15762961 = 0.375801411930157
rat: replaced 0.3859371997533123 by 2934328/7603123 = 0.3859371997533119
rat: replaced 0.396214392678111 by 30414315/76762267 = 0.396214392678111
rat: replaced 0.4066329629854911 by 13711485/33719561 = 0.4066329629854908
rat: replaced 0.4171928688187707 by 20838614/49949593 = 0.4171928688187709
rat: replaced 0.4278940541878331 by 16106690/37641771 = 0.427894054187833
rat: replaced 0.4387364489747257 by 4869080/11097961 = 0.4387364489747261
rat: replaced 0.4497199689406718 by 4550581/10118699 = 0.4497199689406711
rat: replaced 0.4608445157344944 by 7970699/17295853 = 0.4608445157344943
```

```
rat: replaced 0.4721099769024512 by 25424083/53852035 = 0.4721099769024513
rat: replaced 0.48351622589948 by 17675673/36556525 = 0.4835162258994803
rat: replaced 0.4950631221018528 by 7053395/14247466 = 0.495063122101853
rat: replaced 0.5067505108212387 by 13754758/27143057 = 0.5067505108212388
rat: replaced 0.5185782233201719 by 21662467/41772805 = 0.518578223320172
rat: replaced 0.5305460768289253 by 10488897/19770002 = 0.530546076828925
rat: replaced 0.5426538745637882 by 22388393/41257225 = 0.5426538745637886
rat: replaced 0.5549014057467435 by 9960301/17949677 = 0.5549014057467441
rat: replaced 0.5672884456265459 by 28078535/49496046 = 0.5672884456265456
rat: replaced 0.5798147555011964 by 18086313/31193261 = 0.5798147555011962
rat: replaced 0.5924800827418131 by 20592707/34756792 = 0.5924800827418134
rat: replaced 0.6052841608178928 by 26813845/44299598 = 0.6052841608178927
part: invalid index of list or matrix.
#0: lineIntersection(g=[1,-1,2],h=[1,1,5])
#1: projectToLine(a=[2,0],g=[1,1,5])
#2: lineCircleIntersections(g=[1,1,5],c=[2,0,4])
-- an error. To debug this try: debugmode(true);

Error in:
$lineCircleIntersections(l,c) | radcan, // titik potong lingka ...
^
```

```
>C=A+normalize([-3,-4])*4; plotPoint(C); plotSegment(P1,C); plotSegment(P2,C);
>degprint(computeAngle(P1,C,P2))
```

57°58'20.06'',

```
>C=A+normalize([-4,-5])*4; plotPoint(C); plotSegment(P1,C); plotSegment(P2,C);
>degprint(computeAngle(P1,C,P2))
```

57°58'20.06'',

```
>insimg;
```

**Garis Sumbu**

---

```
>A=[3,3]; B=[-2,-3];
>c1=circleWithCenter(A,distance(A,B));
>c2=circleWithCenter(B,distance(A,B));
>{P1,P2,f}=circleCircleIntersections(c1,c2);
>l=lineThrough(P1,P2);
>setPlotRange(5); plotCircle(c1); plotCircle(c2);
>plotPoint(A); plotPoint(B); plotSegment(A,B); plotLine(l);
```

```
>A &= [a1,a2]; B &= [b1,b2];
>c1 &= circleWithCenter(A,distance(A,B));
>c2 &= circleWithCenter(B,distance(A,B));
>P &= circleCircleIntersections(c1,c2); P1 &= P[1]; P2 &= P[2];
>g &= getLineEquation(lineThrough(P1,P2),x,y);
>$solve(g,y)
```

Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);

Error in:  
\$solve(g,y) ...  
^

```
>$solve(getLineEquation(middlePerpendicular(A,B),x,y),y)
```

Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);

Error in:  
... (getLineEquation(middlePerpendicular(A,B),x,y),y) ...  
^

```
>h &= getLineEquation(lineThrough(A,B),x,y);
>$solve(h,y)
```

```
Maxima said:  
solve: all variables must not be numbers.  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
$solve(h,y) ...  
^
```

## Garis Euler dan Parabola

---

```
>A:=[-1.5,-1.5]; B:=[3,0]; C:=[1.5,3];  
>setPlotRange(3); plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C");
```

```
>plotSegment(A,B,""); plotSegment(B,C,""); plotSegment(C,A,"");  
>$areaTriangle(A,B,C)
```

```
>c &= lineThrough(A,B)
```

$$\begin{bmatrix} 3 & 9 & 9 \\ - & -, & - \\ 2 & 2 & 2 \end{bmatrix}$$

```
>$getLineEquation(c,x,y)
>$getHesseForm(c,x,y,C), $at(%,[x=C[1],y=C[2]])
```

```
>LL &= circleThrough(A,B,C); $getCircleEquation(LL,x,y)
```

```
Maxima said:  
rat: replaced -7.57493712521158e-5 by -291512/3848375177 = -7.574937125211583e-5  
  
rat: replaced -3.059898801345065e-4 by -367004/1199399143 = -3.059898801345067e-4  
  
rat: replaced -6.951984652882083e-4 by -649868/934794929 = -6.951984652882086e-4  
  
rat: replaced -0.001247836168679406 by -996993/798977482 = -0.001247836168679407  
  
rat: replaced -0.0019683476894981 by -1171852/595348071 = -0.001968347689498099
```

rat: replaced  $-0.002861160939693026$  by  $-414045/144712237 = -0.002861160939693027$   
rat: replaced  $-0.003930686601183196$  by  $-1414939/359972479 = -0.003930686601183198$   
rat: replaced  $-0.00518131768479372$  by  $-1585327/305969851 = -0.005181317684793722$   
rat: replaced  $-0.006617429090958894$  by  $-2655242/401249785 = -0.00661742909095889$   
rat: replaced  $-0.008243377172234598$  by  $-1494085/181246711 = -0.00824337717223459$   
rat: replaced  $-0.01006349929766813$  by  $-2785964/276838495 = -0.01006349929766812$   
rat: replaced  $-0.01208211341906348$  by  $-403273/33377687 = -0.01208211341906346$   
rat: replaced  $-0.01430351763919102$  by  $-2688199/187939713 = -0.01430351763919103$   
rat: replaced  $-0.01673198978198$  by  $-3399597/203179481 = -0.01673198978198$   
rat: replaced  $-0.01937178696474014$  by  $-4095384/211409717 = -0.01937178696474013$   
rat: replaced  $-0.02222714517245272$  by  $-1488848/66983321 = -0.02222714517245271$   
rat: replaced  $-0.02530227883417678$  by  $-11141142/440321683 = -0.02530227883417678$   
rat: replaced  $-0.02860138040160899$  by  $-5896067/206146239 = -0.02860138040160898$   
rat: replaced  $-0.03212861992984196$  by  $-3474579/108145915 = -0.03212861992984201$   
rat: replaced  $-0.03588814466036214$  by  $-2498277/69612877 = -0.03588814466036219$   
rat: replaced  $-0.03988407860632956$  by  $-5523906/138499025 = -0.03988407860632954$   
rat: replaced  $-0.04412052214017978$  by  $-4053557/91874638 = -0.04412052214017975$   
rat: replaced  $-0.04860155158359004$  by  $-3943740/81144323 = -0.04860155158359014$

rat: replaced -0.05333121879985003 by -1834427/34396870 = -0.0533312187998501  
rat: replaced -0.05831355078867968 by -2465946/42287701 = -0.05831355078867967  
rat: replaced -0.06355254928353218 by -4583196/72116635 = -0.06355254928353216  
rat: replaced -0.06905219035142413 by -9155887/132593723 = -0.0690521903514241  
rat: replaced -0.07481642399533184 by -2967077/39658097 = -0.0748164239953319  
rat: replaced -0.08084917375919423 by -6800433/84112585 = -0.0808491737591943  
rat: replaced -0.08715433633556302 by -5843645/67049389 = -0.08715433633556303  
rat: replaced -0.09373578117593417 by -7402616/78973215 = -0.09373578117593415  
rat: replaced -0.1005973501038089 by -1640864/16311205 = -0.100597350103809  
rat: replaced -0.1077428569305121 by -20150833/187027090 = -0.107742856930512  
rat: replaced -0.115176087073816 by -3594765/31211036 = -0.1151760870738158  
rat: replaced -0.1229007971794005 by -4862770/39566627 = -0.1229007971794007  
rat: replaced -0.130920714745193 by -4199712/32078285 = -0.1309207147451929  
rat: replaced -0.1392395377486195 by -36213847/260083074 = -0.1392395377486195  
rat: replaced -0.1478609342768128 by -4198057/28391928 = -0.1478609342768128  
rat: replaced -0.1567885421598042 by -14899832/95031383 = -0.1567885421598042  
rat: replaced -0.1660259686067453 by -12607897/75939307 = -0.1660259686067454

rat: replaced  $-0.1755767898451896$  by  $-9911603/56451670 = -0.1755767898451897$   
rat: replaced  $-0.185444550763472$  by  $-7194550/38796233 = -0.1854445507634723$   
rat: replaced  $-0.1956327645562239$  by  $-14925693/76294444 = -0.1956327645562238$   
rat: replaced  $-0.206144912373057$  by  $-10508817/50977814 = -0.206144912373057$   
rat: replaced  $-0.2169844429704491$  by  $-5288053/24370655 = -0.2169844429704495$   
rat: replaced  $-0.2281547723668733$  by  $-3759235/16476688 = -0.2281547723668737$   
rat: replaced  $-0.2396592835011996$  by  $-18130307/75650343 = -0.2396592835011997$   
rat: replaced  $-0.2515013258944011$  by  $-9665078/38429531 = -0.2515013258944014$   
rat: replaced  $-0.2636842153146071$  by  $-16839380/63861919 = -0.2636842153146071$   
rat: replaced  $-0.2762112334455278$  by  $-9903377/35854360 = -0.2762112334455279$   
rat: replaced  $-0.2890856275582896$  by  $-4178583/14454482 = -0.2890856275582895$   
rat: replaced  $-0.3023106101867103$  by  $-5206455/17222204 = -0.3023106101867101$   
rat: replaced  $-0.3158893588060475$  by  $-15779177/49951594 = -0.3158893588060473$   
rat: replaced  $-0.3298250155152552$  by  $-20176073/61172052 = -0.3298250155152552$   
rat: replaced  $-0.3441206867227753$  by  $-22215819/64558220 = -0.3441206867227752$   
rat: replaced  $-0.358779442835901$  by  $-40621537/113221473 = -0.3587794428359009$   
rat: replaced  $-0.3738043179537355$  by  $-4462655/11938479 = -0.373804317953736$   
rat: replaced  $-0.3891983095637891$  by  $-17279077/44396588 = -0.389198309563789$

rat: replaced  $-0.4049643782422284$  by  $-15521239/38327418 = -0.4049643782422286$   
rat: replaced  $-0.4211054473578241$  by  $-18145515/43090193 = -0.4211054473578246$   
rat: replaced  $-0.4376244027796156$  by  $-12318025/28147482 = -0.4376244027796163$   
rat: replaced  $-0.4545240925883269$  by  $-18977389/41752218 = -0.4545240925883267$   
rat: replaced  $-0.4718073267915598$  by  $-11534269/24446990 = -0.47180732679156$   
rat: replaced  $-0.4894768770427974$  by  $-19580265/40002431 = -0.4894768770427977$   
rat: replaced  $-0.5075354763642387$  by  $-14211341/28000685 = -0.5075354763642389$   
rat: replaced  $-0.5259858188735007$  by  $-33496033/63682388 = -0.5259858188735008$   
rat: replaced  $-0.5448305595142084$  by  $-33841376/62113579 = -0.5448305595142087$   
rat: replaced  $-0.5640723137905005$  by  $-15307610/27137673 = -0.5640723137905007$   
rat: replaced  $-0.5837136575054853$  by  $-47878079/82023229 = -0.5837136575054854$   
rat: replaced  $-0.6037571265036585$  by  $-12602624/20873665 = -0.6037571265036591$   
rat: replaced  $-0.6242052164173237$  by  $-10481453/16791678 = -0.624205216417323$   
rat: replaced  $-0.6450603824170293$  by  $-7607359/11793251 = -0.6450603824170282$   
rat: replaced  $-0.6663250389660548$  by  $-21098582/31664099 = -0.6663250389660542$   
rat: replaced  $-0.6880015595789664$  by  $-31067245/45155777 = -0.6880015595789659$   
rat: replaced  $-0.7100922765842661$  by  $-5131430/7226427 = -0.710092276584265$

rat: replaced  $-0.7325994808911623$  by  $-12479523/17034578 = -0.7325994808911614$   
rat: replaced  $-0.7555254217604808$  by  $-16394539/21699520 = -0.7555254217604813$   
rat: replaced  $-0.7788723065797394$  by  $-17129047/21992112 = -0.7788723065797409$   
rat: replaced  $-0.8026423006424119$  by  $-78532681/97842689 = -0.8026423006424118$   
rat: replaced  $-0.8268375269314006$  by  $-14288533/17280944 = -0.8268375269313991$   
rat: replaced  $-0.8514600659067393$  by  $-29344334/34463547 = -0.8514600659067391$   
rat: replaced  $-0.8765119552975495$  by  $-17808806/20317813 = -0.876511955297551$   
rat: replaced  $-0.9019951898982683$  by  $-17368197/19255310 = -0.901995189898267$   
rat: replaced  $-0.9279117213691723$  by  $-4803773/5176972 = -0.927911721369171$   
rat: replaced  $-0.9542634580412112$  by  $-20199596/21167735 = -0.9542634580412123$   
rat: replaced  $-0.9810522647251774$  by  $-22272134/22702291 = -0.9810522647251768$   
rat: replaced  $-1.008279962525227$  by  $-9926001/9844489 = -1.008279962525226$   
rat: replaced  $-1.035948328656769$  by  $-20447689/19738136 = -1.035948328656769$   
rat: replaced  $-1.064059096268749$  by  $-43784241/41148317 = -1.064059096268749$   
rat: replaced  $-1.092613954270329$  by  $-24228202/22174531 = -1.092613954270329$   
rat: replaced  $-1.121614547162007$  by  $-18354705/16364539 = -1.121614547162007$   
rat: replaced  $-1.15106247487116$  by  $-22035757/19143841 = -1.151062474871161$   
rat: replaced  $-1.180959292592057$  by  $-20456643/17322056 = -1.180959292592057$

```
rat: replaced -1.21130651063034 by -30900377/25509957 = -1.211306510630339
rat: replaced -1.242105594251995 by -6989671/5627276 = -1.242105594251997
rat: replaced -1.273357963536825 by -22090312/17348077 = -1.273357963536823
rat: replaced -1.305064993236445 by -14582607/11173855 = -1.305064993236444
rat: replaced -1.337228012636809 by -33469619/25029104 = -1.337228012636809
rat: replaced -1.369848305425279 by -32941560/24047597 = -1.369848305425278
rat: replaced -2.254981225063221e-4 by -476777/2114328025 = -2.254981225063221e-4
rat: replaced -9.039699204008572e-4 by -554629/613548070 = -9.039699204008579e-4
rat: replaced -0.002038347522069071 by -1271429/623754775 = -0.00203834752206907
rat: replaced -0.003631517465696898 by -2066351/569004836 = -0.0036315174656969
rat: replaced -0.005686320410616744 by -635713/111796901 = -0.005686320410616749
rat: replaced -0.008205550853247354 by -2741742/334132595 = -0.008205550853247347
rat: replaced -0.01119195684764358 by -1145556/102355291 = -0.01119195684764357
rat: replaced -0.01464823973069444 by -3593060/245289541 = -0.01464823973069443
rat: replaced -0.01857705385199264 by -2624072/141253399 = -0.01857705385199261
rat: replaced -0.02298100630839936 by -2189611/95279161 = -0.02298100630839938
rat: replaced -0.0278626566833399 by -3478181/124833071 = -0.02786265668333995
```

rat: replaced  $-0.03322451679084377$  by  $-2100144/63210671 = -0.03322451679084375$   
rat: replaced  $-0.03906905042436903$  by  $-3541941/90658487 = -0.03906905042436899$   
rat: replaced  $-0.04539867311042303$  by  $-2490333/54854753 = -0.04539867311042308$   
rat: replaced  $-0.05221575186701224$  by  $-4506215/86299916 = -0.05221575186701224$   
rat: replaced  $-0.0595226049669409$  by  $-10922963/183509492 = -0.0595226049669409$   
rat: replaced  $-0.06732150170598852$  by  $-4631344/68794425 = -0.06732150170598852$   
rat: replaced  $-0.07561466217598092$  by  $-14346317/189729301 = -0.07561466217598092$   
rat: replaced  $-0.0844042570427819$  by  $-3521587/41722860 = -0.08440425704278182$   
rat: replaced  $-0.09369240732922907$  by  $-5174175/55225126 = -0.0936924073292291$   
rat: replaced  $-0.1034811842030341$  by  $-2097183/20266322 = -0.103481184203034$   
rat: replaced  $-0.1137726087696672$  by  $-11392983/100138189 = -0.1137726087696673$   
rat: replaced  $-0.1245686518702483$  by  $-6834267/54863458 = -0.1245686518702485$   
rat: replaced  $-0.1358712338844633$  by  $-8117277/59742425 = -0.1358712338844632$   
rat: replaced  $-0.1476822245385299$  by  $-6303644/42683837 = -0.1476822245385297$   
rat: replaced  $-0.1600034427182252$  by  $-9148317/57175751 = -0.1600034427182251$   
rat: replaced  $-0.1728366562869992$  by  $-22187021/128369881 = -0.1728366562869993$   
rat: replaced  $-0.1861835819091898$  by  $-17269805/92756863 = -0.1861835819091898$   
rat: replaced  $-0.200045884878356$  by  $-7864089/39311426 = -0.2000458848783557$

rat: replaced -0.2144251789507544 by -7172489/33449845 = -0.2144251789507545  
rat: replaced -0.2293230261839593 by -10351388/45138895 = -0.2293230261839595  
rat: replaced -0.244740936780663 by -8538850/34889341 = -0.2447409367806632  
rat: replaced -0.2606803689376539 by -18287762/70153967 = -0.260680368937654  
rat: replaced -0.277142728699999 by -8575365/30942053 = -0.2771427286999993  
rat: replaced -0.2941293698204409 by -4485287/15249368 = -0.2941293698204411  
rat: replaced -0.3116415936240235 by -9259347/29711525 = -0.311641593624023  
rat: replaced -0.3296806488779594 by -11717987/35543448 = -0.3296806488779592  
rat: replaced -0.3482477316667564 by -36675861/105315434 = -0.3482477316667564  
rat: replaced -0.3673439852726074 by -9533778/25953271 = -0.3673439852726078  
rat: replaced -0.386970500061066 by -7376119/19061192 = -0.3869705000610665  
rat: replaced -0.4071283133720089 by -28281643/69466166 = -0.4071283133720091  
rat: replaced -0.4278184094159029 by -15961498/37309049 = -0.4278184094159034  
rat: replaced -0.4490417191753848 by -13063519/29091994 = -0.4490417191753855  
rat: replaced -0.4707991203121662 by -38805045/82423784 = -0.4707991203121662  
rat: replaced -0.493091437079264 by -14163447/28723774 = -0.493091437079264  
rat: replaced -0.5159194402385774 by -24505042/47497807 = -0.5159194402385777

rat: replaced  $-0.5392838469838155$  by  $-46178693/85629661 = -0.5392838469838156$   
rat: replaced  $-0.5631853208687732$  by  $-40347511/71641624 = -0.5631853208687732$   
rat: replaced  $-0.58762447174098$  by  $-29058549/49450883 = -0.5876244717409799$   
rat: replaced  $-0.6126018556807142$  by  $-18023233/29420794 = -0.6126018556807135$   
rat: replaced  $-0.6381179749453973$  by  $-17107844/26809845 = -0.6381179749453979$   
rat: replaced  $-0.6641732779193661$  by  $-21974679/33085762 = -0.6641732779193661$   
rat: replaced  $-0.6907681590690352$  by  $-10003471/14481662 = -0.690768159069035$   
rat: replaced  $-0.7179029589034526$  by  $-28073639/39105061 = -0.7179029589034525$   
rat: replaced  $-0.7455779639402473$  by  $-23235768/31164773 = -0.7455779639402476$   
rat: replaced  $-0.773793406676978$  by  $-8384139/10835113 = -0.7737934066769769$   
rat: replaced  $-0.8025494655678836$  by  $-7868837/9804800 = -0.8025494655678851$   
rat: replaced  $-0.8318462650060389$  by  $-30346636/36481063 = -0.8318462650060389$   
rat: replaced  $-0.861683875310914$  by  $-16277728/18890603 = -0.8616838753109152$   
rat: replaced  $-0.8920623127213426$  by  $-37841947/42420744 = -0.8920623127213422$   
rat: replaced  $-0.9229815393938994$  by  $-70218740/76078163 = -0.9229815393938994$   
rat: replaced  $-0.954441463406683$  by  $-24594815/25768804 = -0.9544414634066836$   
rat: replaced  $-0.9864419387685092$  by  $-15639887/15854848 = -0.9864419387685079$   
rat: replaced  $-1.018982765433508$  by  $-45306349/44462331 = -1.018982765433508$

rat: replaced  $-1.052063689321131$  by  $-30825646/29300171 = -1.052063689321131$   
rat: replaced  $-1.085684402341557$  by  $-38378733/35349806 = -1.085684402341557$   
rat: replaced  $-1.119844542426501$  by  $-24894563/22230374 = -1.119844542426502$   
rat: replaced  $-1.15454369356542$  by  $-13440326/11641245 = -1.154543693565422$   
rat: replaced  $-1.189781385847118$  by  $-34027123/28599475 = -1.189781385847118$   
rat: replaced  $-1.22555709550673$  by  $-35019680/28574499 = -1.225557095506731$   
rat: replaced  $-1.261870244978105$  by  $-35214941/27906943 = -1.261870244978105$   
rat: replaced  $-1.298720202951555$  by  $-33824443/26044442 = -1.298720202951555$   
rat: replaced  $-1.336106284436991$  by  $-5396397/4038898 = -1.336106284436992$   
rat: replaced  $-1.374027750832421$  by  $-21232969/15453086 = -1.374027750832423$   
rat: replaced  $-1.412483809997805$  by  $-11919739/8438850 = -1.412483809997808$   
rat: replaced  $-1.451473616334275$  by  $-8780993/6049709 = -1.451473616334273$   
rat: replaced  $-1.490996270868687$  by  $-44673937/29962474 = -1.490996270868687$   
rat: replaced  $-1.531050821343523$  by  $-64281527/41985234 = -1.531050821343523$   
rat: replaced  $-1.571636262312116$  by  $-16616467/10572718 = -1.571636262312113$   
rat: replaced  $-1.612751535239189$  by  $-65392401/40547102 = -1.612751535239189$   
rat: replaced  $-1.654395528606713$  by  $-27646066/16710675 = -1.654395528606714$

rat: replaced  $-1.696567078025054$  by  $-7906291/4660170 = -1.696567078025051$   
rat: replaced  $-1.739264966349412$  by  $-21773512/12518801 = -1.739264966349413$   
rat: replaced  $-1.782487923801538$  by  $-19245269/10796858 = -1.782487923801536$   
rat: replaced  $-1.826234628096705$  by  $-18221771/9977782 = -1.826234628096705$   
rat: replaced  $-1.870503704575938$  by  $-10770134/5757879 = -1.870503704575939$   
rat: replaced  $-1.915293726343482$  by  $-16780009/8761063 = -1.915293726343481$   
rat: replaced  $-1.960603214409484$  by  $-45722957/23320862 = -1.960603214409484$   
rat: replaced  $-2.006430637837895$  by  $-146296719/72913918 = -2.006430637837895$   
rat: replaced  $-2.052774413899562$  by  $-58550872/28522799 = -2.052774413899562$   
rat: replaced  $-2.099632908230499$  by  $-93949097/44745487 = -2.099632908230499$   
rat: replaced  $-2.147004434995322$  by  $-25274650/11772053 = -2.147004434995323$   
rat: replaced  $-2.194887257055829$  by  $-28867932/13152353 = -2.194887257055829$   
rat: replaced  $-2.243279586144718$  by  $-38403199/17119221 = -2.24327958614472$   
rat: replaced  $-2.292179583044406$  by  $-15612340/6811133 = -2.292179583044407$   
rat: replaced  $-2.341585357770954$  by  $-20809175/8886789 = -2.341585357770956$   
rat: replaced  $-2.391494969763059$  by  $-22142156/9258709 = -2.391494969763063$   
rat: replaced  $-2.441906428076114$  by  $-36070003/14771247 = -2.441906428076113$   
rat: replaced  $-2.492817691581298$  by  $-26575204/10660709 = -2.492817691581301$

```
part: invalid index of list or matrix.  
#0: lineIntersection(g=[-9/2,-3/2,-9/4],h=[-3,-9/2,-27/8])  
#1: circleThrough(a=[-3/2,-3/2],b=[3,0],c=[3/2,3])  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
LL &= circleThrough(A,B,C); $getCircleEquation(LL,x,y) ...  
^
```

```
>0 &= getCircleCenter(LL); $0  
>plotCircle(LL()); plotPoint(0(),"0"):
```

```
Function LL not found.  
Try list ... to find functions!  
Error in:  
plotCircle(LL()); plotPoint(0(),"0") ...  
^
```

```
>H &= lineIntersection(perpendicular(A,lineThrough(C,B)),...  
> perpendicular(B,lineThrough(A,C))); $H
```

```
Maxima said:  
rat: replaced -1.497487512542063e-4 by -299560/2000417349 = -1.497487512542064e-4  
  
rat: replaced -5.979800402663507e-4 by -330831/553247563 = -5.979800402663499e-4  
  
rat: replaced -0.001343149056780863 by -584699/435319518 = -0.001343149056780863  
  
rat: replaced -0.002383681297017493 by -756665/317435473 = -0.002383681297017489
```

rat: replaced -0.003717972721118644 by  $-2149606/578166157 = -0.003717972721118646$   
rat: replaced -0.005344389913554327 by  $-2779816/520137199 = -0.005344389913554327$   
rat: replaced -0.007261270246460387 by  $-1541197/212248952 = -0.007261270246460392$   
rat: replaced -0.009466922045900723 by  $-8174891/863521529 = -0.009466922045900722$   
rat: replaced -0.01195962476103374 by  $-2414321/201872638 = -0.01195962476103374$   
rat: replaced -0.01473762913616476 by  $-3432147/232883252 = -0.01473762913616476$   
rat: replaced -0.01779915738567177 by  $-1659683/93245032 = -0.01779915738567176$   
rat: replaced -0.0211424033717803 by  $-2960631/140032850 = -0.02114240337178026$   
rat: replaced -0.02476553278517801 by  $-1738361/70192756 = -0.024765532785178$   
rat: replaced -0.02866668332844304 by  $-1475047/51455098 = -0.02866668332844299$   
rat: replaced -0.03284396490227211 by  $-4221724/128538805 = -0.03284396490227212$   
rat: replaced -0.03729545979448817 by  $-6430372/172417019 = -0.03729545979448815$   
rat: replaced -0.04201922287181174 by  $-2263313/53863752 = -0.04201922287181183$   
rat: replaced -0.04701328177437193 by  $-1674543/35618509 = -0.04701328177437186$   
rat: replaced -0.05227563711293993 by  $-3997444/76468585 = -0.05227563711293991$   
rat: replaced -0.05780426266886693 by  $-3832681/66304470 = -0.05780426266886682$   
rat: replaced -0.06359710559670453 by  $-5078877/79860191 = -0.06359710559670462$

rat: replaced  $-0.06965208662948744$  by  $-10918553/156758448 = -0.06965208662948742$   
rat: replaced  $-0.07596710028665828$  by  $-5036501/66298450 = -0.07596710028665829$   
rat: replaced  $-0.08254001508461323$  by  $-6160264/74633667 = -0.08254001508461323$   
rat: replaced  $-0.0893686737498502$  by  $-3979484/44528847 = -0.08936867374985029$   
rat: replaced  $-0.09645089343469301$  by  $-4946630/51286513 = -0.09645089343469306$   
rat: replaced  $-0.1037844659355751$  by  $-7809283/75245201 = -0.1037844659355751$   
rat: replaced  $-0.1113671579138579$  by  $-6096479/54742162 = -0.1113671579138581$   
rat: replaced  $-0.1191967111191618$  by  $-7273952/61024771 = -0.1191967111191618$   
rat: replaced  $-0.1272708426151914$  by  $-9595393/75393490 = -0.1272708426151913$   
rat: replaced  $-0.1355872450080251$  by  $-5659716/41742245 = -0.1355872450080249$   
rat: replaced  $-0.1441435866768541$  by  $-2581028/17905951 = -0.144143586676854$   
rat: replaced  $-0.1529375120071418$  by  $-1082663/7079120 = -0.1529375120071421$   
rat: replaced  $-0.161966641626183$  by  $-5766177/35601016 = -0.1619666416261828$   
rat: replaced  $-0.1712285726410404$  by  $-5923297/34592924 = -0.1712285726410407$   
rat: replaced  $-0.1807208788788305$  by  $-55437725/306758828 = -0.1807208788788305$   
rat: replaced  $-0.1904411111293399$  by  $-5908417/31024903 = -0.1904411111293402$   
rat: replaced  $-0.2003867973899436$  by  $-6986853/34866833 = -0.2003867973899436$   
rat: replaced  $-0.2105554431128032$  by  $-4264228/20252281 = -0.2105554431128029$

rat: replaced  $-0.2209445314543208$  by  $-9342805/42285749 = -0.2209445314543205$   
rat: replaced  $-0.2315515235268193$  by  $-4085380/17643503 = -0.2315515235268189$   
rat: replaced  $-0.2423738586524308$  by  $-187964237/775513655 = -0.2423738586524308$   
rat: replaced  $-0.2534089546191609$  by  $-7570461/29874481 = -0.2534089546191614$   
rat: replaced  $-0.2646542079391092$  by  $-6530305/24674858 = -0.2646542079391095$   
rat: replaced  $-0.2761069941088149$  by  $-10501531/38034281 = -0.2761069941088146$   
rat: replaced  $-0.2877646678717041$  by  $-78631265/273248504 = -0.2877646678717041$   
rat: replaced  $-0.2996245634826158$  by  $-7854364/26214019 = -0.2996245634826159$   
rat: replaced  $-0.3116839949743722$  by  $-12170593/39047860 = -0.3116839949743725$   
rat: replaced  $-0.3239402564263729$  by  $-11348921/35033994 = -0.3239402564263726$   
rat: replaced  $-0.3363906222351865$  by  $-33578595/99820247 = -0.3363906222351864$   
rat: replaced  $-0.3490323473871076$  by  $-11921804/34156731 = -0.349032347387108$   
rat: replaced  $-0.3618626677326557$  by  $-17595895/48625892 = -0.3618626677326557$   
rat: replaced  $-0.3748788002629876$  by  $-46177544/123179929 = -0.3748788002629876$   
rat: replaced  $-0.3880779433881974$  by  $-16190143/41718792 = -0.3880779433881978$   
rat: replaced  $-0.401457277217472$  by  $-60525431/150764314 = -0.401457277217472$   
rat: replaced  $-0.415013963841077$  by  $-12594557/30347309 = -0.4150139638410773$

rat: replaced  $-0.4287451476141481$  by  $-27500639/64142158 = -0.4287451476141479$   
rat: replaced  $-0.4426479554422498$  by  $-14824369/33490201 = -0.4426479554422501$   
rat: replaced  $-0.4567194970686855$  by  $-21704313/47522195 = -0.4567194970686855$   
rat: replaced  $-0.4709568653635186$  by  $-16486730/35006879 = -0.470956865363519$   
rat: replaced  $-0.4853571366142837$  by  $-267196237/550514697 = -0.4853571366142837$   
rat: replaced  $-0.4999173708183561$  by  $-15013400/30031763 = -0.4999173708183566$   
rat: replaced  $-0.5146346119769494$  by  $-20942773/40694451 = -0.5146346119769499$   
rat: replaced  $-0.5295058883907107$  by  $-26908094/50817365 = -0.5295058883907106$   
rat: replaced  $-0.5445282129568924$  by  $-22151821/40680759 = -0.544528212956892$   
rat: replaced  $-0.5596985834680561$  by  $-41889600/74843141 = -0.5596985834680562$   
rat: replaced  $-0.5750139829122923$  by  $-11820697/20557234 = -0.5750139829122926$   
rat: replaced  $-0.5904713797749195$  by  $-13730652/23253713 = -0.5904713797749203$   
rat: replaced  $-0.6060677283416327$  by  $-16634707/27446944 = -0.6060677283416325$   
rat: replaced  $-0.621799969003072$  by  $-3133380/5039209 = -0.6217999690030717$   
rat: replaced  $-0.6376650285607812$  by  $-24667763/38684516 = -0.6376650285607812$   
rat: replaced  $-0.6536598205345254$  by  $-15672861/23977091 = -0.6536598205345261$   
rat: replaced  $-0.6697812454709364$  by  $-4173133/6230591 = -0.6697812454709353$   
rat: replaced  $-0.6860261912534547$  by  $-19587769/28552509 = -0.6860261912534552$

rat: replaced  $-0.7023915334135393$  by  $-14227114/20255247 = -0.7023915334135397$   
rat: replaced  $-0.7188741354431122$  by  $-40957539/56974562 = -0.7188741354431123$   
rat: replaced  $-0.7354708491082056$  by  $-15902500/21622203 = -0.735470849108206$   
rat: replaced  $-0.7521785147637838$  by  $-19967209/26545838 = -0.7521785147637833$   
rat: replaced  $-0.7689939616697044$  by  $-16336853/21244449 = -0.7689939616697049$   
rat: replaced  $-0.7859140083077888$  by  $-21511393/27371179 = -0.7859140083077898$   
rat: replaced  $-0.8029354626999737$  by  $-9186705/11441399 = -0.8029354626999723$   
rat: replaced  $-0.8200551227275041$  by  $-39606167/48296957 = -0.8200551227275044$   
rat: replaced  $-0.8372697764511438$  by  $-3747190/4475487 = -0.8372697764511438$   
rat: replaced  $-0.8545762024323653$  by  $-43827549/51285712 = -0.8545762024323655$   
rat: replaced  $-0.8719711700554936$  by  $-21551370/24715691 = -0.8719711700554923$   
rat: replaced  $-0.8894514398507607$  by  $-14730441/16561265 = -0.8894514398507601$   
rat: replaced  $-0.9070137638182549$  by  $-23593213/26011968 = -0.9070137638182547$   
rat: replaced  $-0.9246548857527144$  by  $-17429936/18850207 = -0.9246548857527135$   
rat: replaced  $-0.942371541569146$  by  $-21072616/22361261 = -0.9423715415691449$   
rat: replaced  $-0.9601604596292325$  by  $-54513257/56775153 = -0.9601604596292326$   
rat: replaced  $-0.978018361068492$  by  $-77467650/79208789 = -0.978018361068492$

```
rat: replaced -0.9959419601241615 by -12215999/12265774 = -0.9959419601241634
rat: replaced -1.013927964463772 by -21561239/21265060 = -1.013927964463773
rat: replaced -1.031973075514378 by -37324525/36168119 = -1.031973075514378
rat: replaced -1.050073988792411 by -29992669/28562434 = -1.050073988792412
rat: replaced -1.068227394234129 by -23767018/22249025 = -1.068227394234129
rat: replaced -1.086429976526613 by -11795807/10857402 = -1.086429976526613
rat: replaced -1.104678415439305 by -16973206/15364839 = -1.104678415439303
rat: replaced -1.122969386156019 by -33371626/29717307 = -1.12296938615602
rat: replaced 2.254981225063221e-4 by 476777/2114328025 = 2.254981225063221e-4
rat: replaced 9.039699204008572e-4 by 554629/613548070 = 9.039699204008579e-4
rat: replaced 0.002038347522069071 by 1271429/623754775 = 0.00203834752206907
rat: replaced 0.003631517465696898 by 2066351/569004836 = 0.0036315174656969
rat: replaced 0.005686320410616744 by 635713/111796901 = 0.005686320410616749
rat: replaced 0.008205550853247354 by 2741742/334132595 = 0.008205550853247347
rat: replaced 0.01119195684764358 by 1145556/102355291 = 0.01119195684764357
rat: replaced 0.01464823973069444 by 3593060/245289541 = 0.01464823973069443
rat: replaced 0.01857705385199264 by 2624072/141253399 = 0.01857705385199261
rat: replaced 0.02298100630839936 by 2189611/95279161 = 0.02298100630839938
```

rat: replaced 0.0278626566833399 by 3478181/124833071 = 0.02786265668333995  
rat: replaced 0.03322451679084377 by 2100144/63210671 = 0.03322451679084375  
rat: replaced 0.03906905042436903 by 3541941/90658487 = 0.03906905042436899  
rat: replaced 0.04539867311042303 by 2490333/54854753 = 0.04539867311042308  
rat: replaced 0.05221575186701224 by 4506215/86299916 = 0.05221575186701224  
rat: replaced 0.0595226049669409 by 10922963/183509492 = 0.0595226049669409  
rat: replaced 0.06732150170598852 by 4631344/68794425 = 0.06732150170598852  
rat: replaced 0.07561466217598092 by 14346317/189729301 = 0.07561466217598092  
rat: replaced 0.0844042570427819 by 3521587/41722860 = 0.08440425704278182  
rat: replaced 0.09369240732922907 by 5174175/55225126 = 0.0936924073292291  
rat: replaced 0.1034811842030341 by 2097183/20266322 = 0.103481184203034  
rat: replaced 0.1137726087696672 by 11392983/100138189 = 0.1137726087696673  
rat: replaced 0.1245686518702483 by 6834267/54863458 = 0.1245686518702485  
rat: replaced 0.1358712338844633 by 8117277/59742425 = 0.1358712338844632  
rat: replaced 0.1476822245385299 by 6303644/42683837 = 0.1476822245385297  
rat: replaced 0.1600034427182252 by 9148317/57175751 = 0.1600034427182251  
rat: replaced 0.1728366562869992 by 22187021/128369881 = 0.1728366562869993

rat: replaced 0.1861835819091898 by 17269805/92756863 = 0.1861835819091898  
rat: replaced 0.200045884878356 by 7864089/39311426 = 0.2000458848783557  
rat: replaced 0.2144251789507544 by 7172489/33449845 = 0.2144251789507545  
rat: replaced 0.2293230261839593 by 10351388/45138895 = 0.2293230261839593  
rat: replaced 0.244740936780663 by 8538850/34889341 = 0.2447409367806632  
rat: replaced 0.2606803689376539 by 18287762/70153967 = 0.260680368937654  
rat: replaced 0.277142728699999 by 8575365/30942053 = 0.2771427286999993  
rat: replaced 0.2941293698204409 by 4485287/15249368 = 0.2941293698204411  
rat: replaced 0.3116415936240235 by 9259347/29711525 = 0.311641593624023  
rat: replaced 0.3296806488779594 by 11717987/35543448 = 0.3296806488779592  
rat: replaced 0.3482477316667564 by 36675861/105315434 = 0.3482477316667564  
rat: replaced 0.3673439852726074 by 9533778/25953271 = 0.3673439852726078  
rat: replaced 0.386970500061066 by 7376119/19061192 = 0.3869705000610665  
rat: replaced 0.4071283133720089 by 28281643/69466166 = 0.4071283133720091  
rat: replaced 0.4278184094159029 by 15961498/37309049 = 0.4278184094159034  
rat: replaced 0.4490417191753848 by 13063519/29091994 = 0.4490417191753855  
rat: replaced 0.4707991203121662 by 38805045/82423784 = 0.4707991203121662  
rat: replaced 0.493091437079264 by 14163447/28723774 = 0.493091437079264

rat: replaced 0.5159194402385774 by 24505042/47497807 = 0.5159194402385777  
rat: replaced 0.5392838469838155 by 46178693/85629661 = 0.5392838469838156  
rat: replaced 0.5631853208687732 by 40347511/71641624 = 0.5631853208687732  
rat: replaced 0.58762447174098 by 29058549/49450883 = 0.5876244717409799  
rat: replaced 0.6126018556807142 by 18023233/29420794 = 0.6126018556807135  
rat: replaced 0.6381179749453973 by 17107844/26809845 = 0.6381179749453979  
rat: replaced 0.6641732779193661 by 21974679/33085762 = 0.6641732779193661  
rat: replaced 0.6907681590690352 by 10003471/14481662 = 0.690768159069035  
rat: replaced 0.7179029589034526 by 28073639/39105061 = 0.7179029589034525  
rat: replaced 0.7455779639402473 by 23235768/31164773 = 0.7455779639402476  
rat: replaced 0.773793406676978 by 8384139/10835113 = 0.7737934066769769  
rat: replaced 0.8025494655678836 by 7868837/9804800 = 0.8025494655678851  
rat: replaced 0.8318462650060389 by 30346636/36481063 = 0.8318462650060389  
rat: replaced 0.861683875310914 by 16277728/18890603 = 0.8616838753109152  
rat: replaced 0.8920623127213426 by 37841947/42420744 = 0.8920623127213422  
rat: replaced 0.9229815393938994 by 70218740/76078163 = 0.9229815393938994  
rat: replaced 0.954441463406683 by 24594815/25768804 = 0.9544414634066836

rat: replaced 0.9864419387685092 by  $15639887/15854848 = 0.9864419387685079$   
rat: replaced 1.018982765433508 by  $45306349/44462331 = 1.018982765433508$   
rat: replaced 1.052063689321131 by  $30825646/29300171 = 1.052063689321131$   
rat: replaced 1.085684402341557 by  $38378733/35349806 = 1.085684402341557$   
rat: replaced 1.119844542426501 by  $24894563/22230374 = 1.119844542426502$   
rat: replaced 1.15454369356542 by  $13440326/11641245 = 1.154543693565422$   
rat: replaced 1.189781385847118 by  $34027123/28599475 = 1.189781385847118$   
rat: replaced 1.22555709550673 by  $35019680/28574499 = 1.225557095506731$   
rat: replaced 1.261870244978105 by  $35214941/27906943 = 1.261870244978105$   
rat: replaced 1.298720202951555 by  $33824443/26044442 = 1.298720202951555$   
rat: replaced 1.336106284436991 by  $5396397/4038898 = 1.336106284436992$   
rat: replaced 1.374027750832421 by  $21232969/15453086 = 1.374027750832423$   
rat: replaced 1.412483809997805 by  $11919739/8438850 = 1.412483809997808$   
rat: replaced 1.451473616334275 by  $8780993/6049709 = 1.451473616334273$   
rat: replaced 1.490996270868687 by  $44673937/29962474 = 1.490996270868687$   
rat: replaced 1.531050821343523 by  $64281527/41985234 = 1.531050821343523$   
rat: replaced 1.571636262312116 by  $16616467/10572718 = 1.571636262312113$   
rat: replaced 1.612751535239189 by  $65392401/40547102 = 1.612751535239189$

rat: replaced 1.654395528606713 by 27646066/16710675 = 1.654395528606714  
rat: replaced 1.696567078025054 by 7906291/4660170 = 1.696567078025051  
rat: replaced 1.739264966349412 by 21773512/12518801 = 1.739264966349413  
rat: replaced 1.782487923801538 by 19245269/10796858 = 1.782487923801536  
rat: replaced 1.826234628096705 by 18221771/9977782 = 1.826234628096705  
rat: replaced 1.870503704575938 by 10770134/5757879 = 1.870503704575939  
rat: replaced 1.915293726343482 by 16780009/8761063 = 1.915293726343481  
rat: replaced 1.960603214409484 by 45722957/23320862 = 1.960603214409484  
rat: replaced 2.006430637837895 by 146296719/72913918 = 2.006430637837895  
rat: replaced 2.052774413899562 by 58550872/28522799 = 2.052774413899562  
rat: replaced 2.099632908230499 by 93949097/44745487 = 2.099632908230499  
rat: replaced 2.147004434995322 by 25274650/11772053 = 2.147004434995323  
rat: replaced 2.194887257055829 by 28867932/13152353 = 2.194887257055829  
rat: replaced 2.243279586144718 by 38403199/17119221 = 2.24327958614472  
rat: replaced 2.292179583044406 by 15612340/6811133 = 2.292179583044407  
rat: replaced 2.341585357770954 by 20809175/8886789 = 2.341585357770956  
rat: replaced 2.391494969763059 by 22142156/9258709 = 2.391494969763063

```
rat: replaced 2.441906428076114 by 36070003/14771247 = 2.441906428076113  
  
rat: replaced 2.492817691581298 by 26575204/10660709 = 2.492817691581301  
part: invalid index of list or matrix.  
#0: lineIntersection(g=[3/2,-3,9/4],h=[3,9/2,9])  
-- an error. To debug this try: debugmode(true);  
  
Error in:  
perpendicular(B,lineThrough(A,C))); $H ...  
^
```

```
>el &= lineThrough(H,O); $getLineEquation(el,x,y)
```

```
>plotPoint(H(),"H"); plotLine(el(),"Garis Euler"):
```

```
Function H not found.  
Try list ... to find functions!  
Error in:  
plotPoint(H(),"H"); plotLine(el(),"Garis Euler"): ...  
^
```

```
>M &= (A+B+C)/3; $getLineEquation(el,x,y) with [x=M[1],y=M[2]]  
>plotPoint(M(),"M"): // titik berat
```

Variable C not found!  
Use global variables or parameters for string evaluation.  
Error in expression: (C+B+A)/3  
Error in:  
plotPoint(M(),"M"): // titik berat ...  
^

```
>$distance(M,H)/distance(M,O)|radcan
```

```
>$computeAngle(A,C,B), degprint(%())
```

Variable or function A not found.  
Error in expression: computeAngle(A,C,B)  
Error in:  
\$computeAngle(A,C,B), degprint(%()) ...  
^

```
>Q &= lineIntersection(angleBisector(A,C,B),angleBisector(C,B,A))|radcan; $Q  
>r &= distance(Q,projectToLine(Q,lineThrough(A,B)))|ratsimp; $r  
>LD &= circleWithCenter(Q,r); // Lingkaran dalam
```

```
>color(5); plotCircle(LD()):
```

```
Variable or function A not found.  
Error in expression: circleWithCenter(lineIntersection(angleBisector(A,C,B),angleBisector(C,B,A)),  
Error in:  
color(5); plotCircle(LD()): ...  
^
```

### contoh lain dari materi trigonometri rasional

---

```
>A&:=[2,3]; B&:=[5,4]; C&:=[0,5]; ...  
>setPlotRange(-1,5,1,7); ...  
>plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C"); ...  
>plotSegment(B,A,"c"); plotSegment(A,C,"b"); plotSegment(C,B,"a"); ...  
>insimg;
```

```
Function setPlotRange not found.  
Try list ... to find functions!  
Error in:  
... ,3]; B&:=[5,4]; C&:=[0,5]; setPlotRange(-1,5,1,7); plotPoint(A ...  
^
```

```
>$distance(A,B)  
>c &= quad(A,B); $c, b &= quad(A,C); $b, a &= quad(B,C); $a,
```

```
>wb &= computeAngle(A,B,C); $wb, $(wb/pi*180)()
```

```
Function computeAngle not found.  
Try list ... to find functions!  
Error in expression: 180*computeAngle([2,3],[5,4],[0,5])/pi  
Error in:  
wb &= computeAngle(A,B,C); $wb, $(wb/pi*180)() ...
```

```
>$crosslaw(a,b,c,x), $solve(% ,x) , // $(b+c-a)^2 = 4b.c(1-x)$   
>sb &= spread(b,a,c); $sb  
>$sin(computeAngle(A,B,C))^2  
>ha &= c*sb; $ha  
>$sqrt(ha)  
>$sqrt(ha)*sqrt(a)/2
```

```
>$areaTriangle(B,A,C)
```

## Aturan penyebaran 3 kali lipat

---

```
>setPlotRange(1); ...
>color(1); plotCircle(circleWithCenter([0,0],1)); ...
>A:=[cos(1),sin(1)]; B:=[cos(2),sin(2)]; C:=[cos(6),sin(6)]; ...
>plotPoint(A,"A"); plotPoint(B,"B"); plotPoint(C,"C"); ...
>color(3); plotSegment(A,B,"c"); plotSegment(A,C,"b"); plotSegment(C,B,"a"); ...
>color(1); O:=[0,0]; plotPoint(O,"O"); ...
>plotSegment(A,O); plotSegment(B,O); plotSegment(C,O,"r"); ...
>insimg;
```

```
Function setPlotRange not found.
Try list ... to find functions!
Error in:
setPlotRange(1); color(1); plotCircle(circleWithCenter([0,0],1 ...
^
```

```
>&remvalue(a,b,c,r); // hapus nilai-nilai sebelumnya untuk perhitungan baru
>rabc &= rhs(solve(triplespread(spread(b,r,r),spread(a,r,r),spread(c,r,r)),r)[4]); $rabc
```

```
Maxima said:
part: invalid index of list or matrix.
-- an error. To debug this try: debugmode(true);

Error in:
... spread(b,r,r),spread(a,r,r),spread(c,r,r)),r)[4]); $rabc ...
^
```

```
>function periradius(a,b,c) &= rabc;
```

```
>a:=quadrance(B,C); b:=quadrance(A,C); c:=quadrance(A,B);
```

```
Function quadrance not found.  
Try list ... to find functions!  
Error in:  
a:=quadrance(B,C); b:=quadrance(A,C); c:=quadrance(A,B); ...  
^
```

```
>periradius(a,b,c)
```

```
Variable rabc not found!  
Use global or local variables defined in function periradius.  
Try "trace errors" to inspect local variables after errors.  
periradius:  
    useglobal; return rabc  
Error in:  
periradius(a,b,c) ...  
^
```

```
>$spread(b,a,c)*rabc | ratsimp  
>$doublespread(b/(4*r))-spread(b,r,r) | ratsimp
```

## Contoh 6: Jarak Minimal pada Bidang

---

### Catatan awal

---

Fungsi yang, ke titik M di bidang, menetapkan jarak AM antara titik tetap A dan M, memiliki garis level yang agak sederhana: lingkaran berpusat di A.

```
>&remvalue();  
>A=[-2,-2];  
>function d1(x,y):=sqrt((x-A[1])^2+(y-A[2])^2)  
>fcontour("d1",xmin=-2,xmax=0,ymin=-2,ymax=0,hue=1, ...  
>title="If you see ellipses, please set your window square":
```

dan grafiknya juga agak sederhana: bagian atas kerucut:

```
>plot3d("d1",xmin=-2,xmax=0,ymin=-2,ymax=0):
```

Ternyata setelah mencoba yang bisa hanya dengan memasukkan angka 1, karena ketika memakai angka 2, plot tidak membentuk kerucut diatas.

## Dua poin

---

```
>B=[2,-2];
>function d2(x,y):=d1(x,y)+sqrt((x-B[1])^2+(y-B[2])^2)
>fcontour("d2",xmin=-2,xmax=2,ymin=-3,ymax=1,hue=1):
```

Grafiknya lebih menarik:

```
>plot3d("d2",xmin=-2,xmax=2,ymin=-3,ymax=1):
```

Pembatasan garis (AB) lebih terkenal:

```
>plot2d("abs(x+1)+abs(x-1)",xmin=-3,xmax=3):
```

**Tiga poin**

---

Contoh:

```
>C=[-3,2];
>function d3(x,y):=d2(x,y)+sqrt((x-C[1])^2+(y-C[2])^2)
>plot3d("d3",xmin=-5,xmax=3,ymin=-4,ymax=4);
>insimg;
>fcontour("d3",xmin=-4,xmax=1,ymin=-2,ymax=2,hue=1,title="The minimum is on A");
>P=(A_B_C_A)';
>plot2d(P[1],P[2],add=1,color=12);
>insimg;
```

Tetapi jika semua sudut segitiga ABC kurang dari  $120^\circ$ , minimumnya adalah pada titik F di bagian dalam segitiga, yang merupakan satu-satunya titik yang melihat sisi-sisi ABC dengan sudut yang sama (maka masing-masing  $120^\circ$ ):

```
>C=[-1,2];
>plot3d("d3",xmin=-2,xmax=2,ymin=-2,ymax=2);
>fcontour("d3",xmin=-2,xmax=2,ymin=-2,ymax=2,hue=1,title="The Fermat point");
>P=(A_B_C_A)';
>plot2d(P[1],P[2],add=1,color=12);
>insimg;
```

## Empat poin

---

Langkah selanjutnya adalah menambahkan 4 titik D dan mencoba meminimalkan MA+MB+MC+MD; katakan bahwa Anda adalah operator TV kabel dan ingin mencari di bidang mana Anda harus meletakkan antena sehingga Anda dapat memberi makan empat desa dan menggunakan panjang kabel sesedikit mungkin!

```
>D=[2,2];
>function d4(x,y):=d3(x,y)+sqrt((x-D[1])^2+(y-D[2])^2)
>plot3d("d4",xmin=-1.5,xmax=1.5,ymin=-1.5,ymax=1.5);
>fcontour("d4",xmin=-1.5,xmax=1.5,ymin=-1.5,ymax=1.5,hue=1);
>P=(A_B_C_D)';
>plot2d(P[1],P[2],points=1,add=1,color=12);
>insimg;
```

## Contoh 7: Bola Dandelin dengan Povray

---

```
>load geometry;
```

Pertama dua garis yang membentuk kerucut.

```
>g1 &= lineThrough([0,0],[2,a])
```

$[- a, 2, 0]$

```
>g2 &= lineThrough([0,0],[-2,a])
```

$[- a, - 2, 0]$

```
>g &= lineThrough([-2,0],[2,2])
```

[ - 2, 4, 4]

```
>setPlotRange(-2,2,0,3);
>color(black); plotLine(g(),"")
>a:=2; color(blue); plotLine(g1(),""), plotLine(g2(),""):
```

Sekarang kita ambil titik umum pada sumbu y.

```
>P &= [0,u]
```

[0, u]

Hitung jarak ke g1.

```
>d1 &= distance(P,projectToLine(P,g1)); $d1
```

Hitung jarak ke g.

```
>d &= distance(P,projectToLine(P,g)); $d
```

Dan temukan pusat kedua lingkaran yang jaraknya sama.

```
>sol &= solve(d1^2=d^2,u); $sol
```

Ada dua solusi.

```
>u := sol()
```

[0.558482, 4.77485]

```
>dd := d()
```

```
[0.394906, 3.37633]
```

Plot lingkaran ke dalam gambar.

```
>color(red);
>plotCircle(circleWithCenter([0,u[1]],dd[1]), "");
>plotCircle(circleWithCenter([0,u[2]],dd[2]), "");
>insimg;
```

---

## Latihan

1. Gambarlah segi-n beraturan jika diketahui titik pusat O, n, dan jarak titik pusat ke titik-titik sudut segi-n tersebut (jari-jari lingkaran luar segi-n), r.

Petunjuk:

- Besar sudut pusat yang menghadap masing-masing sisi segi-n adalah  $(360/n)$ .
- Titik-titik sudut segi-n merupakan perpotongan lingkaran luar segi-n dan garis-garis yang melalui pusat dan saling membentuk sudut sebesar kelipatan  $(360/n)$ .
- Untuk  $n$  ganjil, pilih salah satu titik sudut adalah di atas.
- Untuk  $n$  genap, pilih 2 titik di kanan dan kiri lurus dengan titik pusat.
- Anda dapat menggambar segi-3, 4, 5, 6, 7, dst beraturan.

Penyelesaian :

```
>load geometry
```

Numerical and symbolic geometry.

```
>setPlotRange(-3.5,3.5,-3.5,3.5);
>A=[-2,-2]; plotPoint(A,"A");
>B=[2,-2]; plotPoint(B,"B");
>C=[0,3]; plotPoint(C,"C");
>plotSegment(A,B,"c");
>plotSegment(B,C,"a");
>plotSegment(A,C,"b");
>aspect(1);
>c=circleThrough(A,B,C);
>R=getCircleRadius(c);
>O=getCircleCenter(c);
>plotPoint(O,"O");
>l=angleBisector(A,C,B);
>color(2); plotLine(l); color(1);
```

```
>plotCircle(c,"Lingkaran luar segitiga ABC"):
```

2. Gambarlah suatu parabola yang melalui 3 titik yang diketahui.

Petunjuk:

- Misalkan persamaan parabolanya  $y = ax^2 + bx + c$ .
- Substitusikan koordinat titik-titik yang diketahui ke persamaan tersebut.
- Selesaikan SPL yang terbentuk untuk mendapatkan nilai-nilai  $a$ ,  $b$ ,  $c$ .

Penyelesaian :

```
>load geometry;
>setPlotRange(5); P=[2,0]; Q=[4,0]; R=[0,-4];
>plotPoint(P,"P"); plotPoint(Q,"Q"); plotPoint(R,"R");
>sol &= solve([a+b=-c,16*a+4*b=-c,c=-4],[a,b,c])
```

```
[[a = - 1, b = 5, c = - 4]]
```

Sehingga didapatkan nilai  $a = -1$ ,  $b = 5$  dan  $c = -4$

```
>function y&=-x^2+5*x-4
```

$$-x^2 + 5x - 4$$

```
>plot2d("-x^2+5*x-4", -5, 5, -5, 5):
```

3. Gambarlah suatu segi-4 yang diketahui keempat titik sudutnya, misalnya A, B, C, D.

- Tentukan apakah segi-4 tersebut merupakan segi-4 garis singgung

(sisinya-sisinya merupakan garis singgung lingkaran yang sama yakni lingkaran dalam segi-4 tersebut).

- Suatu segi-4 merupakan segi-4 garis singgung apabila keempat

garis bagi sudutnya bertemu di satu titik.

- Jika segi-4 tersebut merupakan segi-4 garis singgung, gambar lingkaran dalamnya.

- Tunjukkan bahwa syarat suatu segi-4 merupakan segi-4 garis singgung apabila hasil kali panjang sisi-sisi yang berhadapan sama.

Penyelesaian :

```
>load geometry
```

Numerical and symbolic geometry.

```
>setPlotRange(-4.5,4.5,-4.5,4.5);
>A=[-3,-3]; plotPoint(A,"A");
>B=[3,-3]; plotPoint(B,"B");
>C=[3,3]; plotPoint(C,"C");
>D=[-3,3]; plotPoint(D,"D");
>plotSegment(A,B,"");
>splotSegment(B,C,"");
```

```
Function splotSegment not found.  
Try list ... to find functions!  
Error in:  
splotSegment(B,C,""); ...  
^
```

```
>plotSegment(C,D,"");  
>plotSegment(A,D,"");  
>aspect(1);  
>l=angleBisector(A,B,C);  
>m=angleBisector(B,C,D);  
>P=lineIntersection(l,m);  
>color(5); plotLine(l); plotLine(m); color(1);  
>plotPoint(P,"P");
```

Dari gambar diatas terlihat bahwa keempat garis bagi sudutnya bertemu di satu titik yaitu titik P.

```
>r=norm(P-projectToLine(P,lineThrough(A,B)));  
>plotCircle(circleWithCenter(P,r),"Lingkaran dalam segiempat ABCD");
```

Dari gambar diatas, terlihat bahwa sisi-sisinya merupakan garis singgung lingkaran yang sama yaitu lingkaran dalam segiempat.

Akan ditunjukkan bahwa hasil kali panjang sisi-sisi yang berhadapan sama.

```
>AB=norm(A-B) //panjang sisi AB
```

6

```
>CD=norm(C-D) //panjang sisi CD
```

6

```
>AD=norm(A-D) //panjang sisi AD
```

6

```
>BC=norm(B-C) //panjang sisi BC
```

6

```
>AB.CD
```

36

```
>AD.BC
```

36

Terbukti bahwa hasil kali panjang sisi-sisi yang berhadapan sama yaitu 36. Jadi dapat dipastikan bahwa segiempat tersebut merupakan segiempat garis singgung.

4. Gambarlah suatu ellips jika diketahui kedua titik fokusnya, misalnya P dan Q. Ingat ellips dengan fokus P dan Q adalah tempat kedudukan titik-titik yang jumlah jarak ke P dan ke Q selalu sama (konstan).

Penyelesaian :

Diketahui kedua titik fokus  $P = [-1,-1]$  dan  $Q = [1,-1]$

```
>P=[-1,-1]; Q=[1,-1];
>function d1(x,y):=sqrt((x-P[1])^2+(y-P[2])^2)
>Q=[1,-1]; function d2(x,y):=sqrt((x-P[1])^2+(y-P[2])^2)+sqrt((x-Q[1])^2+(y-Q[2])^2)
>fcontour("d2",xmin=-2,xmax=2,ymin=-3,ymax=1,hue=1):
```

Grafik yang lebih menarik

```
>plot3d("d2",xmin=-2,xmax=2,ymin=-3,ymax=1):
```

Batasan ke garis PQ

```
>plot2d("abs(x+1)+abs(x-1)",xmin=-3,xmax=3):
```

5. Gambarlah suatu hiperbola jika diketahui kedua titik fokusnya, misalnya P dan Q. Ingat ellips dengan fokus P dan Q adalah tempat kedudukan titik-titik yang selisih jarak ke P dan ke Q selalu sama (konstan).

Penyelesaian :

```
>P=[-1,-1]; Q=[1,-1];
>function d1(x,y):=sqrt((x-p[1])^2+(y-p[2])^2)
>Q=[1,-1]; function d2(x,y):=sqrt((x-P[1])^2+(y-P[2])^2)+sqrt((x+Q[1])^2+(y+Q[2])^2)
>fcontour("d2",xmin=-2,xmax=2,ymin=-3,ymax=1,hue=1):
```

Grafik yang lebih menarik

```
>plot3d("d2",xmin=-2,xmax=2,ymin=-3,ymax=1):
>plot2d("abs(x+1)+abs(x-1)",xmin=-3,xmax=3):
```

Dalam buku catatan ini, kami mendemonstrasikan plot statistik utama, pengujian, dan distribusi di Euler.

Mari kita mulai dengan beberapa statistik deskriptif. Ini bukan pengantar statistik. Jadi, Anda mungkin memerlukan beberapa latar belakang untuk memahami detailnya.

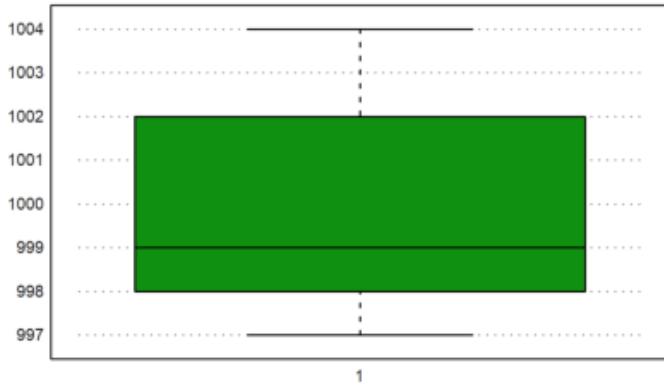
Asumsikan pengukuran berikut. Kami ingin menghitung nilai rata-rata dan standar deviasi yang diukur.

```
>M=[1000,1004,998,997,1002,1001,998,1004,998,997]; ...
>mean(M), dev(M),
```

```
999.9
2.72641400622
```

Kita dapat memplot plot kotak-dan-kumis untuk data. Dalam kasus kami tidak ada outlier.

```
>aspect(1.75); boxplot(M):
```



Kami menghitung probabilitas bahwa suatu nilai lebih besar dari 1005, dengan asumsi nilai terukur dan distribusi normal.

Semua fungsi untuk distribusi di Euler diakhiri dengan ...dis dan menghitung distribusi probabilitas kumulatif (CPF).

$$\text{normaldis}(x,m,d) = \int_{-\infty}^x \frac{1}{d\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{t-m}{d}\right)^2} dt.$$

Kami mencetak hasilnya dalam % dengan akurasi 2 digit menggunakan fungsi cetak.

```
>print((1-normaldis(1005,mean(M),dev(M)))*100,2,unit=" %")
```

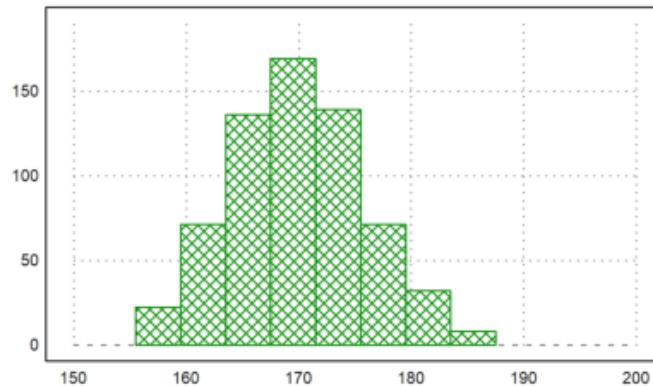
3.07 %

Untuk contoh berikutnya, kami mengasumsikan jumlah pria berikut dalam rentang ukuran yang diberikan.

```
>r=155.5:4:187.5; v=[22,71,136,169,139,71,32,8];
```

Berikut adalah plot distribusinya.

```
>plot2d(r,v,a=150,b=200,c=0,d=190,bar=1,style="\\"");
```



Kita bisa memasukkan data mentah tersebut ke dalam sebuah tabel.

Tabel adalah metode untuk menyimpan data statistik. Tabel kita harus berisi tiga kolom: Awal jangkauan, akhir jangkauan, jumlah orang dalam jangkauan.

Tabel dapat dicetak dengan header. Kami menggunakan vektor string untuk mengatur header.

```
>T:=r[1:8] ' | r[2:9] ' | v'; writetable(T,labc=["from","to","count"])
```

from	to	count
155.5	159.5	22
159.5	163.5	71
163.5	167.5	136
167.5	171.5	169
171.5	175.5	139
175.5	179.5	71
179.5	183.5	32
183.5	187.5	8

Jika kita membutuhkan nilai rata-rata dan statistik lain dari ukuran, kita perlu menghitung titik tengah rentang. Kita dapat menggunakan dua kolom pertama dari tabel kita untuk ini.

Sumbul ”|” digunakan untuk memisahkan kolom, fungsi ”writetable” digunakan untuk menulis tabel, dengan opsiion ”labc” adalah untuk menentukan header kolom.

```
>(T[,1]+T[,2])/2 // the midpoint of each interval
```

```
157.5  
161.5  
165.5  
169.5  
173.5  
177.5  
181.5  
185.5
```

Tetapi lebih mudah, untuk melipat rentang dengan vektor [1/2.1/2].

```
>M=fold(r,[0.5,0.5])
```

```
[157.5, 161.5, 165.5, 169.5, 173.5, 177.5, 181.5, 185.5]
```

Sekarang kita dapat menghitung mean dan deviasi sampel dengan frekuensi yang diberikan.

```
>{m,d}=meandev(M,v); m, d,
```

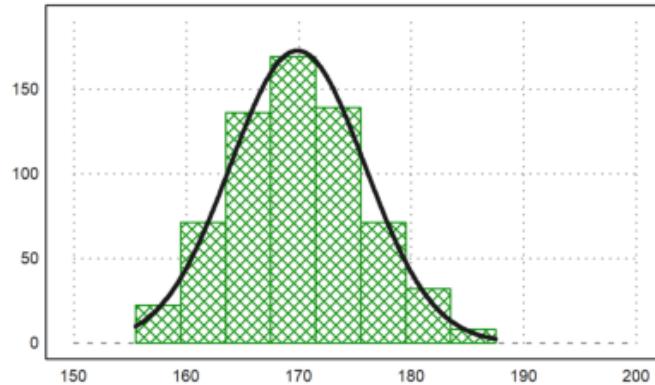
```
169.901234568  
5.98912964449
```

Mari kita tambahkan distribusi normal dari nilai-nilai ke plot batang di atas. Rumus untuk distribusi normal dengan mean m dan standar deviasi d adalah:

$$y = \frac{1}{d\sqrt{2\pi}} e^{-\frac{(x-m)^2}{2d^2}}.$$

Karena nilainya antara 0 dan 1, untuk memplotnya pada bar plot harus dikalikan dengan 4 kali jumlah total data.

```
>plot2d("qnormal(x,m,d)*sum(v)*4", ...
> xmin=min(r),xmax=max(r),thickness=3,add=1):
```



Di direktori notebook ini Anda menemukan file dengan tabel. Data tersebut mewakili hasil survei. Berikut adalah empat baris pertama dari file tersebut. Data berasal dari buku online Jerman "Einführung in die Statistik mit R" oleh A. Handl.

```
>printfile("table.dat",4);
```

```
Could not open the file
table.dat
for reading!
Try "trace errors" to inspect local variables after errors.
printfile:
  open(filename,"r");
```

Tabel berisi 7 kolom angka atau token (string). Kami ingin membaca tabel dari file. Pertama, kami menggunakan terjemahan kami sendiri untuk token.

Untuk ini, kami mendefinisikan set token. Fungsi `strtokens()` mendapatkan vektor string token dari string yang diberikan.

```
>mf:=["m","f"]; yn:=["y","n"]; ev:=strtokens("g vg m b vb");
```

Sekarang kita membaca tabel dengan terjemahan ini.

Argumen tok2, tok4 dll. adalah terjemahan dari kolom tabel. Argumen ini tidak ada dalam daftar parameter readtable(), jadi Anda harus menyediakannya dengan ":=".

```
>{MT,hd}=readtable("table.dat",tok2:=mf,tok4:=yn,tok5:=ev,tok7:=yn);
```

```
Could not open the file
table.dat
for reading!
Try "trace errors" to inspect local variables after errors.
readtable:
    if filename!=none then open(filename,"r"); endif;
```

```
>load over statistics;
```

Untuk mencetak, kita perlu menentukan set token yang sama. Kami mencetak empat baris pertama saja.

```
>writetable(MT[1:4],labc=hd,wc=5,tok2:=mf,tok4:=yn,tok5:=ev,tok7:=yn);
```

```
MT is not a variable!
Error in:
writetable(MT[1:4],labc=hd,wc=5,tok2:=mf,tok4:=yn,tok5:=ev,tok ...  
^
```

Titik ”.” mewakili nilai-nilai, yang tidak tersedia.

Jika kita tidak ingin menentukan token untuk terjemahan terlebih dahulu, kita hanya perlu menentukan, kolom mana yang berisi token dan bukan angka.

```
>ctok=[2,4,5,7]; {MT,hd,tok}=readtable("table.dat",ctok=ctok);
```

```
Could not open the file
table.dat
for reading!
Try "trace errors" to inspect local variables after errors.
readtable:
    if filename!=none then open(filename,"r"); endif;
```

Fungsi readtable() sekarang mengembalikan satu set token.

```
>tok
```

```
Variable tok not found!
Error in:
tok ...
^
```

Tabel berisi entri dari file dengan token yang diterjemahkan ke angka.

String khusus NA="." ditafsirkan sebagai "Tidak Tersedia", dan mendapatkan NAN (bukan angka) dalam tabel. Terjemahan ini dapat diubah dengan parameter NA, dan NAval.

```
>MT[1]
```

```
MT is not a variable!
Error in:
MT[1] ...
^
```

Berikut isi tabel dengan angka yang belum diterjemahkan.

```
>writetable(MT,wc=5)
```

```
Variable or function MT not found.
Error in:
writetable(MT,wc=5) ...
^
```

Untuk kenyamanan, Anda dapat memasukkan output readtable() ke dalam daftar.

```
>Table={{readtable("table.dat",ctok=ctok)}};


```

```
Could not open the file
table.dat
for reading!
Try "trace errors" to inspect local variables after errors.
readtable:
  if filename!=none then open(filename,"r"); endif;
```

Menggunakan kolom token yang sama dan token yang dibaca dari file, kita dapat mencetak tabel. Kita dapat menentukan ctok, tok, dll. Atau menggunakan daftar Tabel.

```
>writetable(Table,ctok=ctok,wc=5);
```

```
Variable or function Table not found.
Error in:
writetable(Table,ctok=ctok,wc=5); ...  
^
```

Fungsi `tablecol()` mengembalikan nilai kolom tabel, melewatkkan setiap baris dengan nilai NAN("." dalam file), dan indeks kolom, yang berisi nilai-nilai ini.

```
>{c,i}=tablecol(MT,[5,6]);
```

```
Variable or function MT not found.  
Error in:  
{c,i}=tablecol(MT,[5,6]); ...  
^
```

Kita dapat menggunakan ini untuk mengekstrak kolom dari tabel untuk tabel baru.

```
>j=[1,5,6]; writetable(MT[i,j],labc=hd[j],ctok=[2],tok=tok)
```

```
Variable or function i not found.  
Error in:  
j=[1,5,6]; writetable(MT[i,j],labc=hd[j],ctok=[2],tok=tok) ...  
^
```

Tentu saja, kita perlu mengekstrak tabel itu sendiri dari daftar Tabel dalam kasus ini.

```
>MT=Table[1];
```

```
Table is not a variable!
Error in:
MT=Table[1];
^
```

Tentu saja, kita juga dapat menggunakan untuk menentukan nilai rata-rata kolom atau nilai statistik lainnya.

```
>mean(tablecol(MT,6))
```

```
Variable or function MT not found.
Error in:
mean(tablecol(MT,6))
^
```

Fungsi getstatistics() mengembalikan elemen dalam vektor, dan jumlahnya. Kami menerapkannya pada nilai "m" dan "f" di kolom kedua tabel kami.

```
>{xu,count}=getstatistics(tablecol(MT,2)); xu, count,
```

```
Variable or function MT not found.  
Error in:  
{xu,count}=getstatistics(tablecol(MT,2)); xu, count, ...  
^
```

Kami dapat mencetak hasilnya dalam tabel baru.

```
>writetable(count',labr=tok[xu])
```

```
Variable count not found!  
Error in:  
writetable(count',labr=tok[xu]) ...  
^
```

Fungsi selecttable() mengembalikan tabel baru dengan nilai dalam satu kolom yang dipilih dari vektor indeks. Pertama kita mencari indeks dari dua nilai kita di tabel token.

```
>v:=indexof(tok,["g","vg"])
```

```
Variable or function tok not found.  
Error in:  
v:=indexof(tok,["g","vg"]) ...  
^
```

Sekarang kita dapat memilih baris tabel, yang memiliki salah satu nilai dalam v di baris ke-5.

```
>MT1:=MT[selectrows(MT,5,v)]; i:=sortedrows(MT1,5);
```

```
Variable or function MT not found.  
Error in:  
MT1:=MT[selectrows(MT,5,v)]; i:=sortedrows(MT1,5); ...  
^
```

Sekarang kita dapat mencetak tabel, dengan nilai yang diekstrak dan diurutkan di kolom ke-5.

```
>writetable(MT1[i],labc=hd,ctok=ctok,tok=tok,wc=7);
```

```
Variable or function i not found.  
Error in:  
writetable(MT1[i],labc=hd,ctok=ctok,tok=tok,wc=7); ...  
^
```

Untuk statistik berikutnya, kami ingin menghubungkan dua kolom tabel. Jadi kami mengekstrak kolom 2 dan 4 dan mengurutkan tabel.

```
>i=sortedrows(MT,[2,4]); ...  
> writetable(tablecol(MT[i],[2,4])',ctok=[1,2],tok=tok)
```

```
Variable or function MT not found.  
Error in:  
i=sortedrows(MT,[2,4]);      writetable(tablecol(MT[i],[2,4])',c ...  
^
```

Dengan getstatistics(), kita juga bisa menghubungkan hitungan dalam dua kolom tabel satu sama lain.

```
>MT24=tablecol(MT,[2,4]); ...
>{xu1,xu2,count}=getstatistics(MT24[1],MT24[2]); ...
>writetable(count,labr=tok[xu1],labc=tok[xu2])
```

```
Variable or function MT not found.
Error in:
MT24=tablecol(MT,[2,4]); {xu1,xu2,count}=getstatistics(MT24[1] ...
```

Tabel dapat ditulis ke file.

```
>filename="test.dat"; ...
>writetable(count,labr=tok[xu1],labc=tok[xu2],file=filename);
```

```
Variable or function count not found.
Error in:
filename="test.dat"; writetable(count,labr=tok[xu1],labc=tok[x ...
```

Kemudian kita bisa membaca tabel dari file tersebut.

```
>{MT2,hd,tok2,hdr}=readtable(filename,>clabs,>rlabs); ...
>writetable(MT2,labr=hdr,labc=hd)
```

```
Could not open the file
test.dat
for reading!
Try "trace errors" to inspect local variables after errors.
readtable:
    if filename!=none then open(filename,"r"); endif;
```

Dan hapus file tersebut.

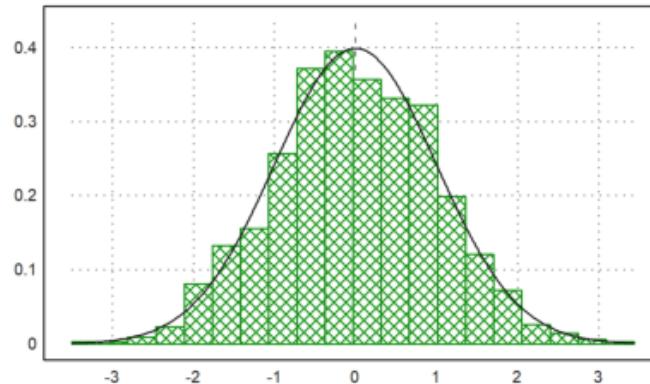
```
>fileremove(filename);
```

## Distribusi

---

Dengan plot2d, terdapat metode yang sangat mudah untuk memplot sebaran data eksperimen.

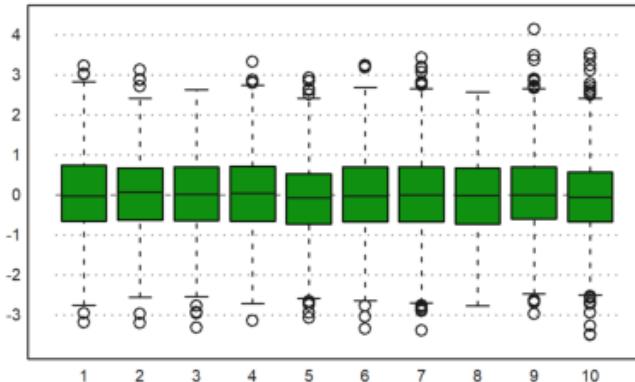
```
>p=normal(1,1000); //1000 random normal-distributed sample p  
>plot2d(p,distribution=20,style="\\\"/\\\""); // plot the random sample p  
>plot2d("qnormal(x,0,1)",add=1); // add the standard normal distribution plot
```



Harap perhatikan perbedaan antara plot batang (sampel) dan kurva normal (distribusi nyata). Masukkan kembali tiga perintah untuk melihat hasil pengambilan sampel lainnya.

Berikut adalah perbandingan 10 simulasi dari 1000 nilai terdistribusi normal menggunakan apa yang disebut plot kotak. Plot ini menunjukkan median, kuartil 25% dan 75%, nilai minimal dan maksimal, dan outlier.

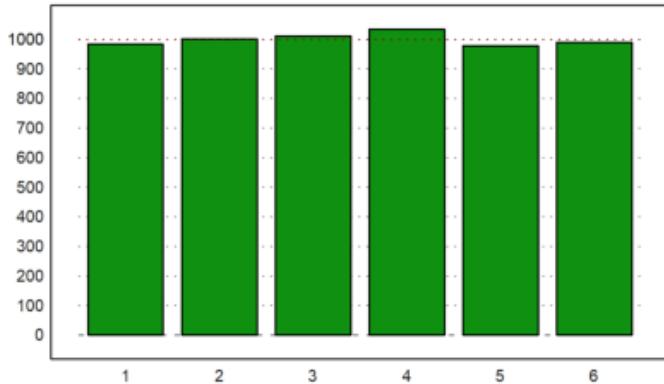
```
>p=normal(10,1000); boxplot(p):
```



Untuk membangkitkan bilangan bulat acak, Euler memiliki intrarandom. Mari kita simulasikan lemparan dadu dan plot distribusinya.

Kita menggunakan fungsi getmultiplicities(v,x), yang menghitung seberapa sering elemen v muncul di x. Kemudian kita memplot hasilnya menggunakan columnplot().

```
>k=intrandom(1,6000,6); ...
>columnplot(getmultiplicities(1:6,k)); ...
>ygrid(1000,color=red):
```



Sementara intrandom(n,m,k) mengembalikan bilangan bulat yang terdistribusi secara seragam dari 1 ke k, dimungkinkan untuk menggunakan distribusi bilangan bulat lain yang diberikan dengan randpint().

Dalam contoh berikut, probabilitas untuk 1,2,3 masing-masing adalah 0,4,0,1,0,5.

```
>randpint(1,1000,[0.4,0.1,0.5]); getmultiplicities(1:3,%)
```

[378, 102, 520]

Euler dapat menghasilkan nilai acak dari lebih banyak distribusi. Coba lihat referensinya.

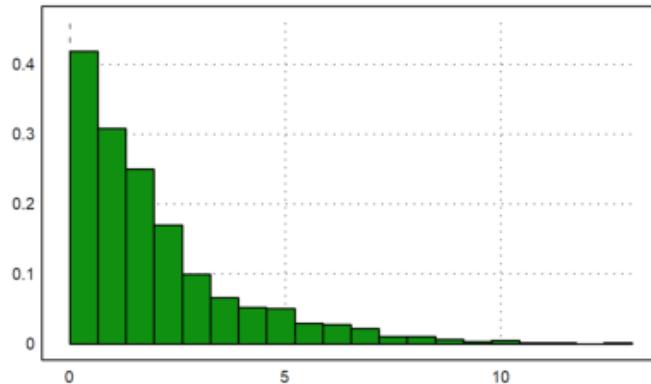
Misalnya, kami mencoba distribusi eksponensial. Variabel acak kontinu X dikatakan memiliki distribusi eksponensial, jika PDF-nya diberikan oleh

$$f_X(x) = \lambda e^{-\lambda x}, \quad x > 0, \quad \lambda > 0,$$

dengan parameter

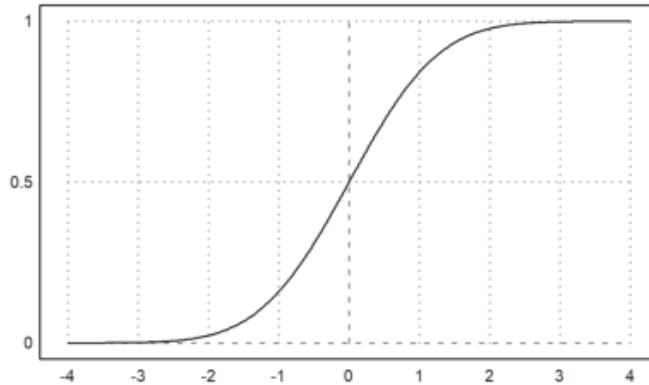
$$\lambda = \frac{1}{\mu}, \quad \mu \text{ adalah rata-rata, dan dilambangkan dengan } X \sim \text{Eksponensial}(\lambda).$$

```
>plot2d(randexponential(1,1000,2),>distribution):
```



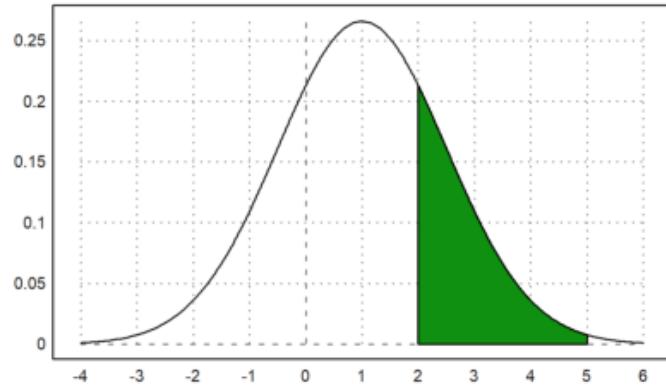
Untuk banyak distribusi, Euler dapat menghitung fungsi distribusi dan inversnya.

```
>plot2d("normaldis",-4,4):
```



Berikut ini adalah salah satu cara untuk memplot kuantil.

```
>plot2d("qnormal(x,1,1.5)",-4,6); ...  
>plot2d("qnormal(x,1,1.5)",a=2,b=5,>add,>filled):
```



$$\text{normaldis}(x,m,d) = \int_{-\infty}^x \frac{1}{d\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{t-m}{d})^2} dt.$$

Probabilitas untuk berada di area hijau adalah sebagai berikut.

```
>normaldis(5,1,1.5)-normaldis(2,1,1.5)
```

0.248662156979

Ini dapat dihitung secara numerik dengan integral berikut.

$$\int_2^5 \frac{1}{1.5\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-1}{1.5})^2} dx.$$

```
>gauss("qnormal(x,1,1.5)",2,5)
```

0.248662156979

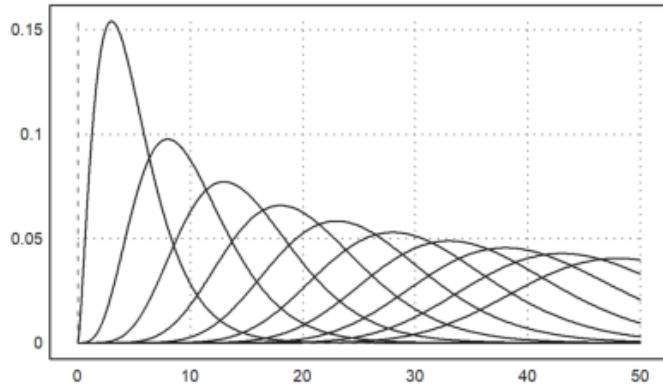
Mari kita bandingkan distribusi binomial dengan distribusi normal mean dan deviasi yang sama. Fungsi invbindis() memecahkan interpolasi linier antara nilai bilangan bulat.

```
>invbindis(0.95,1000,0.5), invnormaldis(0.95,500,0.5*sqrt(1000))
```

525.516721219  
526.007419394

Fungsi qdis() adalah kepadatan distribusi chi-kuadrat. Seperti biasa, Euler memetakan vektor ke fungsi ini. Jadi kita mendapatkan plot dari semua distribusi chi-kuadrat dengan derajat 5 sampai 30 dengan mudah dengan cara berikut.

```
>plot2d("qchidis(x,(5:5:50)'),0,50):
```



Euler memiliki fungsi yang akurat untuk mengevaluasi distribusi. Mari kita periksa chidis() dengan integral.

Penamaan mencoba untuk konsisten. Misalnya.,

- distribusi chi-kuadrat adalah chidis(),
- fungsi kebalikannya adalah invchidis(),
- densitasnya adalah qchidis().

Pelengkap distribusi (ekor atas) adalah chicdis().

```
>chidis(1.5,2), integrate("qchidis(x,2)",0,1.5)
```

```
0.527633447259  
0.527633447259
```

## Distribusi Diskrit

---

Untuk menentukan distribusi diskrit Anda sendiri, Anda dapat menggunakan metode berikut.

Pertama kita mengatur fungsi distribusi.

```
>wd = 0|((1:6)+[-0.01,0.01,0,0,0,0])/6
```

```
[0, 0.165, 0.335, 0.5, 0.666667, 0.833333, 1]
```

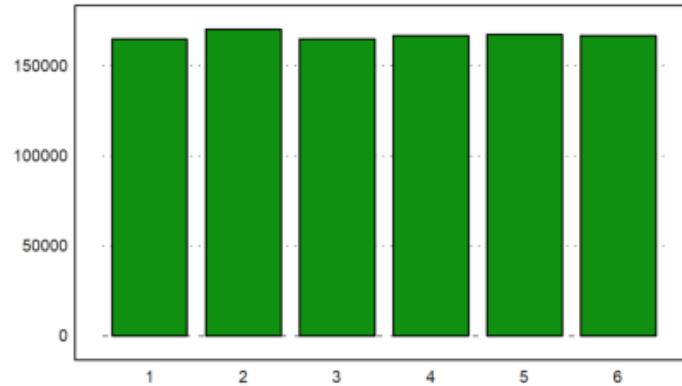
Artinya dengan probabilitas  $wd[i+1]-wd[i]$  kita menghasilkan nilai acak i.

Ini hampir merupakan distribusi yang seragam. Mari kita tentukan generator angka acak untuk ini. Fungsi `find(v,x)` menemukan nilai x dalam vektor v. Fungsi ini juga berlaku untuk vektor x.

```
>function wrongdice (n,m) := find(wd,random(n,m))
```

Kesalahannya sangat halus sehingga kami melihatnya hanya dengan iterasi yang sangat banyak.

```
>columnsplot(getmultiplicities(1:6,wrongdice(1,1000000))):
```



Berikut adalah fungsi sederhana untuk memeriksa distribusi seragam dari nilai 1...K dalam v. Kami menerima hasilnya, jika untuk semua frekuensi

$$\left| f_i - \frac{1}{K} \right| < \frac{\delta}{\sqrt{n}}.$$

```
>function checkrandom (v, delta=1) ...
```

```
K=max(v); n=cols(v);
fr=getfrequencies(v,1:K);
return max(fr/n-1/K)<delta/sqrt(n);
endfunction
```

Memang fungsi menolak distribusi seragam.

```
>checkrandom(wrongdice(1,1000000))
```

0

Dan itu menerima generator acak bawaan.

```
>checkrandom(intrandom(1,1000000,6))
```

1

Kita dapat menghitung distribusi binomial. Pertama ada binomialsum(), yang mengembalikan probabilitas i atau kurang hit dari n percobaan.

```
>bindis(410,1000,0.4)
```

0.751401349654

Fungsi Beta terbalik digunakan untuk menghitung interval kepercayaan Clopper-Pearson untuk parameter p. Level default adalah alfa.

Arti interval ini adalah jika p berada di luar interval, hasil pengamatan 410 dalam 1000 jarang terjadi.

```
>clopperpearson(410,1000)
```

[0.37932, 0.441212]

Perintah berikut adalah cara langsung untuk mendapatkan hasil di atas. Tapi untuk n besar, penjumlahan langsungnya tidak akurat dan lambat.

```
>p=0.4; i=0:410; n=1000; sum(bin(n,i)*p^i*(1-p)^(n-i))
```

0.751401349655

Omong-omong, invbinsum() menghitung kebalikan dari binomialsum().

```
>invbindis(0.75,1000,0.4)
```

409.932733047

Di Bridge, kami mengasumsikan 5 kartu beredar (dari 52) dengan dua tangan (26 kartu). Mari kita hitung probabilitas distribusi yang lebih buruk dari 3:2 (mis. 0:5, 1:4, 4:1, atau 5:0).

```
>2*hypergeomsum(1,5,13,26)
```

0.321739130435

Ada juga simulasi distribusi multinomial.

```
>randmultinomial(10,1000,[0.4,0.1,0.5])
```

381	100	519
376	91	533
417	80	503
440	94	466
406	112	482
408	94	498
395	107	498
399	96	505
428	87	485
400	99	501

## Merencanakan Data

---

Untuk memplot data, kami mencoba hasil pemilu Jerman sejak 1990, yang diukur dalam jumlah kursi.

```
>BW := [ ...
>1990,662,319,239,79,8,17; ...
>1994,672,294,252,47,49,30; ...
>1998,669,245,298,43,47,36; ...
>2002,603,248,251,47,55,2; ...
>2005,614,226,222,61,51,54; ...
>2009,622,239,146,93,68,76; ...
>2013,631,311,193,0,63,64];
```

Untuk partai, kami menggunakan rangkaian nama.

```
>P := ["CDU/CSU", "SPD", "FDP", "Gr", "Li"];
```

Mari kita cetak persentasenya dengan baik.

Pertama, kami mengekstrak kolom yang diperlukan. Kolom 3 sampai 7 adalah kursi masing-masing partai, dan kolom 2 adalah jumlah kursi. kolom adalah tahun pemilihan.

```
>BT:=BW[,3:7]; BT:=BT/sum(BT); YT:=BW[,1]';
```

Kemudian kami mencetak statistik dalam bentuk tabel. Kami menggunakan nama sebagai tajuk kolom, dan tahun sebagai tajuk untuk baris. Lebar default untuk kolom adalah wc=10, tetapi kami lebih memilih hasil yang lebih padat. Kolom akan diperluas untuk label kolom, jika perlu.

```
>writetable(BT*100,wc=6,dc=0,>fixed,labc=P,labr=YT)
```

	CDU/CSU	SPD	FDP	Gr	Li
1990	48	36	12	1	3
1994	44	38	7	7	4
1998	37	45	6	7	5
2002	41	42	8	9	0
2005	37	36	10	8	9
2009	38	23	15	11	12
2013	49	31	0	10	10

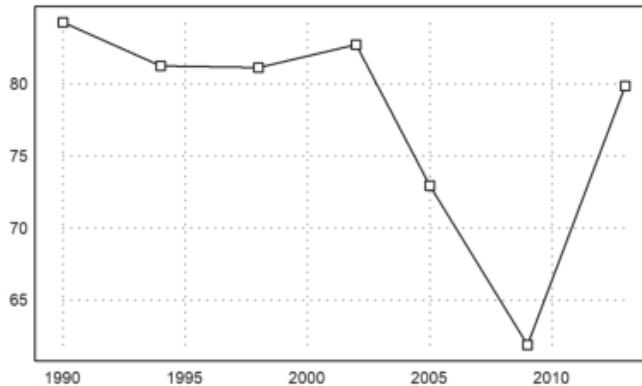
Perkalian matriks berikut mengekstrak jumlah persentase dari dua partai besar yang menunjukkan bahwa partai-partai kecil telah mendapatkan rekaman di parlemen hingga tahun 2009.

```
>BT1:=(BT.[1;1;0;0;0])'*100
```

```
[84.29, 81.25, 81.1659, 82.7529, 72.9642, 61.8971, 79.8732]
```

Ada juga plot statistik sederhana. Kami menggunakananya untuk menampilkan garis dan titik secara bersamaan. Alternatifnya adalah memanggil plot2d dua kali dengan >add.

```
>statplot(YT,BT1,"b"):
```



Tentukan beberapa warna untuk masing-masing pihak.

```
>CP:=[rgb(0.5,0.5,0.5),red,yellow,green,rgb(0.8,0,0)];
```

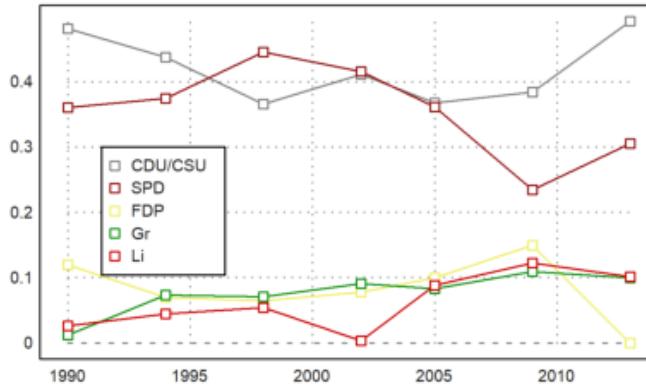
Sekarang kita bisa memplot hasil pemilu 2009 dan mengubahnya menjadi satu plot menggunakan gambar. Kita dapat menambahkan vektor kolom ke setiap plot.

```
>figure(2,1); ...
>figure(1); columnsplot(BW[6,3:7],P,color=CP); ...
>figure(2); columnsplot(BW[6,3:7]-BW[5,3:7],P,color=CP); ...
>figure(0):
```



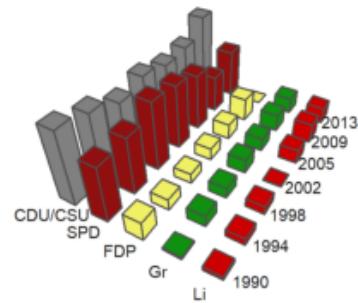
Plot data menggabungkan deretan data statistik dalam satu plot.

```
>J:=BW[,1]'; DP:=BW[,3:7]'; ...
>dataplot(YT,BT',color=CP); ...
>labelbox(P,colors=CP,styles="[]",>points,w=0.2,x=0.3,y=0.4):
```



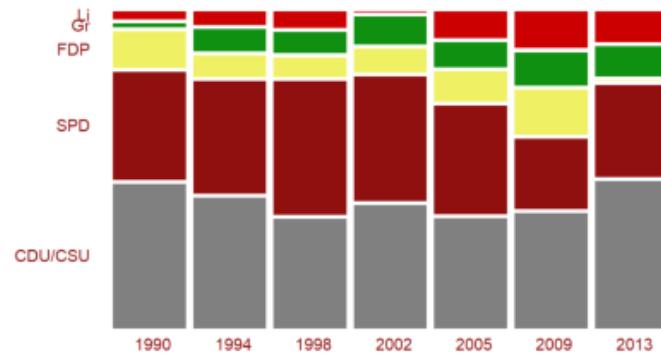
Plot kolom 3D menunjukkan deretan data statistik dalam bentuk kolom. Kami menyediakan label untuk baris dan kolom. sudut adalah sudut pandang.

```
>columnsplot3d(BT,scols=P,srows=YT, ...
> angle=30°,ccols=CP):
```



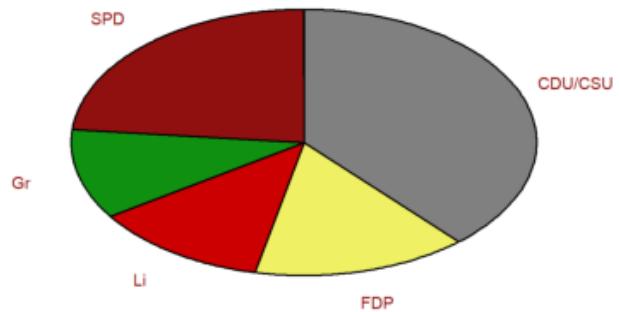
Representasi lainnya adalah plot mozaik. Perhatikan bahwa kolom plot mewakili kolom matriks di sini. Karena panjang label CDU/CSU, kami mengambil jendela yang lebih kecil dari biasanya.

```
>shrinkwindow(>smaller); ...
>mosaicplot(BT',srows=YT,scols=P,color=CP,style="#");
>shrinkwindow():
```



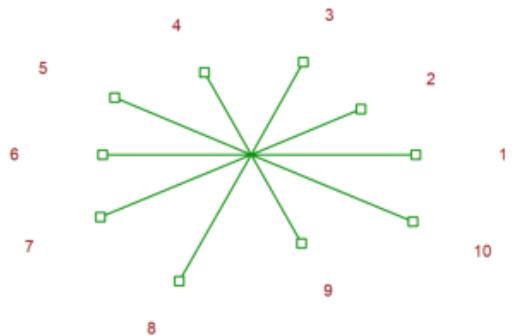
Kita juga bisa membuat diagram lingkaran. Karena hitam dan kuning membentuk koalisi, kami menyusun ulang elemennya.

```
>i=[1,3,5,4,2]; piechart(BW[6,3:7][i],color=CP[i],lab=P[i]):
```



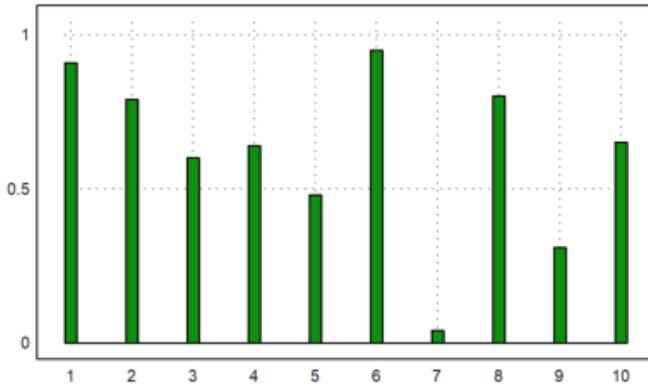
Ini jenis plot lainnya.

```
>starplot(normal(1,10)+4,lab=1:10,>rays):
```



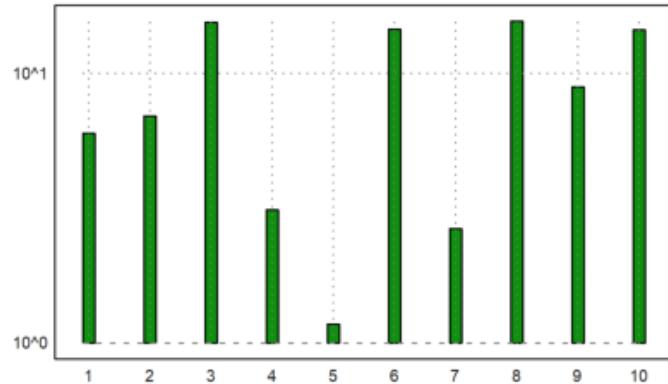
Beberapa plot di plot2d bagus untuk statika. Berikut adalah plot impuls dari data acak, terdistribusi secara seragam di  $[0,1]$ .

```
>plot2d(makeimpulse(1:10,random(1,10)),>bar):
```



Tetapi untuk data yang terdistribusi secara eksponensial, kita mungkin memerlukan plot logaritmik.

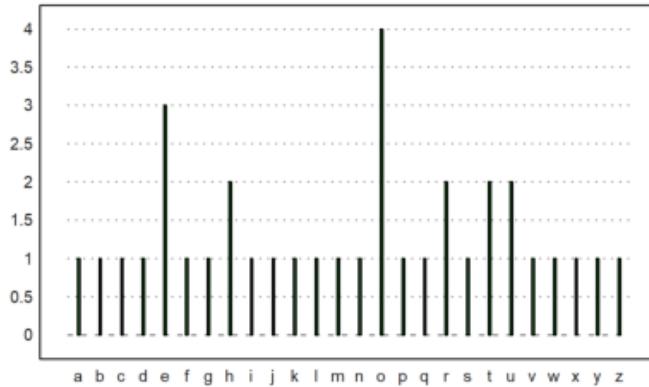
```
>logimpulseplot(1:10,-log(random(1,10))*10):
```



Fungsi `columnplot()` lebih mudah digunakan, karena hanya membutuhkan vektor nilai. Selain itu, ia dapat mengatur labelnya ke apa pun yang kita inginkan, kita telah mendemonstrasikannya di tutorial ini.

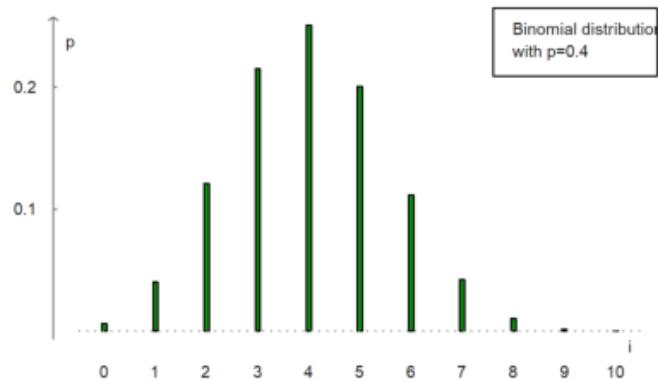
Ini adalah aplikasi lain, di mana kami menghitung karakter dalam sebuah kalimat dan memplot statistik.

```
>v=strtochar("the quick brown fox jumps over the lazy dog"); ...
>w=ascii("a"):ascii("z"); x=getmultiplicities(w,v); ...
>cw=[]; for k=w; cw=cw|char(k); end; ...
>columnsplot(x,lab=cw,width=0.05):
```



Dimungkinkan juga untuk mengatur sumbu secara manual.

```
>n=10; p=0.4; i=0:n; x=bin(n,i)*p^i*(1-p)^(n-i); ...
>columnsplot(x,lab=i,width=0.05,<frame,<grid); ...
>yaxis(0,0:0.1:1,style="->",>left); xaxis(0,style="."); ...
>label("p",0,0.25), label("i",11,0); ...
>textbox(["Binomial distribution","with p=0.4"]):
```



Berikut ini adalah cara memplot frekuensi bilangan dalam vektor.

Kami membuat vektor bilangan acak bilangan bulat 1 hingga 6.

```
>v:=intrandom(1,10,10)
```

```
[8, 5, 8, 8, 6, 8, 8, 3, 5, 5]
```

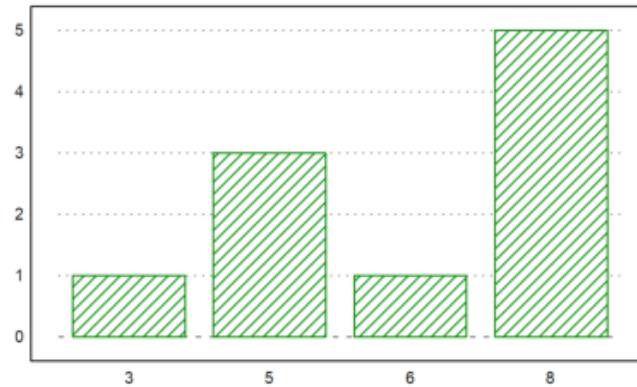
Kemudian ekstrak angka unik di v.

```
>vu:=unique(v)
```

```
[3, 5, 6, 8]
```

Dan plot frekuensi dalam plot kolom.

```
>columnsplot(getmultiplicities(vu,v),lab=vu,style="/"):
```



Kami ingin menunjukkan fungsi untuk distribusi nilai empiris.

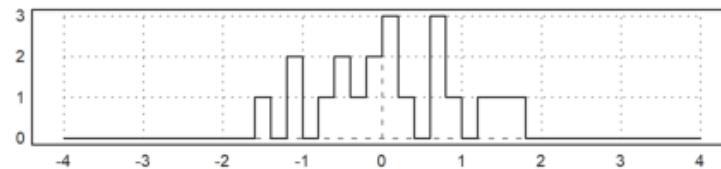
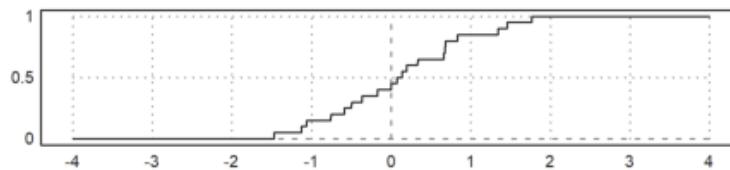
```
>x=normal(1,20);
```

Fungsi empdist(x,vs) membutuhkan array nilai yang diurutkan. Jadi kita harus mengurutkan x sebelum kita dapat menggunakannya.

```
>xs=sort(x);
```

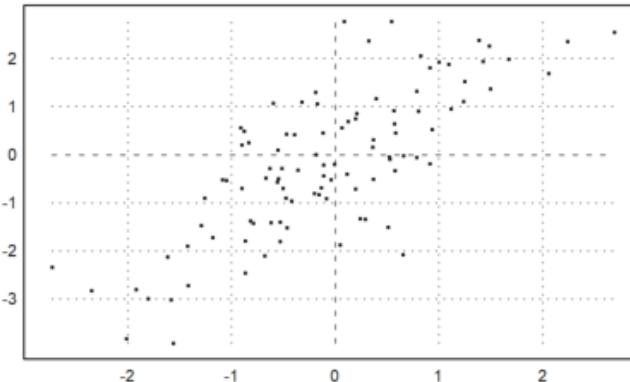
Kemudian kami memplot distribusi empiris dan beberapa batang kepadatan menjadi satu plot. Alih-alih plot batang untuk distribusi, kali ini kami menggunakan plot gigi gergaji.

```
>figure(2,1); ...
>figure(1); plot2d("empdist",-4,4;xs); ...
>figure(2); plot2d(histo(x,v=-4:0.2:4,<bar)); ...
>figure(0):
```



Plot pencar mudah dilakukan di Euler dengan plot titik biasa. Grafik berikut menunjukkan bahwa X dan X+Y jelas berkorelasi positif.

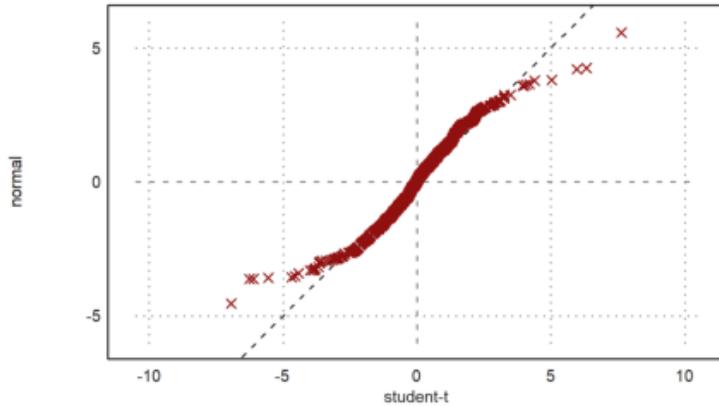
```
>x=normal(1,100); plot2d(x,x+rotright(x),>points,style=".."):
```



Seringkali, kami ingin membandingkan dua sampel dari distribusi yang berbeda. Ini dapat dilakukan dengan plot kuantil-kuantil.

Untuk pengujian, kami mencoba distribusi student-t dan distribusi eksponensial.

```
>x=randt(1,1000,5); y=randnormal(1,1000,mean(x),dev(x)); ...
>plot2d("x",r=6,style="--",yl="normal",xl="student-t",>vertical); ...
>plot2d(sort(x),sort(y),>points,color=red,style="x",>add):
```



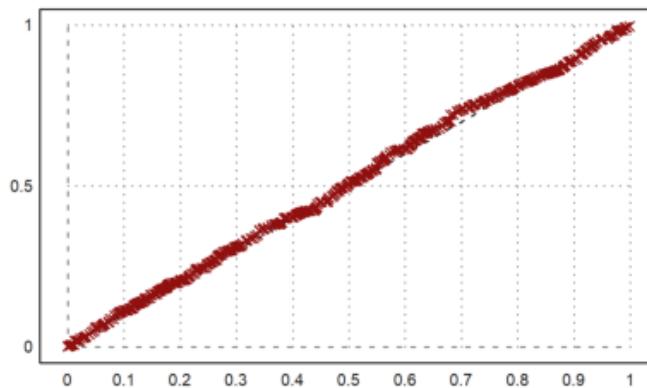
Plot jelas menunjukkan bahwa nilai terdistribusi normal cenderung lebih kecil di ujung ekstrim.

Jika kita memiliki dua distribusi dengan ukuran berbeda, kita dapat memperluas yang lebih kecil atau mengecilkan yang lebih besar. Fungsi berikut ini baik untuk keduanya. Dibutuhkan nilai median dengan persentase antara 0 dan 1.

```
>function medianexpand (x,n) := median(x,p=linspace(0,1,n-1));
```

Mari kita bandingkan dua distribusi yang sama.

```
>x=random(1000); y=random(400); ...
>plot2d("x",0,1,style="--"); ...
>plot2d(sort(medianexpand(x,400)),sort(y),>points,color=red,style="x",>add):
```



## Regresi dan Korelasi

---

Regresi linier dapat dilakukan dengan fungsi polyfit() atau berbagai fungsi fit.

Sebagai permulaan, kami menemukan garis regresi untuk data univariat dengan polyfit(x,y,1).

```
>x=1:10; y=[2,3,1,5,6,3,7,8,9,8]; writetable(x'y',labc=["x","y"])
```

x	y
1	2
2	3
3	1
4	5
5	6
6	3
7	7
8	8
9	9
10	8

Kami ingin membandingkan kecocokan yang tidak berbobot dan berbobot. Pertama koefisien fit linier.

```
>p=polyfit(x,y,1)
```

[0.733333, 0.812121]

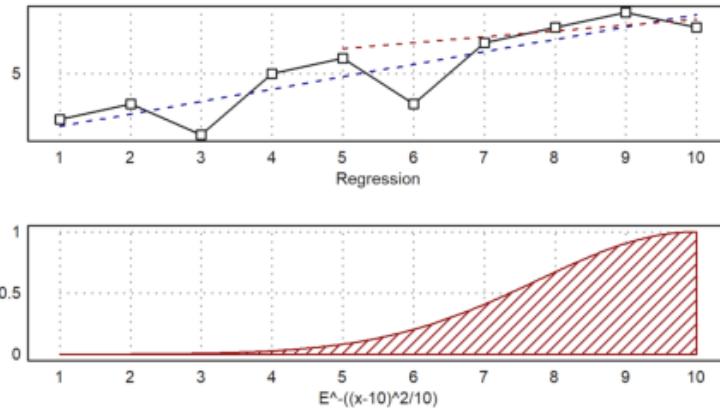
Sekarang koefisien dengan bobot yang menekankan nilai terakhir.

```
>w &= "exp(-(x-10)^2/10)"; pw=polyfit(x,y,1,w=w(x))
```

```
[4.71566,  0.38319]
```

Kami memasukkan semuanya ke dalam satu plot untuk poin dan garis regresi, dan untuk bobot yang digunakan.

```
>figure(2,1); ...
>figure(1); statplot(x,y,"b",xl="Regression"); ...
> plot2d("evalpoly(x,p)",>add,color=blue,style="--"); ...
> plot2d("evalpoly(x,pw)",5,10,>add,color=red,style="--"); ...
>figure(2); plot2d(w,1,10,>filled,style="/",fillcolor=red, xl=w); ...
>figure(0):
```



Sebagai contoh lain kita membaca survei siswa, umur mereka, umur orang tua mereka dan jumlah saudara dari sebuah file.

Tabel ini berisi "m" dan "f" di kolom kedua. Kami menggunakan variabel tok2 untuk menyetel terjemahan yang tepat alih-alih membiarkan readtable() mengumpulkan terjemahan.

```
>{MS,hd}:=readtable("table1.dat",tok2:=[ "m", "f"]);  ...
>writetable(MS,labc=hd,tok2:=[ "m", "f"]);
```

```
Could not open the file
table1.dat
for reading!
Try "trace errors" to inspect local variables after errors.
readtable:
    if filename!=none then open(filename,"r"); endif;
```

Bagaimana usia bergantung satu sama lain? Kesan pertama berasal dari sebar berpasangan.

```
>scatterplots(tablecol(MS,3:5),hd[3:5]):
```

```
Variable or function MS not found.  
Error in:  
scatterplots(tablecol(MS,3:5),hd[3:5]): ...  
^
```

Jelas terlihat bahwa usia ayah dan ibu saling bergantung. Mari kita tentukan dan plot garis regresi.

```
>cs:=MS[,4:5]'; ps:=polyfit(cs[1],cs[2],1)
```

```
MS is not a variable!  
Error in:  
cs:=MS[,4:5]'; ps:=polyfit(cs[1],cs[2],1) ...  
^
```

Ini jelas model yang salah. Garis regresi adalah  $s=17+0,74t$ , di mana t adalah umur ibu dan s adalah umur ayah. Perbedaan usia mungkin sedikit bergantung pada usia, tetapi tidak terlalu banyak.

Sebaliknya, kami mencurigai fungsi seperti  $s=a+t$ . Maka a adalah rata-rata dari s-t. Ini adalah perbedaan usia rata-rata antara ayah dan ibu.

```
>da:=mean(cs[2]-cs[1])
```

```
cs is not a variable!
Error in:
da:=mean(cs[2]-cs[1]) ...  
^
```

Mari kita plot ini menjadi satu plot pencar.

```
>plot2d(cs[1],cs[2],>points); ...
>plot2d("evalpoly(x,ps)",color=red,style=".",>add); ...
>plot2d("x+da",color=blue,>add):
```

```
cs is not a variable!
Error in:
plot2d(cs[1],cs[2],>points);  plot2d("evalpoly(x,ps)",color=re ...
^
```

Berikut adalah plot kotak dari dua zaman. Ini hanya menunjukkan, bahwa usianya berbeda.

```
>boxplot(cs, ["mothers", "fathers"]):
```

```
Variable or function cs not found.  
Error in:  
boxplot(cs, ["mothers", "fathers"]): ...  
^
```

Menariknya, perbedaan median tidak sebesar perbedaan rata-rata.

```
>median(cs[2])-median(cs[1])
```

```
cs is not a variable!  
Error in:  
median(cs[2])-median(cs[1]) ...  
^
```

Koefisien korelasi menunjukkan korelasi positif.

```
>correl(cs[1],cs[2])
```

```
cs is not a variable!  
Error in:  
correl(cs[1],cs[2]) ...  
^
```

Korelasi peringkat adalah ukuran untuk urutan yang sama di kedua vektor. Ini juga cukup positif.

```
>rankcorrel(cs[1],cs[2])
```

```
cs is not a variable!
Error in:
rankcorrel(cs[1],cs[2]) ...  
^
```

## Membuat Fungsi baru

---

Tentu saja, bahasa EMT dapat digunakan untuk memprogram fungsi-fungsi baru. Misalnya, kita mendefinisikan fungsi skewness.

$$\text{sk}(x) = \frac{\sqrt{n} \sum_i (x_i - m)^3}{(\sum_i (x_i - m)^2)^{3/2}}$$

di mana m adalah rata-rata dari x.

```
>function skew (x:vector) ...  
  
m=mean(x);  
return sqrt(cols(x))*sum((x-m)^3)/(sum((x-m)^2))^(3/2);  
endfunction
```

Seperti yang Anda lihat, kita dapat dengan mudah menggunakan bahasa matriks untuk mendapatkan implementasi yang sangat singkat dan efisien. Mari kita coba fungsi ini.

```
>data=normal(20); skew(normal(10))
```

-0.198710316203

Ini adalah fungsi lain, yang disebut koefisien kemiringan Pearson.

```
>function skew1 (x) := 3*(mean(x)-median(x))/dev(x)
>skew1(data)
```

-0.0801873249135

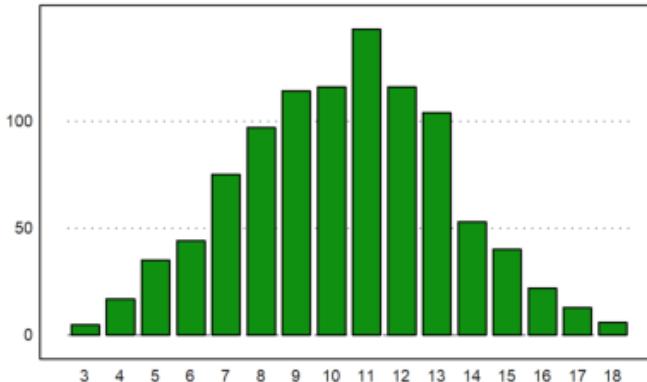
Euler dapat digunakan untuk mensimulasikan kejadian acak. Kita telah melihat contoh sederhana di atas. Ini satu lagi, yang mensimulasikan 1000 kali 3 lemparan dadu, dan meminta distribusi jumlahnya.

```
>ds:=sum(intrandom(1000,3,6)); fs=getmultiplicities(3:18,ds)
```

```
[5, 17, 35, 44, 75, 97, 114, 116, 143, 116, 104, 53, 40,  
22, 13, 6]
```

Kita bisa merencanakan ini sekarang.

```
>columnsplot(fs,lab=3:18):
```



Untuk menentukan distribusi yang diharapkan tidak begitu mudah. Kami menggunakan rekursi lanjutan untuk ini.

Fungsi berikut menghitung banyaknya cara bilangan k dapat dinyatakan sebagai jumlah dari n bilangan dalam rentang 1 sampai m. Ini bekerja secara rekursif dengan cara yang jelas.

```
>function map countways (k; n, m) ...
if n==1 then return k>=1 && k<=m
else
  sum=0;
  loop 1 to m; sum=sum+countways(k-#,n-1,m); end;
  return sum;
end;
endfunction
```

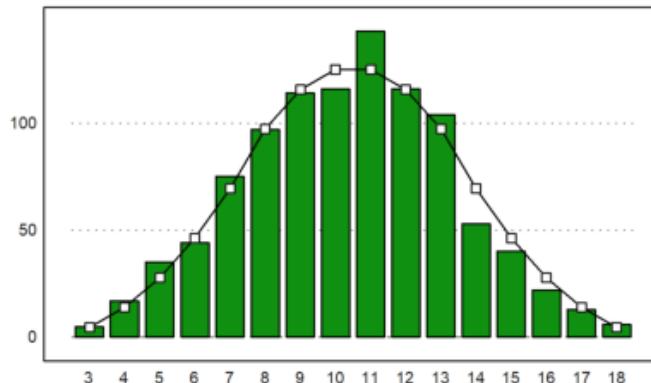
Inilah hasil lemparan dadu sebanyak tiga kali.

```
>cw=countways(3:18,3,6)
```

```
[1,  3,  6,  10,  15,  21,  25,  27,  27,  25,  21,  15,  10,  6,  3,  
1]
```

Kami menambahkan nilai yang diharapkan ke plot.

```
>plot2d(cw/6^3*1000,>add); plot2d(cw/6^3*1000,>points,>add):
```



Untuk simulasi lain, penyimpangan nilai rata-rata n 0-1-variabel acak terdistribusi normal adalah  $1/\sqrt{n}$ .

```
>longformat; 1/sqrt(10)
```

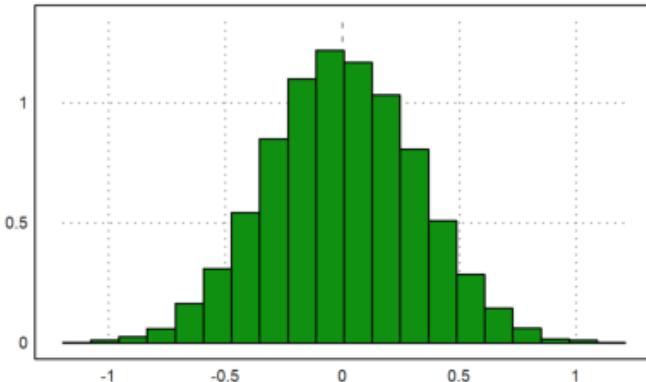
0.316227766017

Mari kita periksa ini dengan simulasi. Kami menghasilkan 10.000 kali 10 vektor acak.

```
>M=normal(10000,10); dev(mean(M)')
```

0.319493614817

```
>plot2d(mean(M)',>distribution):
```



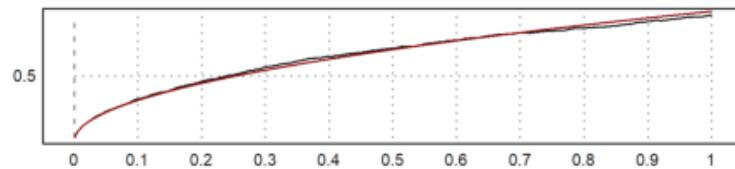
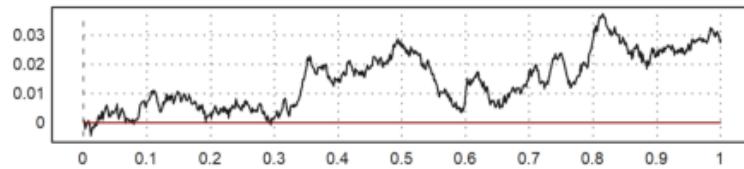
Median dari 10 bilangan acak terdistribusi 0-1-normal memiliki deviasi yang lebih besar.

```
>dev(median(M))
```

0.374460271535

Karena kita dapat dengan mudah membuat jalan acak, kita dapat mensimulasikan proses Wiener. Kami mengambil 1000 langkah dari 1000 proses. Kami kemudian memplot standar deviasi dan rata-rata langkah ke-n dari proses ini bersama dengan nilai yang diharapkan dalam warna merah.

```
>n=1000; m=1000; M=cumsum(normal(n,m)/sqrt(m)); ...
>t=(1:n)/n; figure(2,1); ...
>figure(1); plot2d(t,mean(M'))'; plot2d(t,0,color=red,>add); ...
>figure(2); plot2d(t,dev(M'))'; plot2d(t,sqrt(t),color=red,>add); ...
>figure(0):
```



Tes adalah alat penting dalam statistik. Di Euler, banyak tes yang diterapkan. Semua tes ini mengembalikan kesalahan yang kami terima jika kami menolak hipotesis nol.

Sebagai contoh, kami menguji lemparan dadu untuk distribusi seragam. Pada 600 lemparan, kami mendapat nilai berikut, yang kami masukkan ke uji chi-square.

```
>chitest([90,103,114,101,103,89],dup(100,6)')
```

0.498830517952

Tes chi-kuadrat juga memiliki mode yang menggunakan simulasi Monte Carlo untuk menguji statistik. Hasilnya harus hampir sama. Parameter >p menginterpretasikan vektor-y sebagai vektor probabilitas.

```
>chitest([90,103,114,101,103,89],dup(1/6,6)',>p,>montecarlo)
```

0.526

Kesalahan ini terlalu besar. Jadi kita tidak bisa menolak pemerataan distribusi. Ini tidak membuktikan bahwa dadu kami adil. Tapi kita tidak bisa menolak hipotesis kita.

Selanjutnya kami menghasilkan 1000 lemparan dadu menggunakan generator angka acak, dan melakukan pengujian yang sama.

```
>n=1000; t=random([1,n*6]); chitest(count(t*6,6),dup(n,6)')
```

0.528028118442

Mari kita uji nilai rata-rata 100 dengan uji-t.

```
>s=200+normal([1,100])*10; ...
>ttest(mean(s),dev(s),100,200)
```

0.0218365848476

Fungsi `ttest()` membutuhkan nilai rata-rata, simpangan, jumlah data, dan nilai rata-rata untuk diuji.

Sekarang mari kita periksa dua pengukuran untuk rata-rata yang sama. Kami menolak hipotesis bahwa mereka memiliki rata-rata yang sama, jika hasilnya  $<0,05$ .

```
>tcomparedata(normal(1,10),normal(1,10))
```

0.38722000942

Jika kami menambahkan bias ke satu distribusi, kami mendapat lebih banyak penolakan. Ulangi simulasi ini beberapa kali untuk melihat efeknya.

```
>tcomparedata(normal(1,10),normal(1,10)+2)
```

5.60009101758e-07

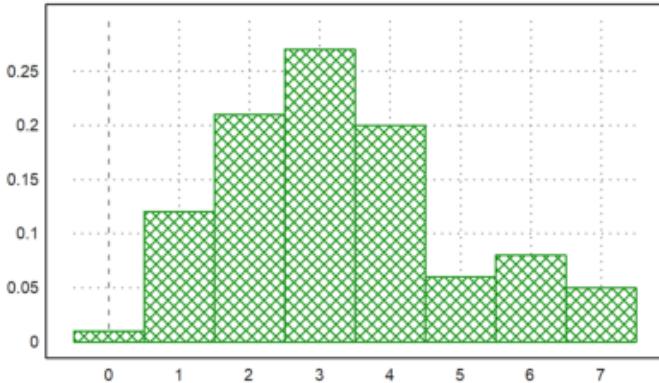
Dalam contoh berikutnya, kami menghasilkan 20 lemparan dadu acak 100 kali dan menghitungnya. Harus ada rata-rata  $20/6=3,3$ .

```
>R=random(100,20); R=sum(R*6<=1); mean(R)
```

3.28

Kami sekarang membandingkan jumlah satu dengan distribusi binomial. Pertama kita memplot distribusi satuan.

```
>plot2d(R,distribution=max(R)+1,even=1,style="\/"):
```



```
>t=count(R,21);
```

Kemudian kami menghitung nilai yang diharapkan.

```
>n=0:20; b=bin(20,n)*(1/6)^n*(5/6)^(20-n)*100;
```

Kita harus mengumpulkan beberapa angka untuk mendapatkan kategori yang cukup besar.

```
>t1=sum(t[1:2])|t[3:7]|sum(t[8:21]); ...
>b1=sum(b[1:2])|b[3:7]|sum(b[8:21]);
```

Uji chi-square menolak hipotesis bahwa distribusi kita adalah distribusi binomial, jika hasilnya <0,05.

```
>chitest(t1,b1)
```

0.53921579764

Contoh berikut berisi hasil dari dua kelompok orang (pria dan wanita, katakanlah) memilih satu dari enam partai.

```
>A=[23,37,43,52,64,74;27,39,41,49,63,76]; ...  
> writetable(A,wc=6,labr=["m","f"],labc=1:6)
```

	1	2	3	4	5	6
m	23	37	43	52	64	74
f	27	39	41	49	63	76

Kami ingin menguji independensi suara dari jenis kelamin. Tes tabel chi^2 melakukan ini. Hasilnya terlalu besar untuk menolak kemerdekaan. Jadi kami tidak bisa mengatakan, jika pemungutan suara tergantung pada jenis kelamin dari data tersebut.

```
>tabletest(A)
```

0.990701632326

Berikut adalah tabel yang diharapkan, jika kita mengasumsikan frekuensi pemungutan suara yang diamati.

```
>writetable(expectedtable(A),wc=6,dc=1,labr=["m","f"],labc=1:6)
```

	1	2	3	4	5	6
m	24.9	37.9	41.9	50.3	63.3	74.7
f	25.1	38.1	42.1	50.7	63.7	75.3

Kita dapat menghitung koefisien kontingensi yang dikoreksi. Karena sangat mendekati 0, kami menyimpulkan bahwa pemungutan suara tidak bergantung pada jenis kelamin.

```
>contingency(A)
```

0.0427225484717

## Beberapa Tes Lagi

---

Selanjutnya kami menggunakan analisis varians (F-test) untuk menguji tiga sampel data yang terdistribusi normal untuk nilai rata-rata yang sama. Metode tersebut dinamakan ANOVA (analysis of variance). Di Euler, fungsi varanalysis() digunakan.

```
>x1=[109,111,98,119,91,118,109,99,115,109,94]; mean(x1),
```

106.545454545

```
>x2=[120,124,115,139,114,110,113,120,117]; mean(x2),
```

119.111111111

```
>x3=[120,112,115,110,105,134,105,130,121,111]; mean(x3)
```

116.3

```
>varanalysis(x1,x2,x3)
```

0.0138048221371

Ini berarti, kami menolak hipotesis dengan nilai rata-rata yang sama. Kami melakukan ini dengan probabilitas kesalahan 1,3%.

Ada juga uji median yang menolak sampel data dengan distribusi rata-rata yang berbeda menguji median sampel bersatu.

```
>a=[56,66,68,49,61,53,45,58,54];  
>b=[72,81,51,73,69,78,59,67,65,71,68,71];  
>mediantest(a,b)
```

0.0241724220052

Tes lain tentang kesetaraan adalah tes peringkat. Ini jauh lebih tajam daripada tes median.

```
>ranktest(a,b)
```

0.00199969612469

Dalam contoh berikut, kedua distribusi memiliki rata-rata yang sama.

```
>ranktest(random(1,100),random(1,50)*3-1)
```

0.129608141484

Mari kita coba mensimulasikan dua perlakuan a dan b yang diterapkan pada orang yang berbeda.

```
>a=[8.0,7.4,5.9,9.4,8.6,8.2,7.6,8.1,6.2,8.9];  
>b=[6.8,7.1,6.8,8.3,7.9,7.2,7.4,6.8,6.8,8.1];
```

Tes signum memutuskan, jika a lebih baik dari b.

```
>signtest(a,b)
```

0.0546875

Ini terlalu banyak kesalahan. Kita tidak dapat menolak bahwa a sama baiknya dengan b.

Tes Wilcoxon lebih tajam dari tes ini, tetapi bergantung pada nilai kuantitatif perbedaannya.

```
>wilcoxon(a,b)
```

0.0296680599405

Mari kita coba dua tes lagi menggunakan rangkaian yang dihasilkan.

```
>wilcoxon(normal(1,20),normal(1,20)-1)
```

0.0068706451766

```
>wilcoxon(normal(1,20),normal(1,20))
```

0.275145971064

Berikut ini adalah tes untuk generator angka acak. Euler menggunakan generator yang sangat bagus, jadi kita tidak perlu berharap ada masalah.

Pertama kami menghasilkan sepuluh juta angka acak di [0,1].

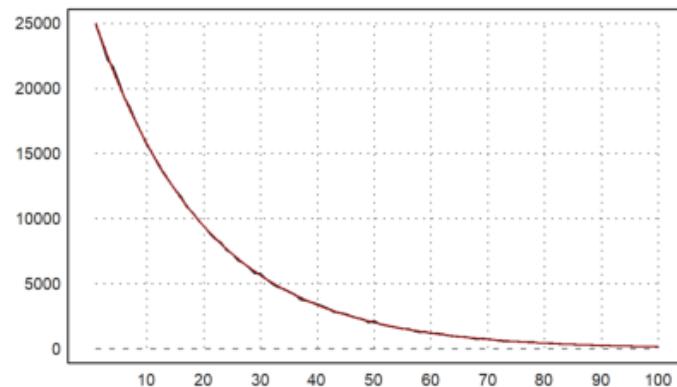
```
>n:=10000000; r:=random(1,n);
```

Selanjutnya kita menghitung jarak antara dua angka kurang dari 0,05.

```
>a:=0.05; d:=differences(nonzeros(r<a));
```

Akhirnya, kami memplot berapa kali, setiap jarak terjadi, dan membandingkannya dengan nilai yang diharapkan.

```
>m=getmultiplicities(1:100,d); plot2d(m); ...
> plot2d("n*(1-a)^(x-1)*a^2",color=red,>add):
```



Hapus datanya.

```
>remvalue n;
```

## **Pengantar untuk Pengguna Proyek R**

---

Jelas, EMT tidak bersaing dengan R sebagai paket statistik. Namun, ada banyak prosedur dan fungsi statistik yang tersedia di EMT juga. Jadi EMT dapat memenuhi kebutuhan dasar. Lagi pula, EMT hadir dengan paket numerik dan sistem aljabar komputer.

Notebook ini cocok untuk Anda jika sudah familiar dengan R, namun perlu mengetahui perbedaan sintaks EMT dan R. Kami mencoba memberikan gambaran umum tentang hal-hal yang jelas dan kurang jelas yang perlu Anda ketahui.

Selain itu, kami mencari cara untuk bertukar data antara kedua sistem.

Perhatikan bahwa ini adalah pekerjaan yang sedang berjalan.

**Sintaks Dasar**

---

Hal pertama yang Anda pelajari di R adalah membuat vektor. Di EMT, perbedaan utamanya adalah operator : dapat mengambil ukuran langkah. Apalagi daya ikatnya rendah.

```
>n=10; 0:n/20:n-1
```

```
[0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6, 6.5,  
7, 7.5, 8, 8.5, 9]
```

Fungsi c() tidak ada. Dimungkinkan untuk menggunakan vektor untuk menggabungkan berbagai hal.

Contoh berikut, seperti banyak lainnya, dari "Introduction to R" yang disertakan dengan proyek R. Jika Anda membaca PDF ini, Anda akan menemukan bahwa saya mengikuti jalannya dalam tutorial ini.

```
>x=[10.4, 5.6, 3.1, 6.4, 21.7]; [x,0,x]
```

```
[10.4, 5.6, 3.1, 6.4, 21.7, 0, 10.4, 5.6, 3.1, 6.4, 21.7]
```

Operator titik dua dengan ukuran langkah EMT diganti dengan fungsi seq() di R. Kita bisa menulis fungsi ini di EMT.

```
>function seq(a,b,c) := a:b:c; ...
>seq(0,-0.1,-1)
```

```
[0, -0.1, -0.2, -0.3, -0.4, -0.5, -0.6, -0.7, -0.8, -0.9, -1]
```

Fungsi rep() dari R tidak ada di EMT. Untuk input vektor, dapat ditulis sebagai berikut.

```
>function rep(x:vector,n:index) := flatten(dup(x,n)); ...
>rep(x,2)
```

```
[10.4, 5.6, 3.1, 6.4, 21.7, 10.4, 5.6, 3.1, 6.4, 21.7]
```

Perhatikan bahwa "==" atau ":=" digunakan untuk tugas. Operator "->" digunakan untuk unit di EMT.

```
>125km -> " miles"
```

```
77.6713990297 miles
```

The "`<-`" operator for assignment is misleading anyway, and not a good idea of R. The following will compare a and -4 in EMT.

```
>a=2; a<-4
```

0

Di R, "`a<-4<3`" berfungsi, tetapi "`a<-4<-3`" tidak. Saya juga memiliki ambiguitas serupa di EMT, tetapi mencoba menghilangkannya sedikit demi sedikit.

EMT dan R memiliki vektor tipe boolean. Namun dalam EMT, angka 0 dan 1 digunakan untuk mewakili salah dan benar. Di R, nilai benar dan salah tetap bisa digunakan dalam aritmatika biasa seperti di EMT.

```
>x<5, %*x
```

```
[0, 0, 1, 0, 0]
[0, 0, 3.1, 0, 0]
```

EMT melempar kesalahan atau menghasilkan NAN tergantung pada bendera "kesalahan".

```
>errors off; 0/0, isNaN(sqrt(-1)), errors on;
```

NAN

1

String sama di R dan EMT. Keduanya berada di lokal saat ini, bukan di Unicode.

Di R ada paket untuk Unicode. Di EMT, sebuah string dapat berupa string Unicode. String unicode dapat diterjemahkan ke pengkodean lokal dan sebaliknya. Selain itu, u”...” dapat berisi entitas HTML.

```
>u"    Ren  Grothmann"
```

© Ren  Grothmann

Berikut ini mungkin atau mungkin tidak ditampilkan dengan benar di sistem Anda sebagai A dengan titik dan garis di atasnya. Itu tergantung pada font yang Anda gunakan.

```
>chartoutf([480])
```

Penggabungan string dilakukan dengan ”+” atau ”|”. Itu bisa termasuk angka, yang akan dicetak dalam format saat ini.

```
>"pi = "+pi
```

pi = 3.14159265359

Sebagian besar waktu, ini akan berfungsi seperti di R.

Tetapi EMT akan menginterpretasikan indeks negatif dari belakang vektor, sedangkan R menginterpretasikan  $x[n]$  sebagai  $x$  tanpa elemen ke-n.

```
>x, x[1:3], x[-2]
```

```
[10.4,  5.6,  3.1,  6.4, 21.7]
[10.4,  5.6,  3.1]
6.4
```

Perilaku R dapat dicapai dalam EMT dengan drop().

```
>drop(x,2)
```

```
[10.4,  3.1,  6.4, 21.7]
```

Vektor logis tidak diperlakukan berbeda sebagai indeks di EMT, berbeda dengan R. Anda perlu mengekstraksi elemen bukan nol terlebih dahulu di EMT.

```
>x, x>5, x[nonzeros(x>5)]
```

```
[10.4, 5.6, 3.1, 6.4, 21.7]
[1, 1, 0, 1, 1]
[10.4, 5.6, 6.4, 21.7]
```

Sama seperti di R, vektor indeks dapat berisi pengulangan.

```
>x[[1,2,2,1]]
```

```
[10.4, 5.6, 5.6, 10.4]
```

Tetapi nama untuk indeks tidak dimungkinkan di EMT. Untuk paket statistik, hal ini sering diperlukan untuk memudahkan akses ke elemen vektor.

Untuk meniru perilaku ini, kita dapat mendefinisikan fungsi sebagai berikut.

```
>function sel (v,i,s) := v[indexof(s,i)]; ...
>s=["first","second","third","fourth"]; sel(x,[ "first","third"],s)
```

```
Trying to overwrite protected function sel!
Error in:
function sel (v,i,s) := v[indexof(s,i)]; ... ...
^
```

```
Trying to overwrite protected function sel!
Error in:
function sel (v,i,s) := v[indexof(s,i)]; ... ...
^
```

```
Trying to overwrite protected function sel!
Error in:
function sel (v,i,s) := v[indexof(s,i)]; ... ...
^
```

[10.4, 3.1]

## Tipe Data

---

EMT memiliki lebih banyak tipe data tetap daripada R. Jelas, di R terdapat vektor yang tumbuh. Anda dapat menyetel vektor numerik kosong  $v$  dan menetapkan nilai ke elemen  $v[17]$ . Ini tidak mungkin di EMT.

Berikut ini agak tidak efisien.

```
>v=[]; for i=1 to 10000; v=v|i; end;
```

EMT sekarang akan membuat vektor dengan  $v$  dan  $i$  ditambahkan pada tumpukan dan menyalin vektor itu kembali ke variabel global  $v$ .

Semakin efisien pra-mendefinisikan vektor.

```
>v=zeros(10000); for i=1 to 10000; v[i]=i; end;
```

Untuk mengubah jenis tanggal di EMT, Anda dapat menggunakan fungsi seperti `complex()`.

```
>complex(1:4)
```

```
[ 1+0i ,  2+0i ,  3+0i ,  4+0i ]
```

Konversi ke string hanya dimungkinkan untuk tipe data dasar. Format saat ini digunakan untuk penggabungan string sederhana. Tapi ada fungsi seperti print() atau frac().

Untuk vektor, Anda dapat dengan mudah menulis fungsi Anda sendiri.

```
>function tostr (v) ...  
  
    s="[";  
    loop 1 to length(v);  
        s=s+print(v[#],2,0);  
        if #<length(v) then s=s+","; endif;  
    end;  
    return s+"]";  
endfunction
```

```
>tostr(linspace(0,1,10))
```

```
[0.00,0.10,0.20,0.30,0.40,0.50,0.60,0.70,0.80,0.90,1.00]
```

Untuk komunikasi dengan Maxima, terdapat fungsi convertmxm(), yang juga dapat digunakan untuk memformat vektor untuk output.

```
>convertm xm(1:10)
```

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

Untuk Lateks, perintah tex dapat digunakan untuk mendapatkan perintah Lateks.

```
>tex(&[1,2,3])
```

```
\left[ 1 , 2 , 3 \right]
```

## Faktor dan Tabel

---

Dalam pengantar R ada contoh dengan apa yang disebut faktor.

Berikut ini adalah daftar wilayah dari 30 negara bagian.

```
>austates = ["tas", "sa", "qld", "nsw", "nsw", "nt", "wa", "wa", ...
>"qld", "vic", "nsw", "vic", "qld", "qld", "sa", "tas", ...
>"sa", "nt", "wa", "vic", "qld", "nsw", "nsw", "wa", ...
>"sa", "act", "nsw", "vic", "vic", "act"];
```

Asumsikan, kita memiliki pendapatan yang sesuai di setiap negara bagian.

```
>incomes = [60, 49, 40, 61, 64, 60, 59, 54, 62, 69, 70, 42, 56, ...
>61, 61, 61, 58, 51, 48, 65, 49, 49, 41, 48, 52, 46, ...
>59, 46, 58, 43];
```

Sekarang, kami ingin menghitung rata-rata pendapatan di wilayah tersebut. Menjadi program statistik, R memiliki factor() dan tappy() untuk ini.

EMT dapat melakukannya dengan menemukan indeks wilayah di daftar unik wilayah.

```
>auterr=sort(unique(austates)); f=indexofsorted(auterr,austates)
```

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

Pada saat itu, kita dapat menulis fungsi loop kita sendiri untuk melakukan sesuatu hanya untuk satu faktor.

Atau kita bisa meniru fungsi tapply() dengan cara berikut.

```
>function map_tappl (i; f$:call, cat, x) ...
```

```
u=sort(unique(cat));  
f=indexof(u,cat);  
return f$(x[nonzeros(f==indexof(u,i))]);  
endfunction
```

Ini sedikit tidak efisien, karena menghitung wilayah unik untuk setiap i, tetapi berhasil.

```
>tappl(auterr,"mean",austates,incomes)
```

```
[44.5, 57.3333333333, 55.5, 53.6, 55, 60.5, 56, 52.25]
```

Perhatikan bahwa ini berfungsi untuk setiap vektor wilayah.

```
>tappl(["act","nsw"],"mean",austates,incomes)
```

```
[44.5, 57.3333333333]
```

Sekarang, paket statistik EMT mendefinisikan tabel seperti pada R. Fungsi readtable() dan writetable() dapat digunakan untuk input dan output.

Sehingga kita bisa mencetak rata-rata pendapatan negara di daerah dengan cara yang bersahabat.

```
>writetable(tappl(auterr,"mean",austates,incomes),labc=auterr,wc=7)
```

act	nsw	nt	qld	sa	tas	vic	wa
44.5	57.33	55.5	53.6	55	60.5	56	52.25

Kami juga dapat mencoba meniru perilaku R sepenuhnya.

Faktor jelas harus disimpan dalam kumpulan dengan jenis dan kategori (negara bagian dan teritori dalam contoh kita). Untuk EMT, kami menambahkan indeks yang telah dihitung sebelumnya.

```
>function makef (t) ...  
  
## Factor data  
## Returns a collection with data t, unique data, indices.  
## See: tapply  
u=sort(unique(t));  
return {{t,u,indexofsorted(u,t)}};  
endfunction
```

```
>statef=makef(austates);
```

Sekarang elemen ketiga dari koleksi akan berisi indeks.

```
>statef[3]
```

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

Sekarang kita bisa meniru tapply() dengan cara berikut. Ini akan mengembalikan tabel sebagai kumpulan data tabel dan judul kolom.

```
>function tapply (t:vector,tf,f$:call) ...  
  
## Makes a table of data and factors  
## tf : output of makef()  
## See: makef  
uf=tf[2]; f=tf[3]; x=zeros(length(uf));  
for i=1 to length(uf);  
    ind=nonzeros(f==i);  
    if length(ind)==0 then x[i]=NAN;  
    else x[i]=f$(t[ind]);  
    endif;  
end;  
return {{x,uf}};  
endfunction
```

Kami tidak menambahkan banyak pengecekan tipe di sini. Satu-satunya tindakan pencegahan menyangkut kategori (faktor) tanpa data. Tetapi orang harus memeriksa panjang t yang benar dan kebenaran koleksi tf.

Tabel ini dapat dicetak sebagai tabel dengan writetable().

```
>writetable(tapply(incomes,statef,"mean"),wc=7)
```

act	nsw	nt	qld	sa	tas	vic	wa
44.5	57.33	55.5	53.6	55	60.5	56	52.25

## Array

---

EMT hanya memiliki dua dimensi untuk array. Tipe datanya disebut matriks. Namun, akan mudah untuk menulis fungsi untuk dimensi yang lebih tinggi atau pustaka C untuk ini.

R memiliki lebih dari dua dimensi. Di R array adalah vektor dengan bidang dimensi.

Dalam EMT, vektor adalah matriks dengan satu baris. Itu dapat dibuat menjadi matriks dengan redim().

```
>shortformat; X=redim(1:20,4,5)
```

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20

Ekstraksi baris dan kolom, atau sub-matriks, sangat mirip dengan R.

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

2	3
7	8
12	13
17	18

Namun, dalam R dimungkinkan untuk menetapkan daftar indeks spesifik vektor ke suatu nilai. Hal yang sama dimungkinkan di EMT hanya dengan satu putaran.

```
>function setmatrixvalue (M, i, j, v) ...
loop 1 to max(length(i),length(j),length(v))
    M[i{#},j{#}] = v{#};
end;
endfunction
```

Kami mendemonstrasikan ini untuk menunjukkan bahwa matriks dilewatkan dengan referensi di EMT. Jika Anda tidak ingin mengubah matriks asli M, Anda perlu menyalinnya ke dalam fungsi.

```
>setmatrixvalue(X,1:3,3:-1:1,0); X,
```

1	2	0	4	5
6	0	8	9	10
0	12	13	14	15
16	17	18	19	20

Produk luar di EMT hanya dapat dilakukan di antara vektor. Ini otomatis karena bahasa matriks. Satu vektor harus berupa vektor kolom dan yang lainnya vektor baris.

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

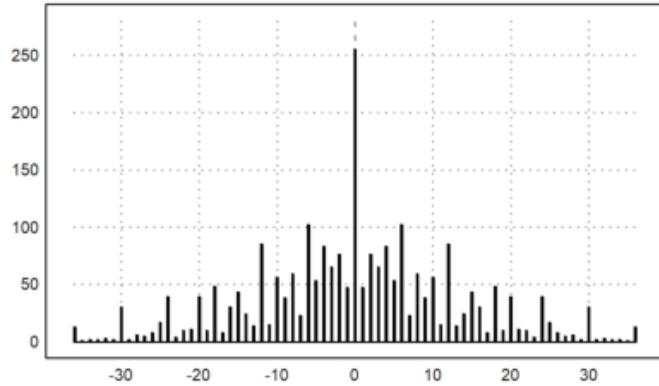
1	2	3	4	5
2	4	6	8	10
3	6	9	12	15
4	8	12	16	20
5	10	15	20	25

Dalam pengantar PDF untuk R ada contoh, yang menghitung distribusi ab-cd untuk a,b,c,d dipilih dari 0 sampai n secara acak. Solusi dalam R adalah membentuk matriks 4 dimensi dan menjalankan table() di atasnya.

Tentu saja, ini bisa dicapai dengan satu putaran. Tapi loop tidak efektif di EMT atau R. Di EMT, kita bisa menulis loop di C dan itu akan menjadi solusi tercepat.

Tapi kami ingin meniru perilaku R. Untuk ini, kami perlu meratakan perkalian ab dan membuat matriks ab-cd.

```
>a=0:6; b=a'; p=flatten(a*b); q=flatten(p-p'); ...
>u=sort(unique(q)); f=getmultiplicities(u,q); ...
>statplot(u,f,"h"):
```



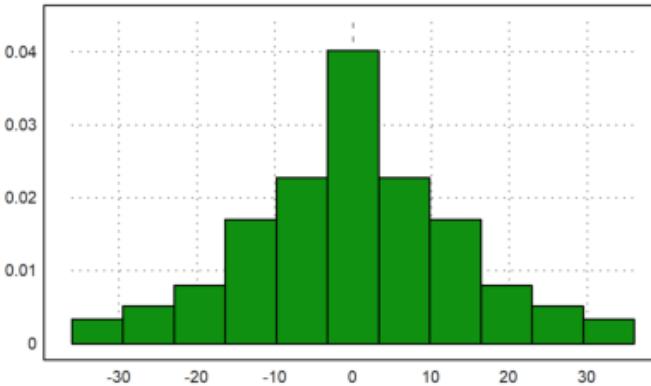
Selain perkalian yang tepat, EMT dapat menghitung frekuensi dalam vektor.

```
>getfrequencies(q,-50:10:50)
```

```
[0, 23, 132, 316, 602, 801, 333, 141, 53, 0]
```

Cara paling mudah untuk memplot ini sebagai distribusi adalah sebagai berikut.

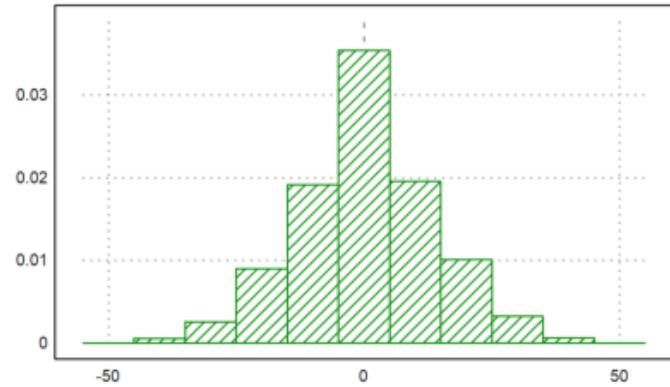
```
>plot2d(q,distribution=11):
```



Tetapi juga memungkinkan untuk melakukan pra-perhitungan hitungan dalam interval yang dipilih sebelumnya. Tentu saja, berikut ini menggunakan getfrequencies() secara internal.

Karena fungsi histo() mengembalikan frekuensi, kita perlu menskalakannya sehingga integral di bawah grafik batang adalah 1.

```
>{x,y}=histo(q,v=-55:10:55); y=y/sum(y)/differences(x); ...
>plot2d(x,y,>bar,style="/"):
```



## Daftar

---

EMT memiliki dua jenis daftar. Salah satunya adalah daftar global yang bisa berubah, dan yang lainnya adalah tipe daftar yang tidak bisa diubah. Kami tidak peduli dengan daftar global di sini.

Jenis daftar yang tidak dapat diubah disebut koleksi di EMT. Ini berperilaku seperti struktur di C, tetapi elemennya hanya diberi nomor dan tidak diberi nama.

```
>L={"Fred","Flintstone",40,[1990,1992]}
```

```
Fred  
Flintstone  
40  
[1990, 1992]
```

Saat ini elemen tidak memiliki nama, meskipun nama dapat diatur untuk tujuan khusus. Mereka diakses oleh nomor.

```
>(L[4])[2]
```

```
1992
```

## File Input dan Output (Membaca dan Menulis Data)

---

Anda sering ingin mengimpor matriks data dari sumber lain ke EMT. Tutorial ini memberitahu Anda tentang banyak cara untuk mencapai hal ini. Fungsi sederhana adalah writematrix() dan readmatrix(). Mari kita tunjukkan cara membaca dan menulis vektor real ke file.

```
>a=random(1,100); mean(a), dev(a),
```

```
0.49815  
0.28037
```

Untuk menulis data ke file, kami menggunakan fungsi writematrix().

Karena pengantar ini kemungkinan besar ada di direktori, di mana pengguna tidak memiliki akses tulis, kami menulis data ke direktori home pengguna. Untuk buku catatan sendiri, hal ini tidak diperlukan, karena file data akan ditulis ke dalam direktori yang sama.

```
>filename="test.dat";
```

Sekarang kita menulis vektor kolom a' ke file. Ini menghasilkan satu nomor di setiap baris file.

```
>writematrix(a',filename);
```

Untuk membaca data, kami menggunakan readmatrix().

```
>a=readmatrix(filename);
```

Dan hapus file tersebut.

```
>fileremove(filename);
>mean(a), dev(a),
```

```
0.49815
0.28037
```

Fungsi writematrix() atau writetable() dapat dikonfigurasi untuk bahasa lain.

Misalnya, jika Anda memiliki sistem bahasa Indonesia (titik desimal dengan koma), Excel Anda memerlukan nilai dengan koma desimal yang dipisahkan oleh titik koma dalam file csv (defaultnya adalah nilai yang dipisahkan koma). File berikut "test.csv" akan muncul di folder cuurent Anda.

```
>filename="test.csv"; ...
>writematrix(random(5,3),file=filename,separator=",");
```

Anda sekarang dapat membuka file ini dengan Excel bahasa Indonesia secara langsung.

```
>fileremove(filename);
```

Terkadang kami memiliki string dengan token seperti berikut ini.

```
>s1:="f m m f m m m f f f m m f"; ...
>s2:="f f f m m f f";
```

Untuk menandai ini, kami mendefinisikan vektor token.

```
>tok:=[ "f" , "m" ]
```

f

m

Kemudian kita dapat menghitung berapa kali setiap token muncul dalam string, dan memasukkan hasilnya ke dalam tabel.

```
>M:=getmultiplicities(tok,strtokens(s1))_ ...
>  getmultiplicities(tok,strtokens(s2));
```

Tulis tabel dengan header token.

```
>writetable(M,labc=tok,labr=1:2,wc=8)
```

	f	m
1	6	7
2	5	2

Untuk statika, EMT dapat membaca dan menulis tabel.

```
>file="test.dat"; open(file,"w"); ...
>writeln("A,B,C"); writematrix(random(3,3)); ...
>close();
```

The file looks like this.

```
>printfile(file)
```

```
A,B,C
0.7003664386138074,0.1875530821001213,0.3262339279660414
0.5926249243193858,0.1522927283984059,0.368140583062521
0.8065535209872989,0.7265910840408142,0.7332619844597152
```

Fungsi readtable() dalam bentuknya yang paling sederhana dapat membaca ini dan mengembalikan kumpulan nilai dan baris heading.

```
>L=readtable(file,>list);
```

Koleksi ini dapat dicetak dengan writetable() ke notebook, atau ke file.

```
>writetable(L,wc=10,dc=5)
```

A	B	C
0.70037	0.18755	0.32623
0.59262	0.15229	0.36814
0.80655	0.72659	0.73326

Matriks nilai adalah elemen pertama dari L. Perhatikan bahwa mean() dalam EMT menghitung nilai rata-rata dari baris matriks.

```
>mean(L[1])
```

0.40472
0.37102
0.75547

## File CSV

---

Pertama, mari kita menulis matriks ke dalam file. Untuk hasilnya, kami membuat file di direktori kerja saat ini.

```
>file="test.csv";  ...
>M=random(3,3); writematrix(M,file);
```

Berikut adalah isi dari file ini.

```
>printfile(file)
```

```
0.8221197733097619,0.821531098722547,0.7771240608094004
0.8482947121863489,0.3237767724883862,0.6501422353377985
0.1482301827518109,0.3297459716109594,0.6261901074210923
```

CSV ini dapat dibuka pada sistem bahasa Inggris ke dalam Excel dengan klik dua kali. Jika Anda mendapatkan file seperti itu di sistem Jerman, Anda perlu mengimpor data ke Excel dengan memperhatikan titik desimal.

Tetapi titik desimal juga merupakan format default untuk EMT. Anda dapat membaca matriks dari file dengan readmatrix().

```
>readmatrix(file)
```

```
0.82212  0.82153  0.77712  
0.84829  0.32378  0.65014  
0.14823  0.32975  0.62619
```

Dimungkinkan untuk menulis beberapa matriks ke satu file. Perintah open() dapat membuka file untuk ditulis dengan parameter "w". Standarnya adalah "r" untuk membaca.

```
>open(file,"w"); writematrix(M); writematrix(M'); close();
```

Matriks dipisahkan oleh garis kosong. Untuk membaca matriks, buka file dan panggil readmatrix() beberapa kali.

```
>open(file); A=readmatrix(); B=readmatrix(); A==B, close();
```

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

Di Excel atau spreadsheet serupa, Anda dapat mengekspor matriks sebagai CSV (nilai yang dipisahkan koma). Di Excel 2007, gunakan "simpan sebagai" dan "format lain", lalu pilih "CSV". Pastikan, tabel saat ini hanya berisi data yang ingin Anda ekspor.

Ini sebuah contoh.

```
>printfile("excel-data.csv")
```

```
Could not open the file  
excel-data.csv  
for reading!  
Try "trace errors" to inspect local variables after errors.  
printfile:  
    open(filename,"r");
```

Seperti yang Anda lihat, sistem Jerman saya menggunakan titik koma sebagai pemisah dan koma desimal. Anda dapat mengubahnya di pengaturan sistem atau di Excel, tetapi tidak perlu membaca matriks ke dalam EMT.

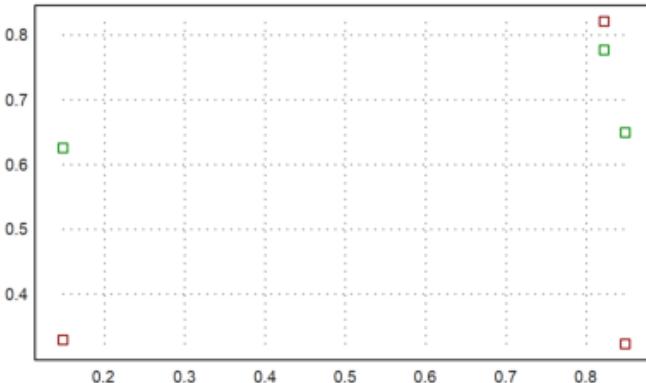
Cara termudah untuk membaca ini ke Euler adalah readmatrix(). Semua koma diganti dengan titik dengan parameter >koma. Untuk CSV bahasa Inggris, hilangkan saja parameter ini.

```
>M=readmatrix("excel-data.csv",>comma)
```

```
Could not open the file  
excel-data.csv  
for reading!  
Try "trace errors" to inspect local variables after errors.  
readmatrix:  
    if filename<>"" then open(filename,"r"); endif;
```

Let us plot this.

```
>plot2d(M'[1],M'[2:3],>points,color=[red,green]):
```



Ada cara yang lebih mendasar untuk membaca data dari file. Anda dapat membuka file dan membaca angka baris demi baris. Fungsi getvectorline() akan membaca angka dari baris data. Secara default, ini mengharapkan titik desimal. Tapi itu juga bisa menggunakan koma desimal, jika Anda memanggil setdecimaldot(“,”) sebelum Anda menggunakan fungsi ini.

Fungsi berikut adalah contoh untuk ini. Itu akan berhenti di akhir file atau baris kosong.

```
>function myload (file) ...
```

```
open(file);
M=[];
repeat
    until eof();
    v=getvectorline(3);
    if length(v)>0 then M=M_v; else break; endif;
end;
return M;
close(file);
endfunction
```

```
>myload(file)
```

0.82212	0	0.82153	0	0.77712
0.84829	0	0.32378	0	0.65014
0.14823	0	0.32975	0	0.62619

Dimungkinkan juga untuk membaca semua angka dalam file itu dengan getvector().

```
>open(file); v=getvector(10000); close(); redim(v[1:9],3,3)
```

0.82212	0	0.82153
0	0.77712	0.84829
0	0.32378	0

Thus it is very easy to save a vector of values, one value in each line and read back this vector.

```
>v=random(1000); mean(v)
```

0.50303

```
>writematrix(v',file); mean(readmatrix(file)')
```

0.50303

## Menggunakan Tabel

---

Tabel dapat digunakan untuk membaca atau menulis data numerik. Sebagai contoh, kami menulis tabel dengan tajuk baris dan kolom ke file.

```
>file="test.tab"; M=random(3,3); ...
>open(file,"w"); ...
>writetable(M,separator=",",labc=["one","two","three"]); ...
>close(); ...
>printfile(file)
```

one	two	three
0.09,	0.39,	0.86
0.39,	0.86,	0.71
0.2,	0.02,	0.83

Ini dapat diimpor ke Excel.

Untuk membaca file di EMT, kami menggunakan readtable().

```
>{M,headings}=readtable(file,>clabs); ...
>writetable(M,labc=headings)
```

one	two	three
0.09	0.39	0.86
0.39	0.86	0.71
0.2	0.02	0.83

## Menganalisis Garis

---

Anda bahkan dapat mengevaluasi setiap baris dengan tangan. Misalkan, kita memiliki garis dengan format berikut.

```
>line="2020-11-03,Tue,1'114.05"
```

2020-11-03, Tue, 1'114.05

Pertama kita dapat menandai garis.

```
>vt=strtoks(line)
```

2020-11-03  
Tue  
1'114.05

Kemudian kita dapat mengevaluasi setiap elemen garis menggunakan evaluasi yang sesuai.

```
>day(vt[1]), ...  
>indexof(["mon","tue","wed","thu","fri","sat","sun"],tolower(vt[2])), ...  
>strrepl(vt[3],'"','"')()
```

```
7.3816e+05  
2  
1114
```

Menggunakan ekspresi reguler, dimungkinkan untuk mengekstraksi hampir semua informasi dari sebaris data.

Asumsikan kita memiliki baris berikut sebuah dokumen HTML.

```
>line=<tr><td>1145.45</td><td>5.6</td><td>-4.5</td><tr>"
```

```
<tr><td>1145.45</td><td>5.6</td><td>-4.5</td><tr>
```

Untuk mengekstrak ini, kami menggunakan ekspresi reguler, yang mencari

- tanda kurung tutup >,
- string apa pun yang tidak mengandung tanda kurung dengan

sub-pertandingan "(...)",

- braket pembuka dan penutup menggunakan solusi terpendek,
- sekali lagi string apa pun yang tidak mengandung tanda kurung,
- dan tanda kurung buka <.

Ekspresi reguler agak sulit dipelajari tetapi sangat kuat.

```
>{pos,s,vt}=strxfind(line,>([`<>]+)<.+?>([`<>]+)<"");
```

Hasilnya adalah posisi kecocokan, string yang cocok, dan vektor string untuk sub-kecocokan.

```
>for k=1:length(vt); vt[k](), end;
```

```
1145.5  
5.6
```

Ini adalah fungsi yang membaca semua item numerik antara <td> dan </td>.

```
>function readtd (line) ...
v=[]; cp=0;
repeat
    {pos,s,vt}=strxfind(line,"<td.*?>(.+?)</td>",cp);
    until pos==0;
    if length(vt)>0 then v=v|vt[1]; endif;
    cp=pos+strlen(s);
end;
return v;
endfunction
```

```
>readtd(line+"<td>non-numerical</td>")
```

```
1145.45
5.6
-4.5
non-numerical
```

Situs web atau file dengan URL dapat dibuka di EMT dan dapat dibaca baris demi baris.

Dalam contoh, kami membaca versi terkini dari situs EMT. Kami menggunakan ekspresi reguler untuk memindai "Versi ..." dalam judul.

```
>function readversion () ...
```

```
urlopen("http://www.euler-math-toolbox.de/Programs/Changes.html");
repeat
    until urleof();
    s=urlgetline();
    k=strfind(s,"Version ",1);
    if k>0 then substring(s,k,strfind(s,<,k)-1), break; endif;
end;
urlclose();
endfunction
```

```
>readversion
```

Version 2024-01-12

## Input dan Output Variabel

---

Anda dapat menulis variabel dalam bentuk definisi Euler ke file atau ke baris perintah.

```
>writevar(pi,"mypi");
```

```
mypi = 3.141592653589793;
```

Untuk pengujian, kami membuat file Euler di direktori kerja EMT.

```
>file="test.e"; ...
>writevar(random(2,2),"M",file); ...
>printfile(file,3)
```

```
M = [ ..
0.5991820585590205, 0.7960280262224293;
0.5167243983231363, 0.2996684599070898];
```

We can now load the file. It will define the matrix M.

```
>load(file); show M,
```

```
M =  
0.59918  0.79603  
0.51672  0.29967
```

By the way, jika writevar() digunakan pada variabel, itu akan mencetak definisi variabel dengan nama variabel ini.

```
>writevar(M); writevar(inch$)
```

```
M = [ ..  
0.5991820585590205, 0.7960280262224293;  
0.5167243983231363, 0.2996684599070898];  
inch$ = 0.0254;
```

Kami juga dapat membuka file baru atau menambahkan file yang sudah ada. Dalam contoh kami menambahkan file yang dihasilkan sebelumnya.

```
>open(file,"a"); ...
>writevar(random(2,2),"M1"); ...
>writevar(random(3,1),"M2"); ...
>close();
>load(file); show M1; show M2;
```

```
M1 =
 0.30287  0.15372
 0.7504   0.75401
M2 =
 0.27213
 0.053211
 0.70249
```

Untuk menghapus file apa pun gunakan fileremove().

```
>fileremove(file);
```

Vektor baris dalam file tidak memerlukan koma, jika setiap angka berada di baris baru. Mari kita buat file seperti itu, menulis setiap baris satu per satu dengan writeln().

```
>open(file,"w"); writeln("M = ["); ...
>for i=1 to 5; writeln(""+random()); end; ...
>writeln("]"); close(); ...
>printfile(file)
```

```
M = [
0.344851384551
0.0807510017715
0.876519562911
0.754157709472
0.688392638934
];
```

```
>load(file); M
```

```
[0.34485, 0.080751, 0.87652, 0.75416, 0.68839]
```