TUGAS PEKAN 16 APLIKASI KOMPUTER

Fransisca Renita Pejoresa

22305144012

Matematika E 2022

PRODI MATEMATIKA
DEPARTEMEN PENDIDIKAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS NEGERI YOGYAKARTA

2023

DAFTAR ISI

BAB 1

KB PEKAN 3: MENGGUNAKAN EMT UNTUK MENYELESAIKAN MASALAH-MASALAH ALJABAR

[a4paper,10pt]article eumat

EMT untuk Perhitungan Aljabar

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

Contoh pertama

Menyederhanakan bentuk aljabar:

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

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

Menjabarkan:

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

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

The Command Line

A command line of Euler consists of one or several Euler commands followed by a semicolon ";" or a comma ",". The semicolon prevents the printing of the result. The comma after the last command can be omitted.
The following command line will only print the result of the expression, not the assignments or the format commands.

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

```
16.7551608191
```

Commands must be separated with a blank. The following command line prints its two results.

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

```
50.2654824574
100.530964915
```

Command lines are executed in the order the user presses return. So you get a new value each time you execute the second line.

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

```
0.540302305868
```

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

```
0.857553215846
```

If two lines are connected with "..." both lines will always execute simultaneously.

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

```
1.41666666667
1.41421568627
1.41421356237
```

This is also a good way to spread a long command over two or more lines. You can press Ctrl+Return to split a line in two at the current cursor position, or Ctlr+Back to join the lines.
To fold all multi-lines press Ctrl+L. Then the subsequent lines will only be visible, if one of them has the focus. To fold a single multi-line start the first line with "%+ ".

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

A line starting with %% will be completely invisible.

```
81
```

Euler supports loops in command lines, as long as they fit into one single line or a multi-line. In programs, this restrictions does not hold, of course. For more information consult the following introduction.

```
>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
```

It is okay to use a multi-line. Make sure the line ends with " ...".

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

```
1.41421356237
```

Conditional structures do also work.

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

```
Thought so!
```

When you execute a command, the cursor can be at any position in the command line. You can go back to a previous command or skip to the next command with the arrow keys. Or you can click into the comment section above the command to go to the command.
When you move the cursor along the line the opening and closing pairs of brackets or parentheses will highlight. Also, watch the status line. After the opening bracket of the sqrt() function, the status line will display a help text for the function. Execute the command with the return key.

```
>sqrt(sin(10°)/cos(20°))
```

```
0.429875017772
```

To see help for the most recent command, open the help window with F1. There, you can enter text to search for. On an empty line, the help for the help window will be displayed. You can press escape to clear the line, or to close the help window.
You can double click on any command to open the help for this command. Try double clicking the exp command below in the command line.

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

```
2.5
```

You can copy and paste in Euler too. Use Ctrl-C and Ctrl-V for this. To mark a text, drag the mouse or use shift together with any cursor key. Moreover, you can copy the highlighted brackets.

Basic Syntax

Euler knows the usual mathematical functions. As you have seen above, trigonometric functions work in radian or degree. To convert to degrees, append the degree symbol (with the F7 key) to the value, or use the function rad(x). The square root function is called sqrt in Euler. Of course, x^(1/2) is also possible.
To set variables, use either "=" or ":=". For the sake of clarity, this introduction uses the latter form. Spaces do not matter. But a space between commands is expected.
Multiple commands in one line are separated with "," or ";". The semicolon suppresses the output of the command. At the end of the command line a "," is assumed, if ";" is missing.

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

```
30.65625
```

EMT uses a programming syntax for expressions. To enter

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

you have to set the correct brackets and use / for fractions. Watch the highlighted brackets for assistance. Note that the Euler constant e is named E in EMT.

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

```
8.77908249441
```

To compute a complicate expression like

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

you need to enter it in line form.

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

```
23.2671801626
```

Carefully put brackets around sub-expressions that need to be computed first. EMT assists you by highlighting the expression that the closing bracket finishes. You will also have to enter the name "pi" for the Greek letter pi.
The result of this computation is a floating point number. It is by default printed with about 12 digits accuracy. In the following command line, we also learn how we can refer to the previous result within the same line.

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

```
0.47619047619
10/21
```

An Euler command can be an expression or a primitive command. An expression is made of operators and functions. If necessary, it must contain brackets to force the correct order of execution. In doubt, setting a bracket is a good idea. Note that EMT shows opening and closing brackets while editing the command line.

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

```
14.4978445072
```

The numerical operators of Euler include

```
+ unary or operator plus
- unary or operator minus
*, /
. the matrix product
a^b power for positive a or integer b (a**b works too)
n! the factorial operator
```

and many more.
Here are some of the functions you might need. There are many more.

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

Some commands have aliases, e.g. ln for log.

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

```
2
0.5
```

```
>sin(30°)
```

```
0.5
```

Make sure to use parentheses (round brackets), whenever there is doubt about the order of execution! The following is not the same as (2^3)^4, which is the default for 2^3^4 in EMT (some numerical systems do it the other way).

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

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

Real Numbers

The primary data type in Euler is the real number. Reals are represented in IEEE format with about 16 decimal digits of accuracy.

```
>longest 1/3
```

```
      0.3333333333333333
```

The internal dual representation takes 8 bytes.

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

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

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

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

Strings

A string in Euler is defined with "...".

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

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

Strings can be concatenated with | or with +. This also works with numbers, which are converted to strings in that case.

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

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

The print function does also convert a number to a string. It can take a number of digits and a number of places (0 for dense output), and optimally a unit.

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

```
Golden Ratio : 1.61803
```

There is a special string none, which does not print. It is returned by some functions, when the result does not matter. (It is returned automatically, if the function does not have a return statement.)

```
>none
```

To convert a string to a number simply evaluate it. This works for expressions too (see below).

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

```
1234.5
```

To define a string vector, use the vector [...] notation.

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

```
affe
charlie
bravo
```

The empty string vector is denoted by [none]. String vectors can be concatenated.

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

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

Strings can contain Unicode characters. Internally, these strings contain UTF-8 code. To generate such a string, use u"..." and one of the HTML entities.
Unicode strings can be concatenated like other strings.

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

```
 = 45°
```

I

In comments, the same entities like , etc. can be used. This may be a quick alternative to Latex. (More details on comments below).

There are some functions to create or analyze unicode strings. The function strtochar() will recognize Unicode strings, and translate them correctly.

```
>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]
```

The result is a vector of Unicode numbers. The converse function is chartoutf().

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

```
Ü is a German letter
```

The function utf() can translate a string with entities in a variable into a Unicode string.

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

```
We have =.
```

It is also possible to use numerical entities.

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

```
Ähnliches
```

Boolean Values

Boolean values are represented with 1=true or 0=false in Euler. Strings can be compared, just like numbers.

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

```
0
1
```

"and" is the operator "&&" and "or" is the operator "||", as in the C language. (The words "and" and "or" can only be used in conditions for "if".)

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

```
1
```

Boolean operators obey the rules of the matrix language.

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

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

You can use the function nonzeros() to extract specific elements form a vector. In the example, we use the conditional isprime(n).

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

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

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

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

Output Formats

The default output format of EMT prints 12 digits. To make sure that we see the default, we reset the format.

```
>defformat; pi
```

```
3.14159265359
```

Internally, EMT uses the IEEE standard for double numbers with about 16 decimal digits. To see the full number of digits, use the command "longestformat", or we use the operator "longest" to display the result in the longest format.

```
>longest pi
```

```
      3.141592653589793
```

Here is the internal hexadecimal representation of a double number.

```
>printhex(pi)
```

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

The output format can be changed permanently with a format command.

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

```
    0.33333
    3.14159
    0.84147
```

The default is format(12).

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

```
0.333333333333
```

Functions like "shortestformat", "shortformat", "longformat" work for vectors in the following way.

```
>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
```

The default format for scalars is format(12). But this can be changed.

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

```
3.1416
```

The function "longestformat" set the scalar format too.

```
>longestformat; pi
```

```
3.141592653589793
```

For reference, here is a list of the most important output formats.

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

The internal accuracy of EMT is about 16 decimal places, which is the IEEE standard. Numbers are stored in this internal format.
But the output format of EMT can be set in a flexible way.

```
>longestformat; pi,
```

```
3.141592653589793
```

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

```
3.14159
```

The default is defformat().

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

There are short operators which print only one value. The operator "longest" will print all valid digits of a number.

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

```
4.934802200544679
```

There is also a short operator for printing a result in fractional format. We have already used it above.

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

```
25/12
```

Since the internal format uses a binary way to store numbers, the value 0.1 will not be represented exactly. The error adds up a bit, as you see in the following computation.

```
>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
```

But with the default "longformat" you will not notice this. For convenience, the output of very small numbers is 0.

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

```
0
```

Expressions

Strings or names can be used to store mathematical expressions, which can be evaluated by EMT. For this, use parentheses after the expression. If you intend to use a string as an expression, use the convention to name it "fx" or "fxy" etc. Expressions take precedence over functions.
Global variables can be used in the evaluation.

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

```
      12.56637061435917
```

Parameters are assigned to x, y, and z in that order. Additional parameters can be added using assigned parameters.

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

```
-0.919535764538
```

Note that expression will always use global variables, even if there is a variable in a function with the same name. (Otherwise the evaluation of expressions in functions could have very confusing results for the user that called the function.)

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

```
36
```

If you want to use another value for "at" than the global value you need to add "at=value".

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

```
45
```

For reference, we remark that call collections (discussed elsewhere) can contain expressions. So we can make the above example as follows.

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

```
45
```

Expressions in x are often used just like functions.
Note that defining a function with the same name like a global symbolic expression deletes this variable to avoid confusion between symbolic expressions and functions.

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

```
12
```

By way of convention, symbolic or numerical expressions should be named fx, fxy etc. This naming scheme should not be used for functions.

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

A special form of an expression allows any variable as an unnamed parameter to the evaluation of the expression, not just "x", "y" etc. For this, start the expression with "@(variables) ...".

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

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

This allows to manipulate expressions in other variables for functions of EMT which need an expression in "x".
The most elementary way to define a simple function is to store its formula in a symbolic or numerical expression. If the main variable is x, the expression can be evaluated just like a function.
As you see in the following example, global variables are visible during the evaluation.

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

```
-0.475
```

All other variables in the expression can be specified in the evaluation using an assigned parameter.

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

```
-0.425
```

An expression needs not be symbolic. This is necessary, if the expression contains functions, which are only known in the numerical kernel, not in Maxima.

Symbolic Mathematics

EMT does symbolic math with the help of Maxima. For details, start with the following tutorial, or browse the reference for Maxima. Experts in Maxima should note that there are differences in the syntax between the original syntax of Maxima and the default syntax of symbolic expressions in EMT.
Symbolic math is integrated seamlessly into Euler with &. Any expression starting with & is a symbolic expression. It is evaluated and printed by Maxima.
First of all, Maxima has an "infinite" arithmetic which can handle very large numbers.

```
>$&44!
```

This way, you can compute large results exactly. Let us compute

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

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

Of course, Maxima has a more efficient function for this (as does the numerical part of EMT).

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

To learn more about a specific function double click on it. E.g., try double clicking on "&binomial" in the previous command line. This opens the documentation of Maxima as provided by the authors of that program. You will learn that the following works too.

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

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

If you want to replace x with any specific value use "with".

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

That way you can use a solution of an equation in another equation.
Symbolic expressions are printed by Maxima in 2D form. The reason for this is a special symbolic flag in the string.
As you will have seen in previous and following examples, if you have LaTeX installed, you can print a symbolic expression with Latex. If not, the following command will issue an error message.
To print a symbolic expression with LaTeX, use $ infront of & (or you may ommit &) before the command. Do not run the Maxima commands with $, if you don't have LaTeX installed.

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

Symbolic expressions are parsed by Euler. If you need a complex syntax in one expression, you can enclose the expression in "...". To use more than a simple expression is possible, but strongly discouraged.

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

For completeness, we remark that symbolic expressions can be used in programs, but need to be enclosed in quotes. Moreover, it is much more effective to call Maxima at compile time if possible.

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

Again, % refers to the previous result.
To make things easier we save the solution to a symbolic variable. Symbolic variables are defined with "&=".

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

Symbolic expressions can be used in other symbolic expressions.

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

A direct input of Maxima commands is available too. Start the command line with "::". The syntax of Maxima is adapted to the syntax of EMT (called the "compatibility mode").

```
>&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
```

If you are an expert in Maxima, you may wish to use the original syntax of Maxima. You can do this with ":::".

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

```
                                          2
                                         g
```

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

```
                          3  x
                         x  E
```

Such variables can be used in other symbolic expressions. Note, that in the following command the right hand side of &= is evaluated before the assignment to Fx.

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

```
                              5
                         125 E

                   18551.64488782208
```

```
>fx(5)
```

```
18551.6448878
```

For the evaluation of an expression with specific values of the variables, you can use the "with" operator.

The following command line also demonstrates that Maxima can evaluate an expression numerically with float().

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

```
                         10          5
                   1000 E    - 125 E

                  2.20079141499189e+7
```

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

To get the Latex code for an expression, you can use the tex command.

```
>tex(fx)
```

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

Symbolic expressions can be evaluated just like numerical expressions.

```
>fx(0.5)
```

```
0.206090158838
```

In symbolic expressions, this does not work, since Maxima does not support it. Instead, use the "with" syntax (a nicer form of the at(...) command of Maxima).

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

The assignment can also be symbolic.

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

The command solve solves symbolic expressions for a variable in Maxima. The result is a vector of solutions.

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

Compare with the numerical "solve" command in Euler, which needs a start value, and optionally a target value.

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

```
1.56155281281
```

The numerical values of the symbolic solution can be computed by evaluation of the symbolic result. Euler will read over the assignments x= etc. If you do not need the numerical results for further computations you can also let Maxima find the numerical values.

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

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

To get a specific symbolic solution, one can use "with" and an index.

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

To solve a system of equations, use a vector of equations. The result is a vector of solutions.

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

Symbolic expressions can have flags, which indicate a special treatment in Maxima. Some flags can be used as commands too, others can't. Flags are appended with "|" (a nicer form of "ev(...,flags)")

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

Functions

In EMT, functions are programs defined with the command "function". It can be a one-line function or multiline function.
A one-line function can be numerical or symbolic. A numerical one-line function is defined by ":=".

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

For an overview, we show all possible definitions for one-line functions. A function can be evaluated just like any built-in Euler function.

```
>f(2)
```

```
4.472135955
```

This function will work for vectors too, obeying the matrix language of Euler, since the expressions used in the function are vectorized.

```
>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]
```

Functions can be plotted. Instead of expressions, we need only provide the function name.
In contrast to symbolic or numerical expressions, the function name must be provided in a string.

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

```
0.786151377757
```

By default, if you need to overwrite a built-in function, you must add the keyword "overwrite". Overwriting built-in functions is dangerous and can cause problems for other functions depending on them.
You can still call the built-in function as "_...", if it is function in the Euler core.

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

```
0.707106781187
```

We better remove this redefinition of sin.

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

```
0.707106781187
```

Default Parameters

Numerical function can have default parameters.

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

Omitting this parameter uses the default value.

```
>f(4)
```

```
16
```

Setting it overwrites the default value.

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

```
80
```

An assigned parameter overwrite it too. This is used by many Euler functions like plot2d, plot3d.

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

```
16
```

If a variable is not a parameter, it must be global. One-line functions can see global variables.

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

```
24
```

But an assigned parameter overrides the global value.
If the argument is not in the list of pre-defined parameters, it must be declared with ":="!

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

```
20
```

Symbolic functions are defined with "&=". They are defined in Euler and Maxima, and work in both worlds. The defining expression is run through Maxima before the definition.

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

Symbolic functions can be used in symbolic expressions.

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

They can also be used in numerical expressions. Of course, this will only work if EMT can interpret everything inside the function.

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

```
178.635099908
```

They can be used to define other symbolic functions or expressions.

```
>function G(x) &= factor(integrate(g(x),x)); $&G(c) // integrate: mengintegralkan
>solve(&g(x),0.5)
```

```
0.703467422498
```

The following works too, since Euler uses the symbolic expression in the function g, if it does not find a symbolic variable g, and if there is a symbolic function g.

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

```
0.703467422498
```

```
>function P(x,n) &= (2*x-1)^n; $&P(x,n)
>function Q(x,n) &= (x+2)^n; $&Q(x,n)
>$&P(x,4), $&expand(%)
>P(3,4)
```

```
625
```

```
>$&P(x,4)+ Q(x,3), $&expand(%)
>$&P(x,4)-Q(x,3), $&expand(%), $&factor(%)
>$&P(x,4)*Q(x,3), $&expand(%), $&factor(%)
>$&P(x,4)/Q(x,1), $&expand(%), $&factor(%)
>function f(x) &= x^3-x; $&f(x)
```

With &= the function is symbolic, and can be used in other symbolic expressions.

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

With := the function is numerical. A good example is a definite integral like

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

which can not be evaluated symbolically.
If we redefine the function with the keyword "map" it can be used for vectors x. Internally, the function is called for all values of x once, and the results are stored in a vector.

```
>function map f(x) := integrate("x^x",1,x)
>f(0:0.5:2)
```

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

Functions can have default values for parameters.

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

Now the function can be called with or without a parameter "base".

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

```
2
6.7
```

Moreover, it is possible to use assigned parameters.

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

```
2
```

Often, we want to use functions for vectors at one place, and for individual elements at other places. This is possible with vector parameters.

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

Such a symbolic function can be used for symbolic variables.
But the function can also be used for a numerical vector.

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

```
17
```

There are also purely symbolic functions, which cannot be used numerically.

```
>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)
```

But of course, they can be used in symbolic expressions or in the definition of symbolic functions.

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

To summarize
- &= defines symbolic functions,
- := defines numerical functions,
- &&= defines purely symbolic functions.

Solving Expressions

Expressions can be solved numerically and symbolically.
To solve a simple expression of one variable, we can use the solve() function. It needs a start value to start the search. Internally, solve() uses the secant method.

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

```
1.41421356237
```

This works for symbolic expression too. Take the following function.

```
>$&solve(x^2=2,x)
>$&solve(x^2-2,x)
>$&solve(a*x^2+b*x+c=0,x)
>$&solve([a*x+b*y=c,d*x+e*y=f],[x,y])
>px &= 4*x^8+x^7-x^4-x; $&px
```

Now we search the point, where the polynomial is 2. In solve(), the default target value y=0 can be changed with an assigned variable.
We use y=2 and check by evaluating the polynomial at the previous result.

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

```
0.966715594851
2
```

Solving a symbolic expression in symbolic form returns a list of solutions. We use the symbolic solver solve() provided by Maxima.

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

The easiest way to get the numerical values is to evaluate the solution numerically just like an expression.

```
>longest sol()
```

```
    -0.6180339887498949       1.618033988749895
```

To use the solutions symbolically in other expressions, the easiest way is "with".

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

Solving systems of equations symbolically can be done with vectors of equations and the symbolic solver solve(). The answer is a list of lists of equations.

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

The function f() can see global variables. But often we want to use local parameters.

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

with a=3.

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

One way to pass the additional parameter to f() is to use a list with the function name and the parameters (the other way are semicolon parameters).

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

```
  2.54116291558
```

This does also work with expressions. But then, a named list element has to be used. (More on lists in the tutorial about the syntax of 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
```

```
>$&fourier_elim([x^2 - 1>0],[x]) // x^2-1 > 0
>$&fourier_elim([x^2 - 1<0],[x]) // x^2-1 < 0
>$&fourier_elim([x^2 - 1 # 0],[x]) // x^-1 <> 0
>$&fourier_elim([x # 6],[x])
>$&fourier_elim([x < 1, x > 1],[x]) // tidak memiliki penyelesaian
>$&fourier_elim([minf < x, x < inf],[x]) // solusinya R
>$&fourier_elim([x^3 - 1 > 0],[x])
>$&fourier_elim([cos(x) < 1/2],[x]) // ??? gagal
>$&fourier_elim([y-x < 5, x - y < 7, 10 < y],[x,y]) // sistem pertidaksamaan
>$&fourier_elim([y-x < 5, x - y < 7, 10 < y],[y,x])
>$&fourier_elim((x + y < 5) and (x - y >8),[x,y])
>$&fourier_elim(((x + y < 5) and x < 1) or  (x - y >8),[x,y])
>&fourier_elim([max(x,y) > 6, x # 8, abs(y-1) > 12],[x,y])
```

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

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

The Matrix Language

The documentation of the EMT core contains a detailed discussion on the matrix language of Euler.
Vectors and matrices are entered with square brackets, elements separated by commas, rows separated by semicolons.

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

```
              1             2
              3             4
```

The matrix product is denoted by a dot.

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

```
              3
              4
```

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

```
 [3,  4]
```

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

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

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

```
             11
             25
```

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

```
              1             0
              0             1
```

The main point of a matrix language is that all functions and operators work element for element.

```
>A.A
```

```
             7            10
            15            22
```

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

```
             1             4
             9            16
```

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

```
            37            54
            81           118
```

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

```
            37            54
            81           118
```

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

```
             1             1
             1             1
```

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

```
      0.333333      0.666667
          0.75             1
```

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

```
            -2
           2.5
```

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

```
            -2
           2.5
```

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

```
             1             0
             0             1
```

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

```
                1                 0
                0                 1
```

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

```
                1                 4
                9                16
```

This is not the matrix product, but a multiplication element by element. The same works for vectors.

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

```
                9
               16
```

If one of the operands is a vector or a scalar it is expanded in the natural way.

```
>2*A
```

```
                2                 4
                6                 8
```

E.g., if the operand is a column vector its elements are applied to all rows of A.

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

```
                1                 4
                3                 8
```

If it is a row vector it is applied to all columns of A.

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

```
                2                 6
                6                12
```

One can imagine this multiplication as if the row vector v had been duplicated to form a matrix of the same size as A.

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

```
                1                 2
                1                 2
```

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

```
                1                 4
                3                 8
```

This does also apply for two vectors where one is a row vector and the other is a column vector. We compute i*j for i,j from 1 to 5. The trick is to multiply 1:5 with its transpose. The matrix language of Euler automatically generates a table of values.

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

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

Again, remember that this is not the matrix product!

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

```
  55
```

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

```
  55
```

Even operators like < or == work in the same way.

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

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

E.g., we can count the number of elements satisfying a certain condition with the function sum().

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

```
  5
```

Euler has comparison operators, like "==", which checks for equality.
We get a vector of 0 and 1, where 1 stands for true.

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

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

From such a vector, "nonzeros" selects the non-zero elements.
In this case, we get the indices of all elements greater than 50.

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

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

Of course, we can use this vector of indices to get the corresponding values in t.

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

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

As an example, let us find all squares of the numbers 1 to 1000, which are 5 modulo 11 and 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 is not completely effective for integer computations. It uses double precision floating point internally. However, it is often very useful.
We can check for primality. Let us find out, how many squares plus 1 are primes.

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

```
112
```

The function nonzeros() works only for vectors. For matrices, there is 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
```

It returns the indices of the elements, which are not zeros.

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

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

These indices can be used to set the elements to some value.

```
>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
```

The function mset() can also set the elements at the indices to the entries of some other matrix.

```
>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
```

And it is possible to get the elements in a vector.

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

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

Another useful function is extrema, which returns the minimal and maximal values in each row of the matrix and their positions.

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

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

We can use this to extract the maximal values in each row.

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

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

This, of course, is the same as the function max().

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

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

But with mget(), we can extract the indices and use this information to extract the elements at the same positions from another matrix.

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

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

Other Matrix Functions (Building Matrix)

To build a matrix, we can stack one matrix on top of another. If both do not have the same number of columns, the shorter one will be filled with 0.

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

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

Likewise, we can attach a matrix to another side by side, if both have the same number of rows.

```
>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
```

If they do not have the same number of rows the shorter matrix is filled with 0.
There is an exception to this rule. A real number attached to a matrix will be used as a column filled with that real number.

```
>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
```

It is possible to make a matrix of row and column vectors.

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

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

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

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

The main purpose of this is to interpret a vector of expressions for column vectors.

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

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

To get the size of A, we can use the following functions.

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

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

For vectors, there is length().

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

```
9
```

There are many other functions, which generate matrices.

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

```
              1              1
              1              1
```

This can also be used with one parameter. To get a vector with another number than 1, use the following.

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

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

Also a matrix of random numbers can be generated with random (uniform distribution) or normal (Gauß distribution).

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

```
        0.66566       0.831835
          0.977       0.544258
```

Here is another useful function, which restructures the elements of a matrix into another matrix.

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

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

With the following function, we can use this and the dup function to write a rep() function, which repeats a vector n times.

```
>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]
```

The function multdup() duplicates elements of a vector.

```
>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]
```

The functions flipx() and flipy() revert the order of the rows or columns of a matrix. I.e., the function flipx() flips horizontally.

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

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

For rotations, Euler has rotleft() and rotright().

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

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

A special function is drop(v,i), which removes the elements with the indices in i from the vector v.

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

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

Note that the vector i in drop(v,i) refers to indices of elements in v, not the values of the elements. If you want to remove elements, you need to find the elements first. The function indexof(v,x) can be used to find elements x in a sorted vector 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]
```

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

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

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

There are some special functions to set diagonals or to generate a diagonal matrix.
We start with the identity matrix.

```
>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
```

Then we set the lower diagonal (-1) to 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
```

Note that we did not change the matrix A. We get a new matrix as result of setdiag().
Here is a function, which returns a tri-diagonal matrix.

```
>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
```

The diagonal of a matrix can also be extracted from the matrix. To demonstrate this, we restructure the vector 1:9 to a 3x3 matrix.

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

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

Now we can extract the diagonal.

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

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

E.g. We can divide the matrix by its diagonal. The matrix language takes care that the column vector d is applied to the matrix row by row.

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

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

Vectorization

Almost all functions in Euler work for matrix and vector input too, whenever this makes sense.
E.g., the sqrt() function computes the square root of all elements of the vector or matrix.

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

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

So you can easily create a table of values. This is one way to plot a function (the alternative uses an expression).

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

With this and the colon operator a:delta:b, vectors of values of functions can be generated easily.
In the following example, we generate a vector of values t[i] with spacing 0.1 from -1 to 1. Then we generate a vector of values of the function

$$s = t^3 - t$$

```
>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 expands operators for scalars, vectors, and matrices in the obvious way.
E.g., a column vector times a row vector expands to matrix, if an an operator is applied. In the following, v' is the transposed vector (a column vector).

```
>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
```

Note, that this is quite different from the matrix product. The matrix product is denoted with a dot "." in EMT.

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

```
55
```

By default, row vectors are printed in a compact format.

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

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

For matrices the special operator . denotes matrix multiplication, and A' denotes transposing. A 1x1 matrix can be used just like a real number.

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

```
5
25
```

To transpose a matrix we use the apostrophe.

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

```
             1
             2
             3
             4
```

So we can compute matrix A times vector b.

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

```
                30
                70
```

Note that v is still a row vector. So v'.v is different from v.v'.

```
>v'.v
```

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

v.v' computes the norm of v squared for row vectors v. The result is a 1x1 vector, which works just like a real number.

```
>v.v'
```

```
  30
```

There is also the function norm (along with many other function of Linear Algebra).

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

```
  30
```

Operators and functions obey the matrix language of Euler.
Here is a summary of the rules.
- A function applied to a vector or matrix is applied to each element.
- An operator operating on two matrices of same size is applied pairwise to the elements of the matrices.
- If the two matrices have different dimensions, both are expanded in a sensible way, so that they have the same size.
E.g., a scalar value times a vector multiplies the value with each element of the vector. Or a matrix times a vector (with *, not .) expands the vector to the size of the matrix by duplicating it.
The following is a simple case with the operator ^.

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

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

Here is a more complicated case. A row vector times a column vector expands both by duplicating.

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

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

Note that the scalar product uses the matrix product, not the *!

```
>v.v'
```

```
14
```

There are numerous functions for matrices. We give a short list. You should to consult the documentation for more information on these commands.

```
sum,prod computes the sum and products of the rows
cumsum,cumprod does the same cumulatively
computes the extremal values of each row
extrema returns a vector with the extremal information
diag(A,i) returns the i-th diagonal
setdiag(A,i,v) sets the i-th diagonal
id(n) the identity matrix
det(A) the determinant
charpoly(A) the characteristic polynomial
eigenvalues(A) the eigenvalues
```

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

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

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

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

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

To concatenate matrices and vectors there are the operators "|" and "_".

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

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

The elements of a matrix are referred with "A[i,j]".

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

```
6
```

For row or column vectors, v[i] is the i-th element of the vector. For matrices, this returns the complete i-th row of the matrix.

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

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

The indices can also be row vectors of indices. : denotes all indices.

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

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

A short form for : is omitting the index completely.

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

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

For purposes of vectorization, the elements of a matrix can be accessed as if they were vectors.

```
>A{4}
```

```
4
```

A matrix can also be flattened, using the redim() function. This is implemented in the function 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]
```

To use matrices for tables, let us reset to the default format, and compute a table of sine and cosine values. Note that angles are in radians by default.

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

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

Now we append columns to a matrix.

```
>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

Using the matrix language, we can generate several tables of several functions at once.
In the following example, we compute t[j]^i for i from 1 to n. We get a matrix, where each row is a table of t^i for one i. I.e., the matrix has the elements

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

A function which does not work for vector input should be "vectorized". This can be achieved by the "map" keyword in the function definition. Then the function will be evaluated for each element of a vector parameter. The numerical integration integrate() works only for scalar interval bounds. So we need to vectorize it.

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

The "map" keyword vectorizes the function. The function will now work
for vectors of numbers.

```
>f([1:5])
```

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

Sub-Matrices and Matrix-Elements

To access a matrix element, use the bracket notation.

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

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

We can access a complete line of a matrix.

```
>A[2]
```

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

In case of row or column vectors, this returns an element of the vector.

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

```
2
```

To make sure, you get the first row for a 1xn and a mxn matrix, specify all columns using an empty second index.

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

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

If the index is a vector of indices, Euler will return the corresponding rows of the matrix.
Here we want the first and second row of A.

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

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

We can even reorder A using vectors of indices. To be precise, we do not change A here, but compute a reordered version of A.

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

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

The index trick works with columns too.
This example selects all rows of A and the second and third column.

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

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

For abbreviation ":" denotes all row or column indices.

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

```
              3
              6
              9
```

Alternatively, leave the first index empty.

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

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

We can also get the last line of A.

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

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

Now let us change elements of A by assigning a submatrix of A to some value. This does in fact change the stored matrix A.

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

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

We can also assign a value to a row of A.

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

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

We can even assign to a sub-matrix if it has the proper size.

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

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

Moreover, some shortcuts are allowed.

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

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

A warning: Indices out of bounds return empty matrices, or an error message, depending on a system setting. The default is an error message. Remember, however, that negative indices may be used to access the elements of a matrix counting from the end.

```
>A[4]
```

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

Sorting and Shuffling

The function sort() sorts a row vector.

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

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

It is often necessary to know the indices of the sorted vector in the original vector. This can be used to reorder another vector in the same way.
Let us shuffle a vector.

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

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

The indices contain the proper order of v.

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

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

This works for string vectors too.

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

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

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

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

As you see, the position of double entries is somewhat random.

```
>ind
```

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

The function unique returns a sorted list of unique elements of a vector.

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

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

This works for string vectors too.

```
>unique(s)
```

```
 a
 aa
 d
 e
```

Linear Algebra

EMT has lots of functions to solve linear systems, sparse systems, or regression problems.
For linear systems Ax=b, you can use the Gauss algorithm, the inverse matrix or a linear fit. The operator A\b uses a version of the Gauss algorithm.

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

```
            -4
           4.5
```

For another example, we generate a 200x200 matrix and the sum of its rows. Then we solve Ax=b using the inverse matrix. We measure the error as the maximal deviation of all elements from 1, which of course is the correct solution.

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

```
    8.790745908981989e-13
```

If the system does not have a solution, a linear fit minimizes the norm of the error Ax-b.

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

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

The determinant of this matrix is 0.

```
>det(A)
```

```
 0
```

Symbolic Matrices

Maxima has symbolic matrices. Of course, Maxima can be used for such simple linear algebra problems. We can define the matrix for Euler and Maxima with &:=, and then use it in symbolic expressions. The usual [...] form to define matrices can be used in Euler to define symbolic matrices.

```
>A &= [a,1,1;1,a,1;1,1,a]; $A
>$&det(A), $&factor(%)
>$&invert(A) with a=0
>A &= [1,a;b,2]; $A
```

Like all symbolic variables, these matrices can be used in other symbolic expressions.

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

The eigenvalues can also be computed automatically. The result is a vector with two vectors of eigenvalues and multiplicities.

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

To extract a specific eigenvector needs careful indexing.

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

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

Symbolic matrices can be evaluated in Euler numerically just like other symbolic expressions.

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

```
          1             4
          5             2
```

In symbolic expressions, use with.

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

Access to rows of symbolic matrices work just like with numerical matrices.

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

A symbolic expression can contain an assignment. And that changes the matrix A.

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

There are symbolic functions in Maxima to create vectors and matrices. For this, refer to the documentation of Maxima or to the tutorial about Maxima in EMT.

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

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

The result can be evaluated numerically in Euler. For more information about Maxima, see the introduction to Maxima.

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

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

Euler has also a powerful function xinv(), which makes a bigger effort and gets more exact results.
Note, that with &:= the matrix B has been defined as symbolic in symbolic expressions and as numerical in numerical expressions. So we can use it here.

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

```
                        1                       0
                        0                       1
```

E.g. the eigenvalues of A can be computed numerically.

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

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

Or symbolically. See the tutorial about Maxima for details on this.

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

Numerical Values in symbolic Expressions

A symbolic expression is just a string containing an expression. If we want to define a value both for symbolic expressions and for numerical expressions, we must use "&:=".

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

```
            1       3.14159
            4             5
```

There is still a difference between the numerical and the symbolic form. When transferring the matrix to the symbolic form, fractional approximations for reals will be used.

```
>$&A
```

To avoid this, there is the function "mxmset(variable)".

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

Maxima can also compute with floating point numbers, and even with big floating numbers with 32 digits. The evaluation is much slower, however.

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

The precision of the big floating point numbers can be changed.

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

```
          3.14159265358979323846264338327950288419716939937510582097494\
  4592307816406286208998628034825342117068b0
```

A numerical variable can be used in any symbolic expressions using "@var".
Note that this is only necessary, if the variable has been defined with ":=" or "=" as a numerical variable.

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

Demo - Interest Rates

Below, we use Euler Math Toolbox (EMT) for the calculation of interest rates. We do that numerically and symbolically to show you how Euler can be used to solve real life problems.
Assume you have a seed capital of 5000 (say in dollars).

```
>K=5000
```

```
  5000
```

Now we assume an interest rate of 3% per year. Let us add one simple rate and compute the result.

```
>K*1.03
```

```
  5150
```

Euler would understand the following syntax too.

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

```
  5150
```

But it is easier to use the factor

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

```
  1.03
  5150
```

For 10 years, we can simply multiply the factors and get the final value with compound interest rates.

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

```
6719.58189672
```

For our purposes, we can set the format to 2 digits after the decimal dot.

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

```
    6719.58
```

Let us print that rounded to 2 digits in a complete sentence.

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

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

What if we want to know the intermediate results from year 1 to year 9? For this, Euler's matrix language is a big help. You do not have to write a loop, but simply enter

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

```
Real 1 x 11 matrix

    5000.00     5150.00     5304.50     5463.64     ...
```

How does this miracle work? First the expression 0:10 returns a vector of integers.

```
>short 0:10
```

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

Then all operators and functions in Euler can be applied to vectors element for element. So

```
>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]
```

is a vector of factors q^0 to q^10. This is multiplied by K, and we get the vector of values.

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

Of course, the realistic way to compute these interest rates would be to round to the nearest cent after each year. Let us add a function for this.

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

Let us compare the two results, with and without rounding.

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

```
                1271.61
              1271.6071
```

Now there is no simple formula for the n-th year, and we must loop over the years. Euler provides many solutions for this.
The easiest way is the function iterate, which iterates a given function a number of times.

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

```
  Real 1 x 11 matrix

      5000.00      5150.00      5304.50      5463.64      ...
```

We can print that in a friendly way, using our format with fixed decimal places.

```
>VKr'
```

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

To get a specific element of the vector, we use indices in square brackets.

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

```
      5150.00
      5000.00      5150.00      5304.50
```

Surprisingly, we can also use a vector of indices. Remember that 1:3 produced the vector [1,2,3].
Let us compare the last element of the rounded values with the full values.

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

```
      6719.60
      6719.58
```

The difference is very small.

Solving Equations

Now we take a more advanced function, which adds a certain rate of money each year.

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

We do not have to specify q or R for the definition of the function. Only if we run the command, we have to define these values. We select R=200.

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

```
 Real 1 x 11 matrix

     5000.00     5350.00     5710.50     6081.82     ...
```

What if we remove the same amount each year?

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

```
 Real 1 x 11 matrix

     5000.00     4950.00     4898.50     4845.45     ...
```

We see that the money decreases. Obviously, if we get only 150 of interest in the first year, but remove 200, we lose money each year.
How can we determine the number of years the money will last? We would have to write a loop for this. The easiest way is to iterate long enough.

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

```
 Real 1 x 51 matrix

     5000.00     4950.00     4898.50     4845.45     ...
```

Using the matrix language, we can determine the first negative value in the following way.

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

```
       48.00
```

The reason for this is that nonzeros(VKR<0) returns a vector of indices i, where VKR[i]<0, and min computes the minimal index.
Since vectors always start with index 1, the answer is 47 years.
The function iterate() has one more trick. It can take an end condition as an argument. Then it will return the value and the number of iterations.

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

```
      -19.83
       47.00
```

Let us try to answer a more ambiguous question. Assume we know that the value is 0 after 50 years. What would be the interest rate?
This is a question, which can only be answered numerically. Below, we will derive the necessary formulas. Then you will see that there is no easy formula for the interest rate. But for now, we aim for a numerical solution.
The first step is to define a function which does the iteration n times. We add all parameters to this function.

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

The iteration is just as above

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

But we do longer use the global value of R in our expression. Functions like iterate() have a special trick in Euler. You can pass the values of variables in the expression as semicolon parameters. In this case P and R.
Moreover, we are only interested in the last value. So we take the index [-1].
Let us try a test.

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

```
    -19.83
```

Now we can solve our problem.

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

```
    3.15
```

The solve routine solves expression=0 for the variable x. The answer is 3.15% per year. We take the start value of 3% for the algorithm. The solve() function always needs a start value.
We can use the same function to solve the following question: How much can we remove per year so that the seed capital is exhausted after 20 years assuming an interest rate of 3% per year.

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

```
    -336.08
```

Note that you cannot solve for the number of years, since our function assumes n to be an integer value.

Symbolic Solutions to the Interest Rate Problem

We can use the symbolic part of Euler to study the problem. First we define our function onepay() symbolically.

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

We can now iterate this.

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

We see a pattern. After n periods we have

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

The formula is the formula for the geometric sum, which is known to Maxima.

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

This is a bit tricky. The sum is evaluated with the flag "simpsum" to reduce it to the quotient.
Let us make a function for this.

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

The function does the same as our function f before. But it is more effective.

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

```
      -19.82504734650985
      -19.82504734652684
```

We can now use it to ask for the time n. When is our capital exhausted? Our initial guess is 30 years.

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

```
        20.51
```

This answer says that it will be negative after 21 years.
We can also use the symbolic side of Euler to compute formulas for the payments.
Assume we get a loan of K, and pay n payments of R (starting after the first year) leaving a residual debt of Kn (at the time of the last payment). The formula for this is clearly

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

Usually this formula is given in terms of

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

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

We can solve for the rate R symbolically.

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

As you can see from the formula, this function returns a floating point error for i=0. Euler plots it nevertheless. Of course, we have the following limit.

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

Clearly, without interest we have to pay back 10 rates of 500.
The equation can also be solved for n. It looks nicer, if we apply some simplification to it.

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

BAB 2

KB PEKAN 4: MENGGUNAKAN EMT UNTUK MENGAMBAR GRAFIK 2 DIMENSI (2D)

[a4paper,10pt]article eumat

Nama : Fransisca Renita Pejoresa
NIM : 22305144012
Kelas : Matematika E 2022

Kegiatan Belajar Visualisasi 2D dengan EMT

Menggambar Grafik 2D dengan EMT

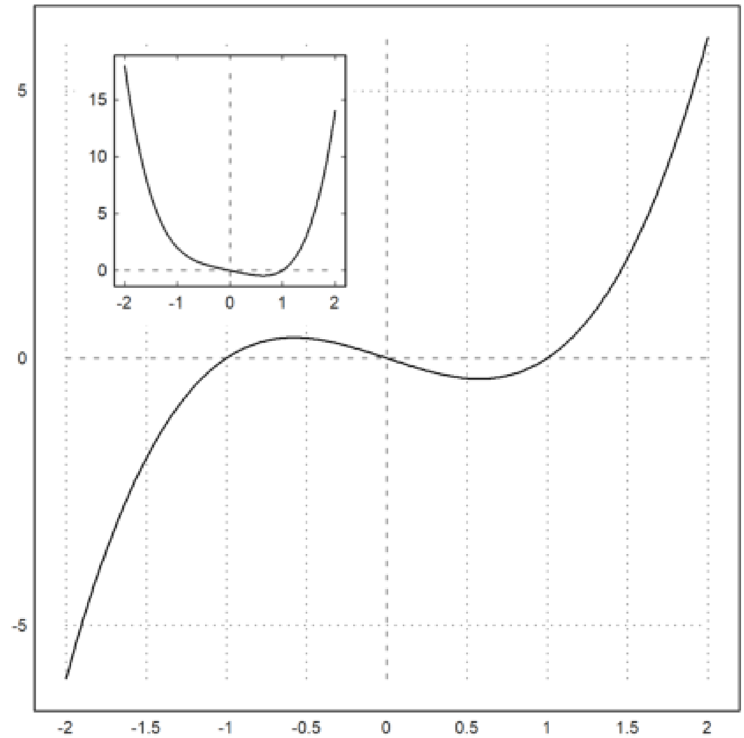
Notebook ini menjelaskan tentang cara menggambar berbagaikurva dan grafik 2D dengan software EMT. EMT menyediakan fungsi plot2d() untuk menggambar berbagai kurva dan grafik dua dimensi (2D).

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.

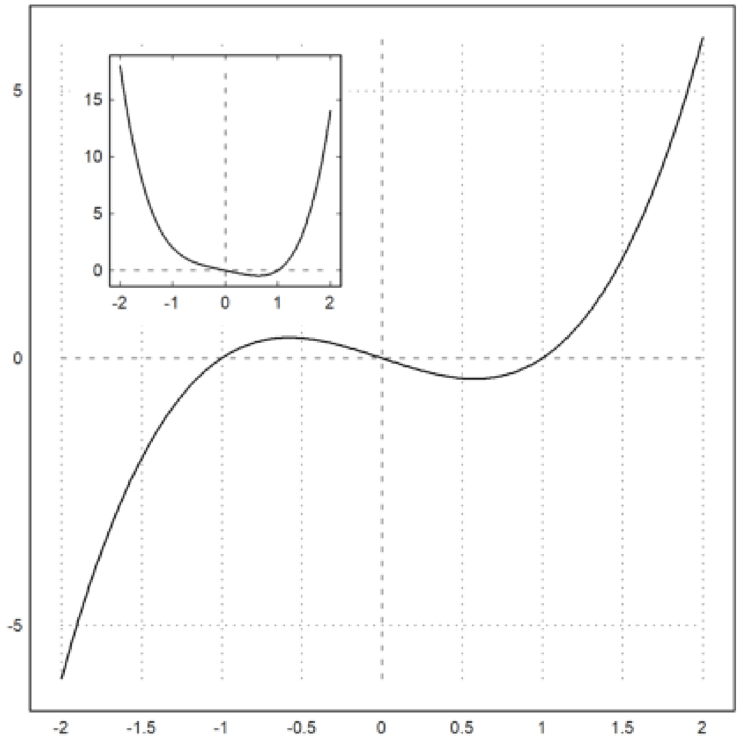
```
> clg; // clear screen
>window(0,0,1024,1024); // use all of the window
>setplot(0,1,0,1); // set plot coordinates
>hold on; // start overwrite mode
>n=100; X=random(n,2); Y=random(n,2);  // get random points
>colors=rgb(random(n),random(n),random(n)); // get random colors
>loop 1 to n; color(colors[#]);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



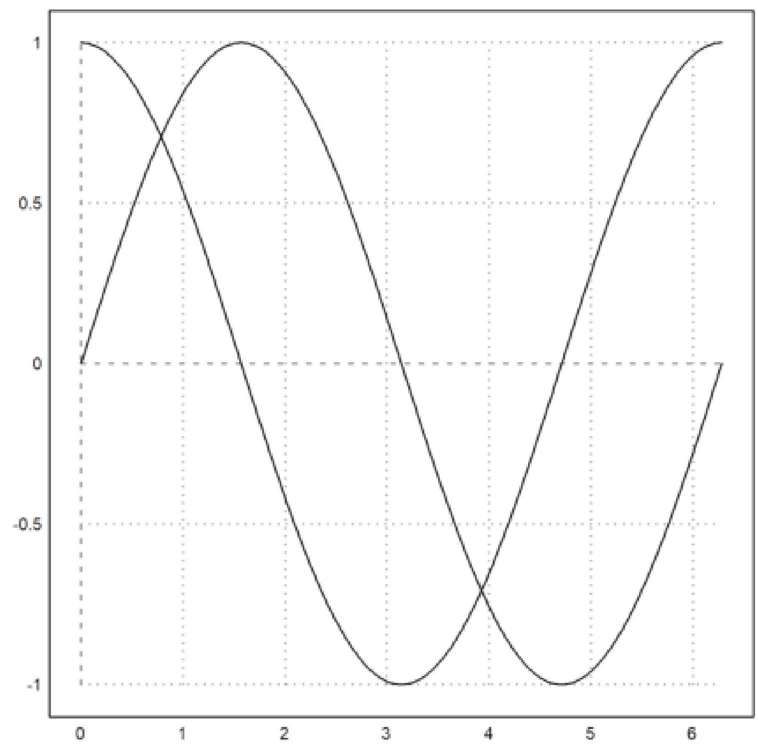
```
>hold off;
>window(ow);
```

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

Aspek Plot

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.

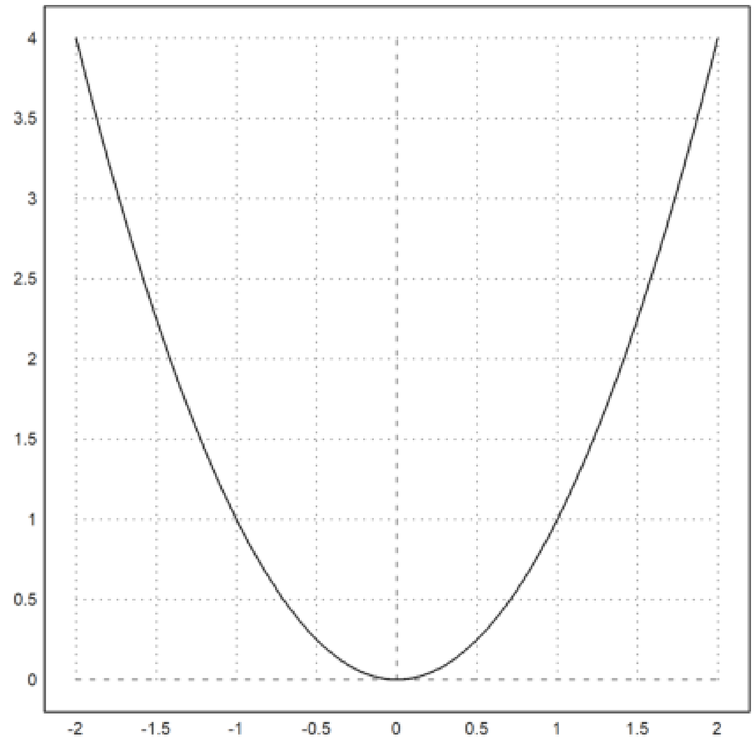
Plot 2D di Euler

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.

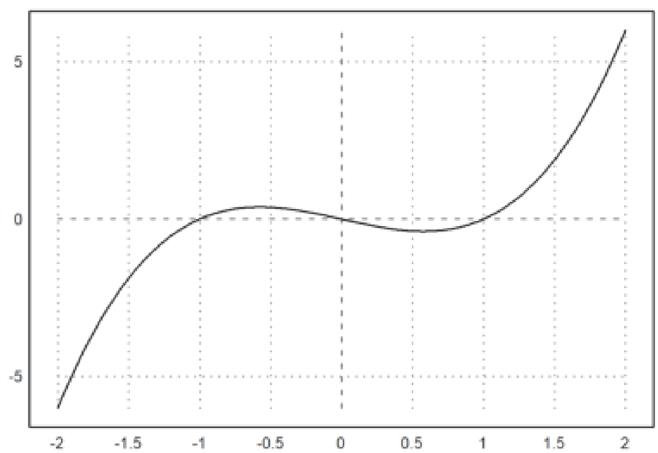
Plot Ekspresi atau Variabel

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.

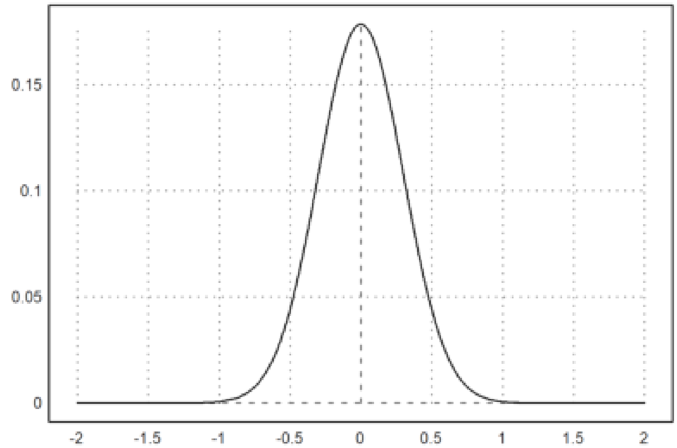
```
>plot2d("x^2"):
```




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

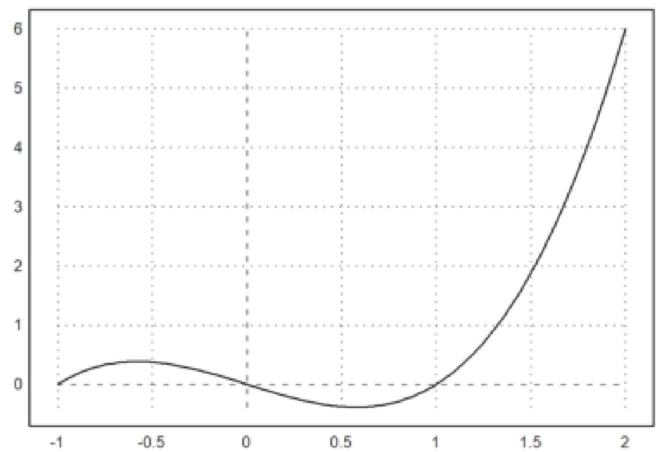



```
>a:=5.6; plot2d("exp(-a*x^2)/a"); insimg(30); // menampilkan gambar hasil plot setinggi 25
```

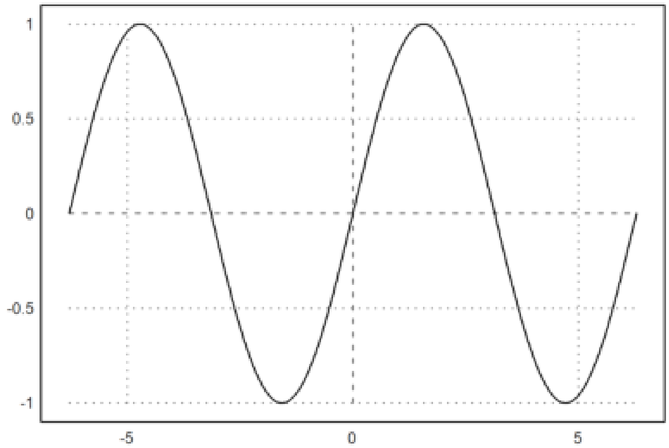


Dari beberapa contoh sebelumnya Anda dapat melihat bahwa Gambaran gambar plot menggunakan sumbu X dengan rentang nilai dari -2 sampai dengan 2. Untuk mengubah rentang nilai X dan Y, Anda dapat menambahkan nilai batas X (dan Y) di belakang ekspresi yang digambar.
Rentang plot diatur dengan parameter yang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

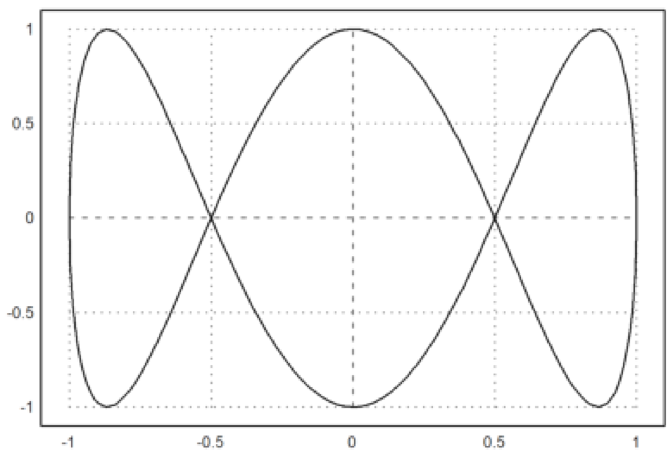
```
>plot2d("x^3-x",-1,2):
```




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

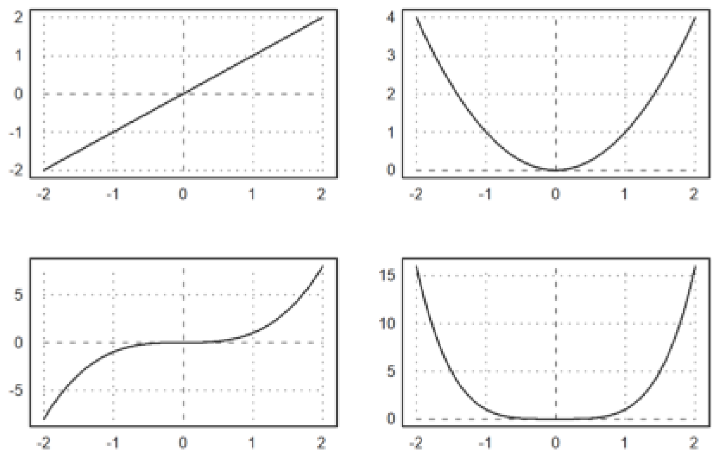



```
>plot2d("cos(x)","sin(3*x)",xmin=0,xmax=2pi):
```



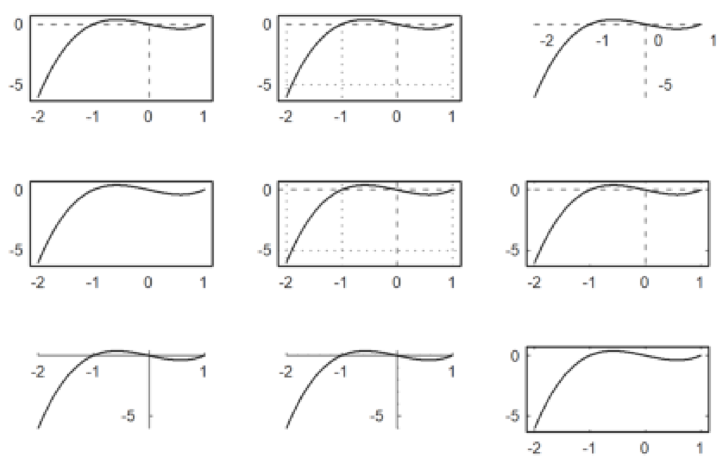
Alternatif untuk titik dua adalah perintah insimg(baris), yang menyisipkan plot yang menempati sejumlah baris teks tertentu.
Dalam opsi, plot dapat diatur untuk muncul
- di jendela terpisah yang dapat diubah ukurannya,
- di jendela buku catatan.
Lebih banyak gaya dapat dicapai dengan perintah plot tertentu.
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):
```



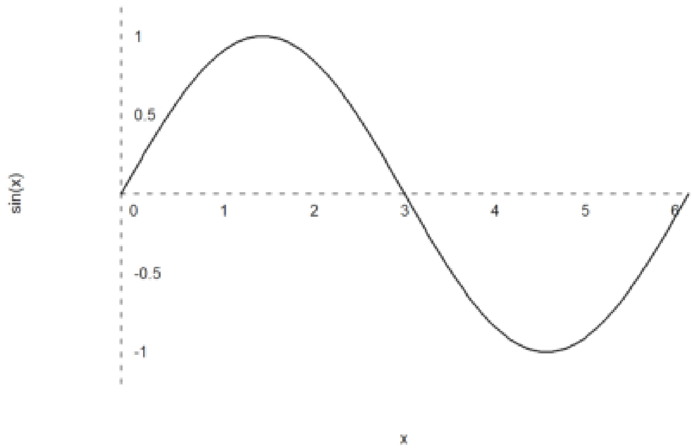
Di plot2d(), ada gaya alternatif yang tersedia dengan grid=x. Untuk gambaran umum, kami menunjukkan berbagai gaya kisi dalam satu gambar (lihat di bawah untuk perintah figure()). Gaya kisi=0 tidak disertakan. Ini menunjukkan tidak ada grid dan tidak ada bingkai.

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

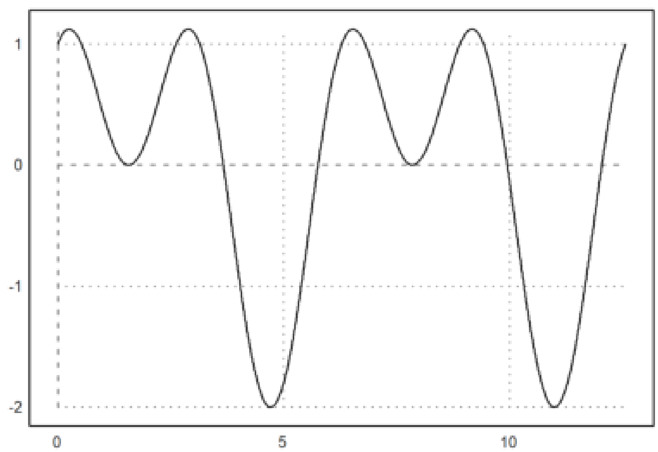


Jika argumen ke plot2d() adalah ekspresi yang diikuti oleh empat angka, angka-angka ini adalah rentang x dan y untuk plot.
Atau, a, b, c, d dapat ditentukan sebagai parameter yang ditetapkan sebagai a=... dll.
Dalam contoh berikut, kita mengubah gaya kisi, menambahkan label, dan menggunakan label vertikal untuk sumbu y.

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

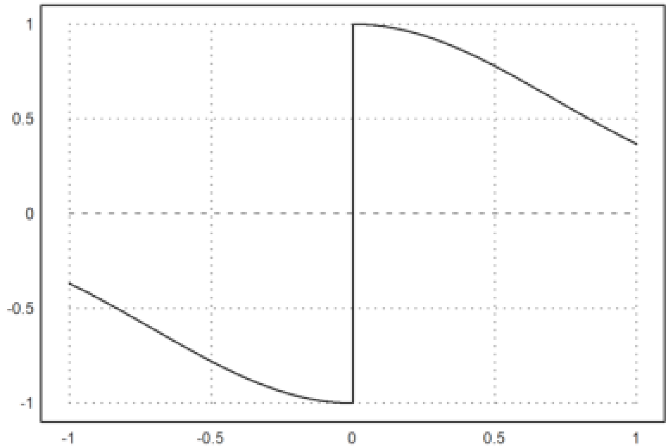



```
>plot2d("sin(x)+cos(2*x)",0,4pi):
```

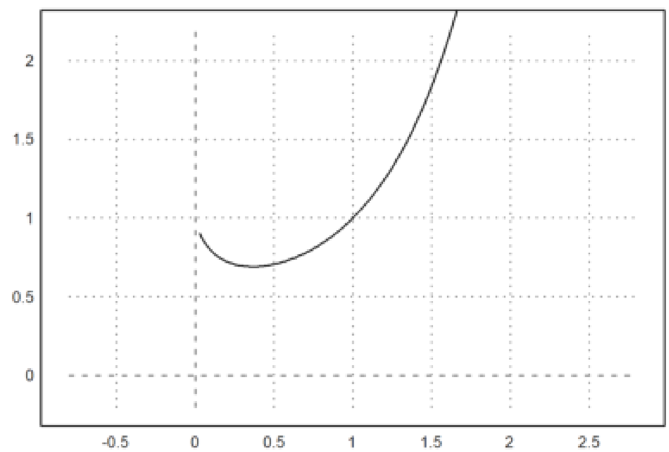


Gambar yang dihasilkan dengan memasukkan plot ke dalam jendela teks disimpan di direktori yang sama dengan buku catatan, secara default di subdirektori bernama "gambar". Mereka juga digunakan oleh ekspor HTML.
Anda cukup menandai gambar apa saja dan menyalinnya ke clipboard dengan Ctrl-C. Tentu saja, 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):
```

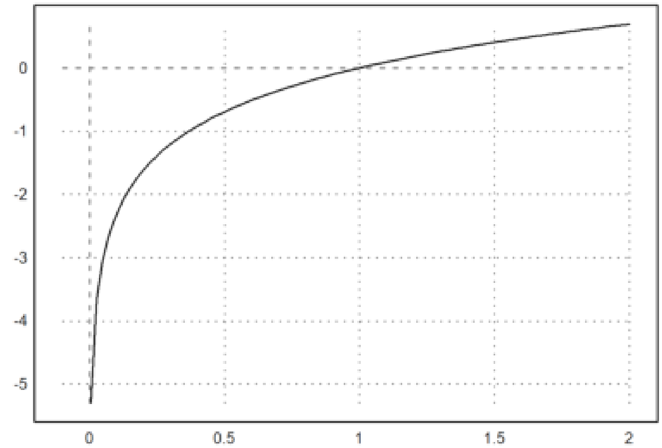



```
>plot2d("x^x",r=1.2,cx=1,cy=1):
```



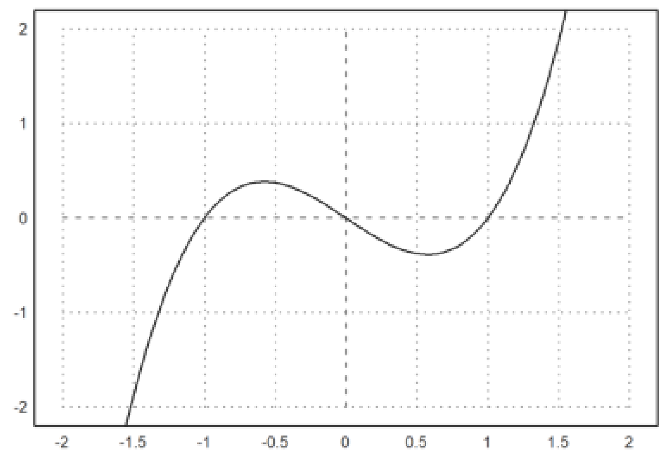
Perhatikan bahwa x^x tidak didefinisikan untuk x<=0. Fungsi plot2d menangkap kesalahan ini, dan mulai merencanakan segera setelah fungsi didefinisikan. Ini berfungsi untuk semua fungsi yang mengembalikan NAN keluar dari jangkauan definisinya.

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

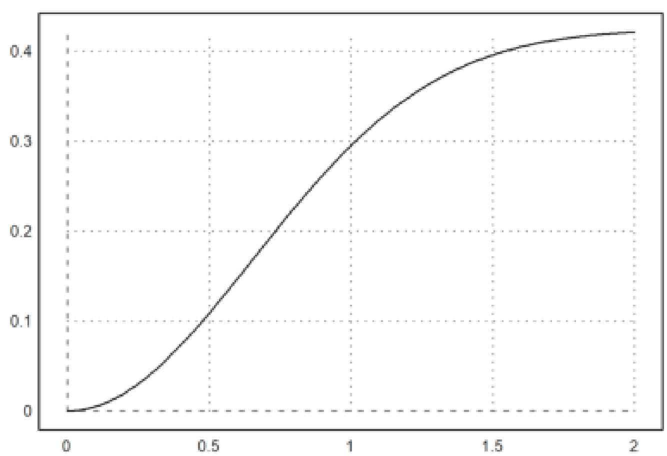


Parameter square=true (atau >square) memilih y-range secara otomatis sehingga hasilnya adalah jendela plot persegi. Perhatikan bahwa secara default, Euler menggunakan ruang persegi di dalam jendela plot.

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

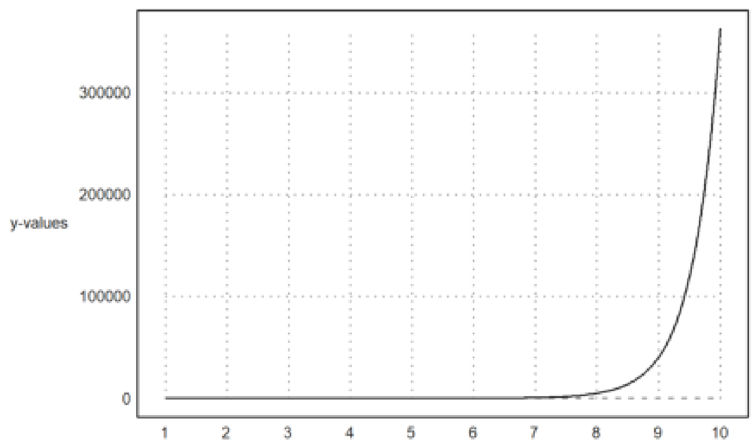


```
>plot2d(''integrate("sin(x)*exp(-x^2)",0,x)'',0,2): // plot integral
```



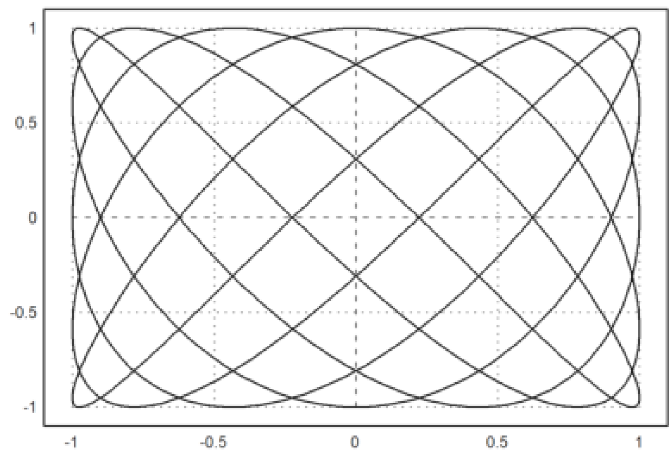
Jika Anda membutuhkan lebih banyak ruang untuk label-y, panggil shrinkwindow() dengan parameter yang lebih kecil, atau tetapkan nilai positif untuk "lebih kecil" di plot2d().

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

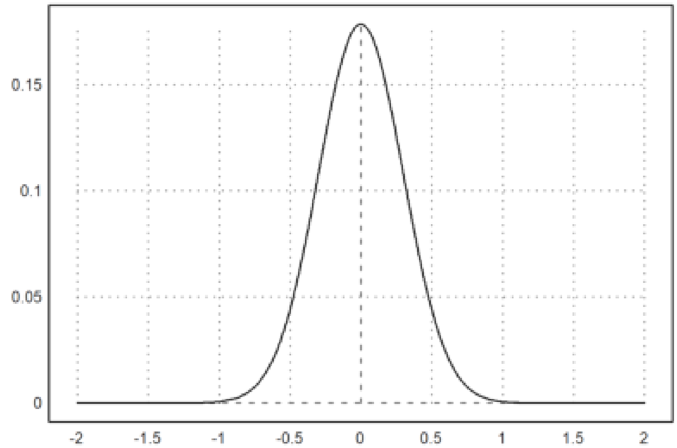


Ekspresi simbolik juga dapat digunakan, karena disimpan sebagai ekspresi string sederhana.

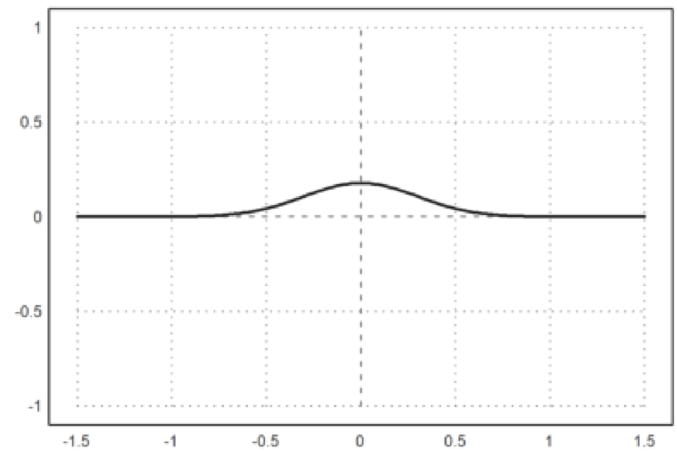
```
>x=linspace(0,2pi,1000); plot2d(sin(5x),cos(7x)):
```



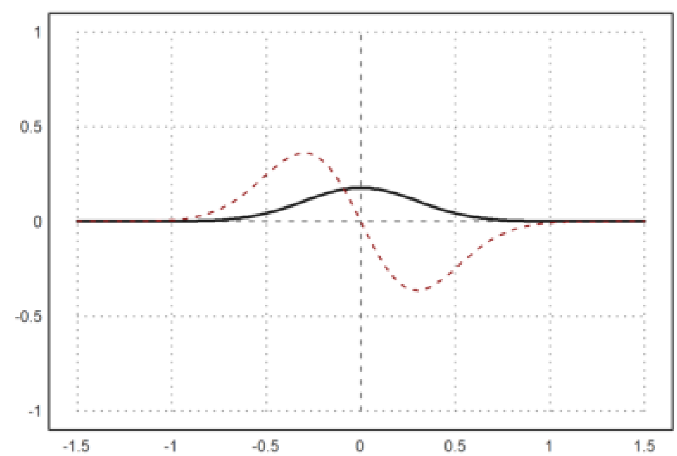

```
>a:=5.6; expr &= exp(-a*x^2)/a; // define expression
>plot2d(expr,-2,2): // plot from -2 to 2
```



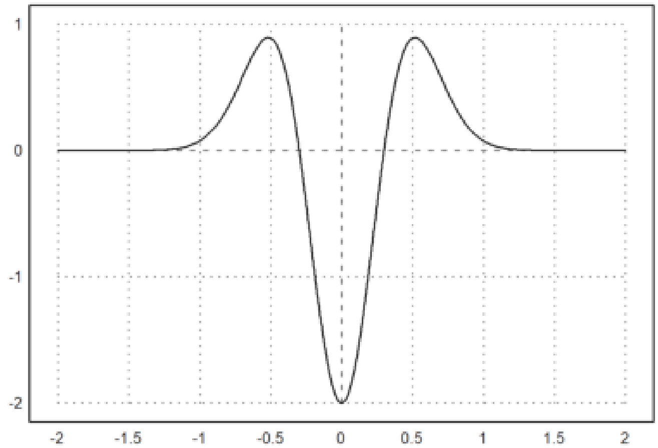

```
>plot2d(expr,r=1,thickness=2): // plot in a square around (0,0)
```



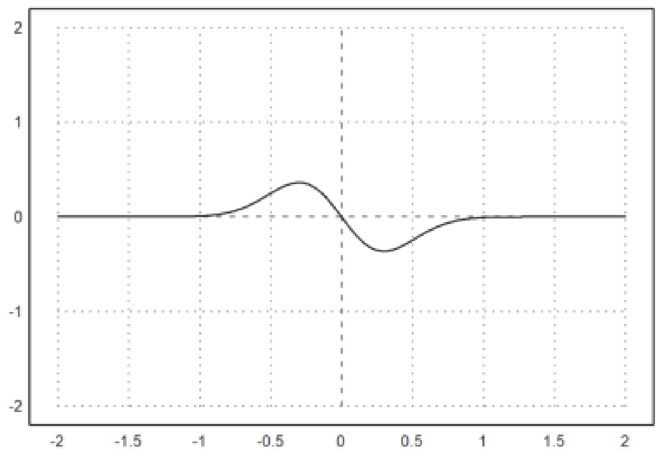

```
>plot2d(&diff(expr,x),>add,style="--",color=red): // add another plot
```



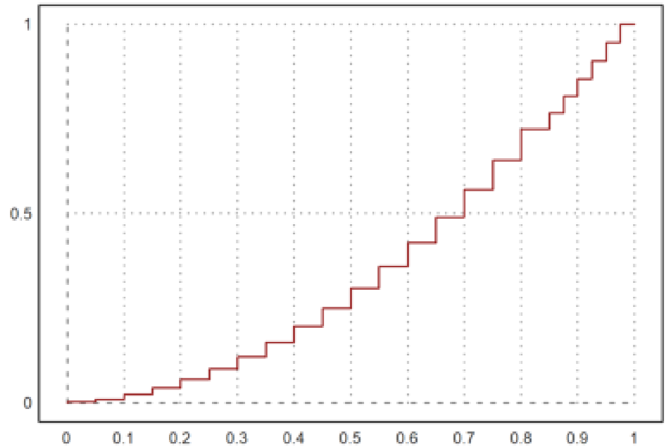
```
>plot2d(&diff(expr,x,2),a=-2,b=2,c=-2,d=1): // plot in rectangle
```



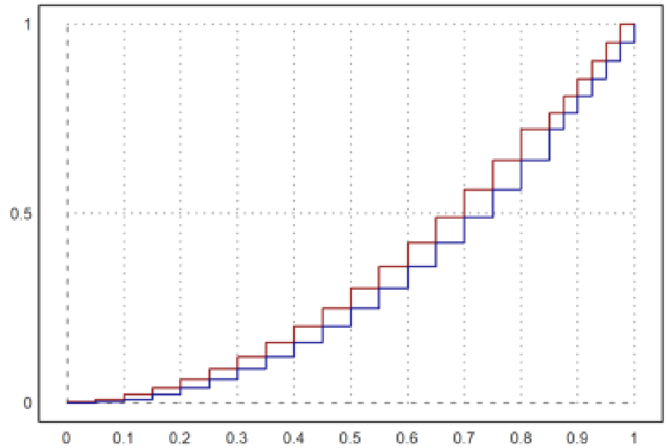

```
>plot2d(&diff(expr,x),a=-2,b=2,>square): // keep plot square
```




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



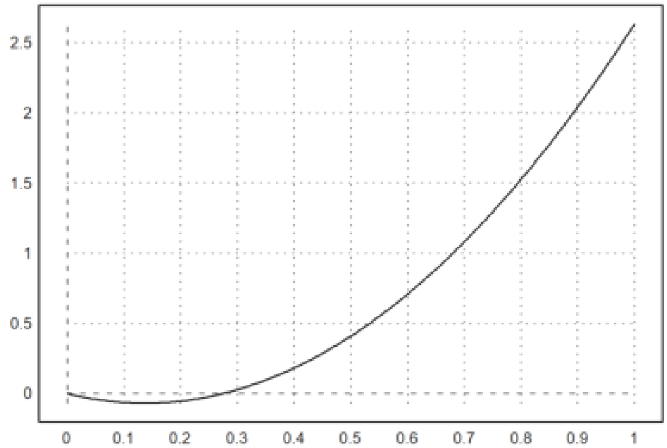

```
>plot2d("x^2",>add,steps=2,color=blue,n=10):
```



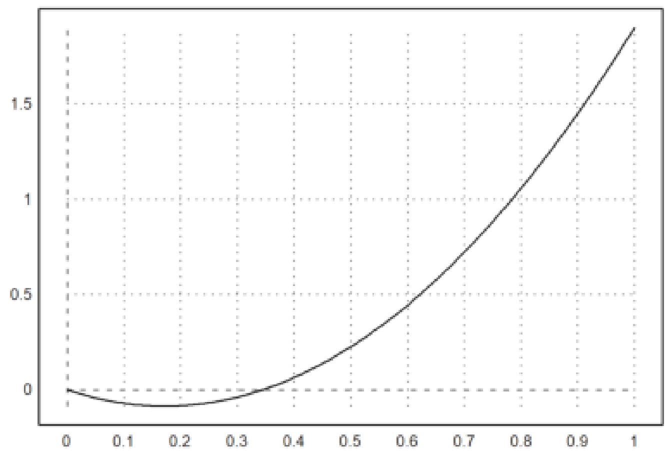
Fungsi dalam satu Parameter

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.

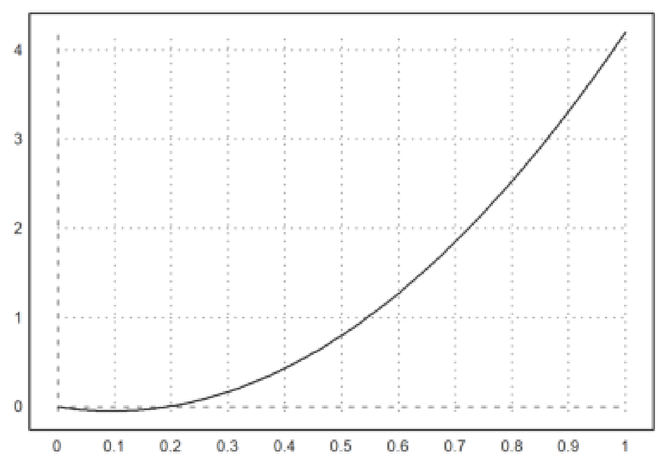
```
>function f(x,a) := x^2/a+a*x^2-x; // define a function
>a=0.3; plot2d("f",0,1;a): // plot with a=0.3
```



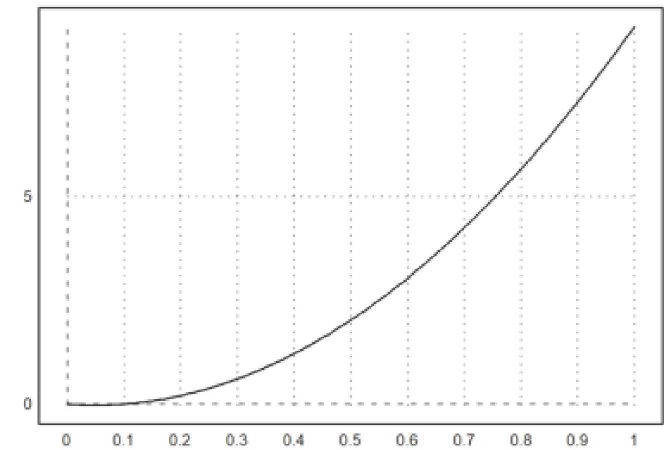

```
>plot2d("f",0,1;0.4): // plot with a=0.4
```



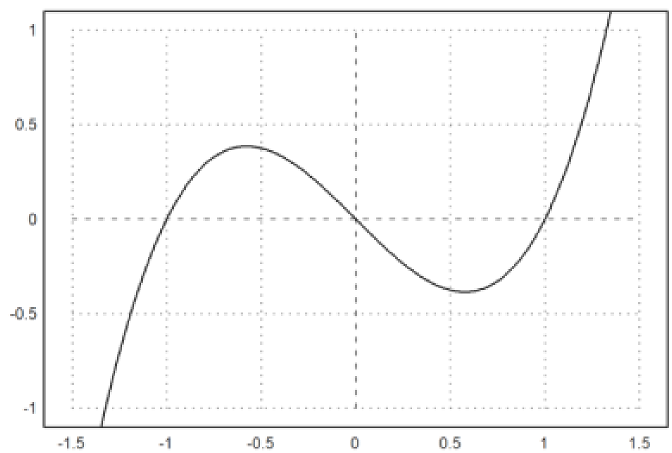

```
>plot2d({{"f",0.2}},0,1): // plot with a=0.2
```



```
>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)
```

```
  x
 x   (log(x) + 1)
```

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

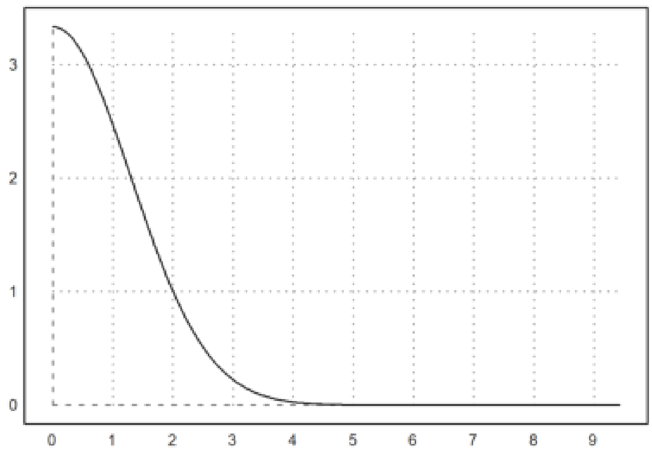
```
Function f needs at least one argument!
Use: f (x)
Error in:
plot2d(f,0,2): ...
                 ^
```

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

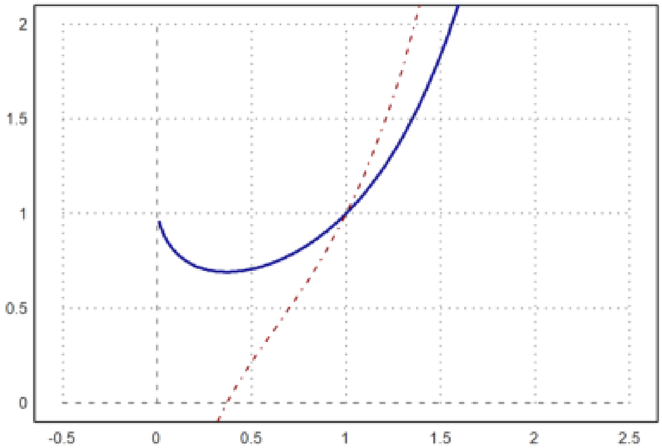
```
>expr &= sin(x)*exp(-x)
```

```
                              - x
                             E    sin(x)
```

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



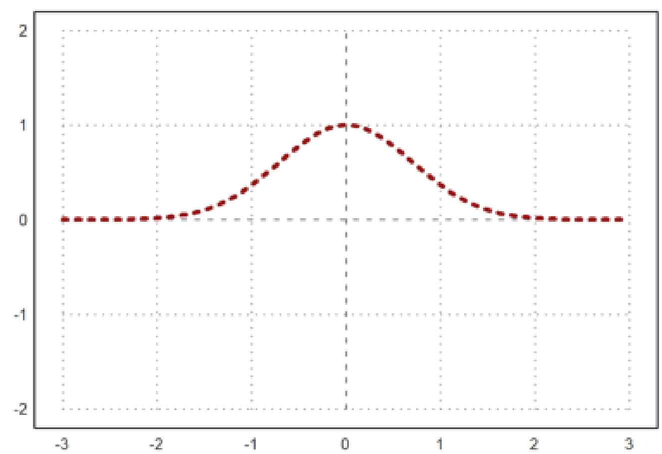

```
>function f(x) &= x^x;
>plot2d(f,r=1,cx=1,cy=1,color=blue,thickness=2);
>plot2d(&diff(f(x),x),>add,color=red,style="-.-"):
```



Untuk gaya garis ada berbagai pilihan.
- gaya="...". Pilih dari "-", "–", "-.", ".", ".-.", "-.-".
- warna: Lihat di bawah untuk warna.
- ketebalan: Default adalah 1.

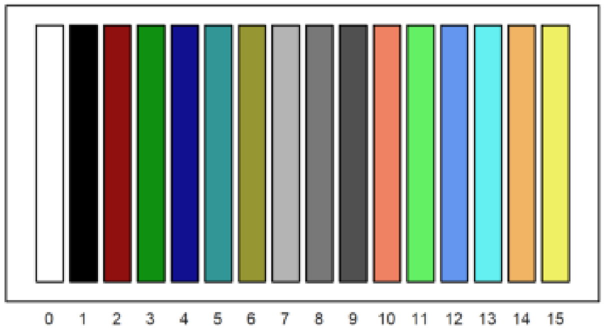
Warna dapat dipilih sebagai salah satu warna default, atau sebagai warna RGB.
- 0.15: indeks warna default.
- konstanta warna: putih, hitam, merah, hijau, biru, cyan, zaitun, abu-abu muda, abu-abu, abu-abu tua, oranye, hijau muda, pirus, biru muda, oranye terang, kuning
- rgb(merah, hijau, biru): parameter adalah real dalam [0,1].

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



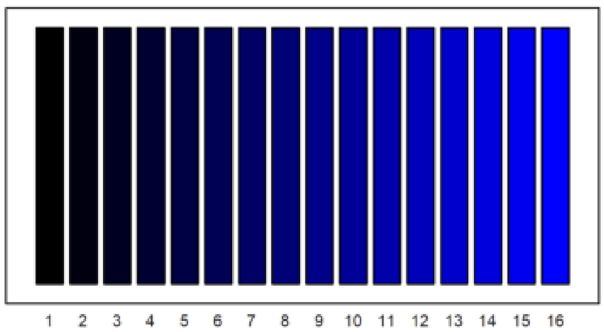
Berikut adalah tampilan warna EMT yang telah ditentukan sebelumnya.

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



Kita bisa memilih warna apapun

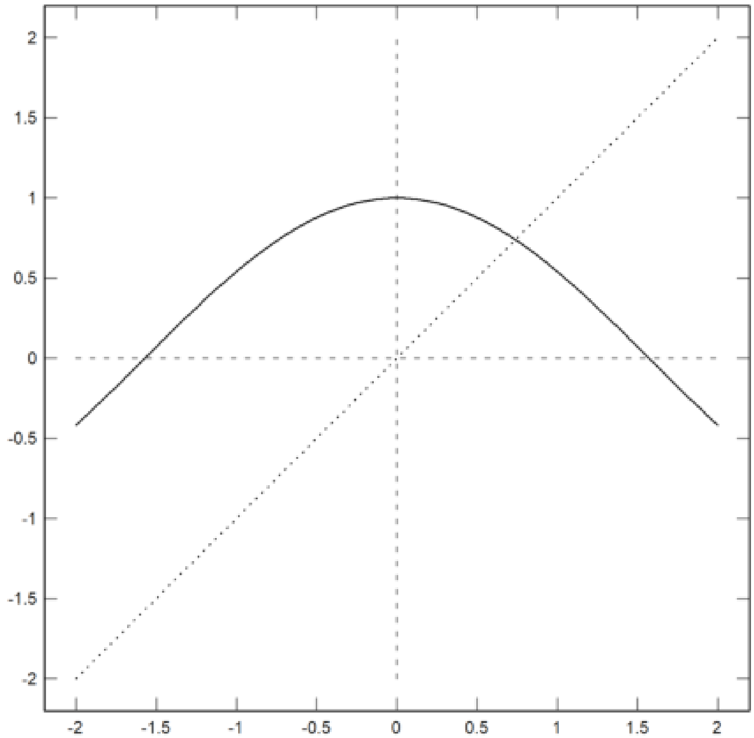
```
>columnsplot(ones(1,16),grid=0,color=rgb(0,0,linspace(0,1,15))):
```



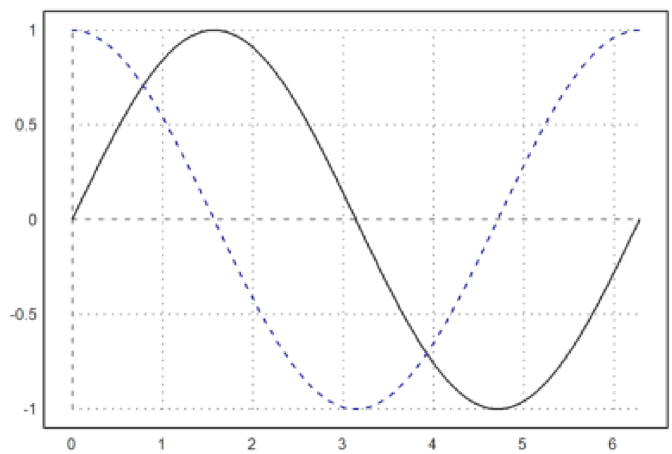
Menggambar Beberapa Kurva pada bidang koordinat yang sama

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):
```

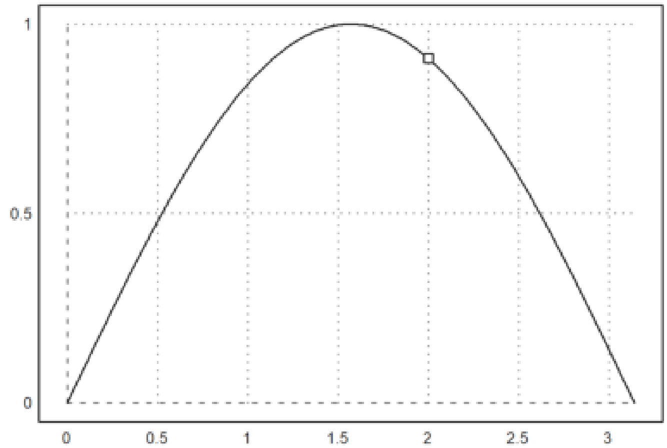



```
>aspect(1.5); plot2d("sin(x)",0,2pi); plot2d("cos(x)",color=blue,style="--",>add):
```



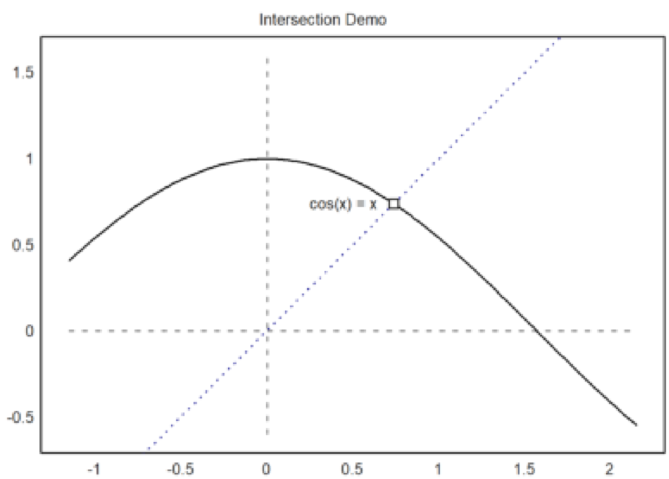
Salah satu kegunaan >add adalah untuk menambahkan titik pada kurva.

```
>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):
```

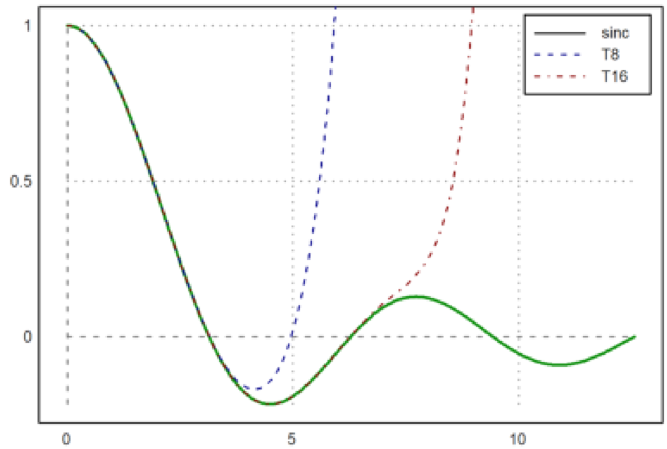


Dalam demo berikut, kami memplot fungsi sinc(x)=sin(x)/x dan ekspansi Taylor ke-8 dan ke-16. Kami menghitung ekspansi ini menggunakan Maxima melalui ekspresi simbolis.
Plot ini dilakukan dalam perintah multi-baris berikut dengan tiga panggilan ke plot2d(). Yang kedua dan yang ketiga memiliki set flag >add, yang membuat plot menggunakan rentang sebelumnya.
Kami menambahkan kotak label yang menjelaskan fungsi.

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

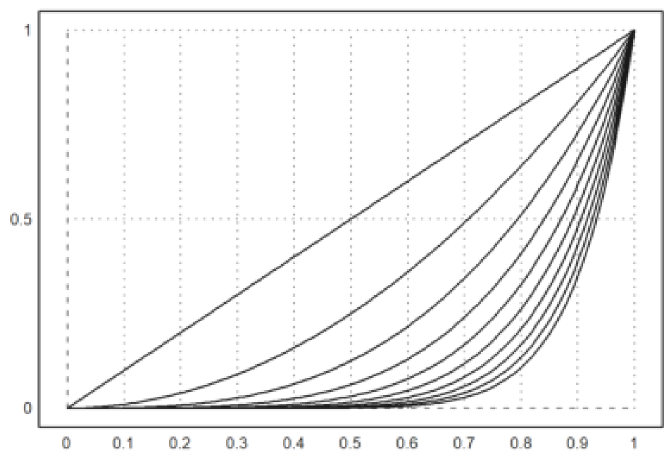
$$\frac{x^4}{120} - \frac{x^2}{6} + 1$$

```
>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]):
```



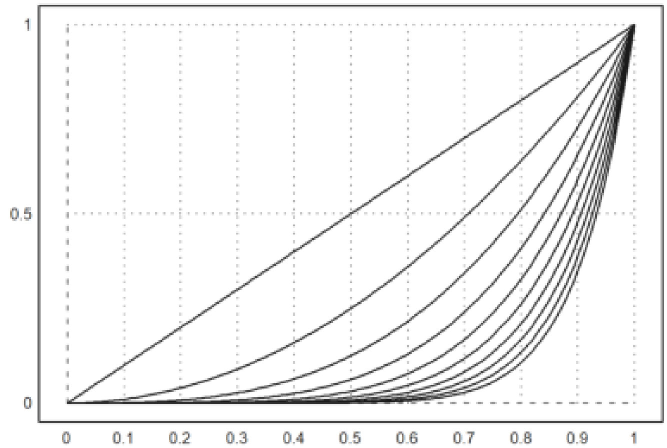
Dalam contoh berikut, kami menghasilkan Bernstein-Polinomial.
lateks: B_i(x) = \binom{n}{i} x^i (1-x)^{n-i}

```
>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;
```



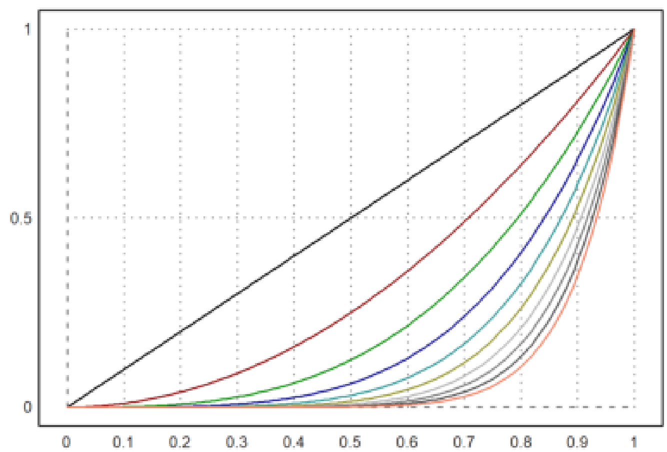
Metode kedua menggunakan pasangan matriks nilai-x dan matriks nilai-y yang berukuran sama.
Kami menghasilkan matriks nilai dengan satu Polinomial Bernstein di setiap baris. Untuk ini, kita cukup menggunakan vektor kolom i. Lihat pengantar tentang bahasa matriks untuk mempelajari lebih detail.

```
>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):
```



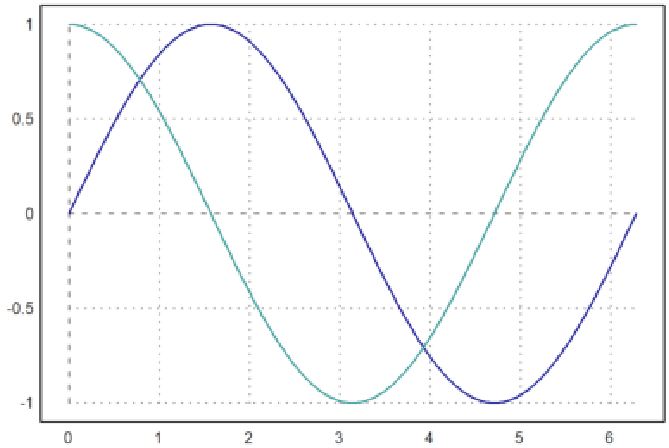
Perhatikan bahwa parameter warna dapat berupa vektor. Kemudian setiap warna digunakan untuk setiap baris matriks.

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

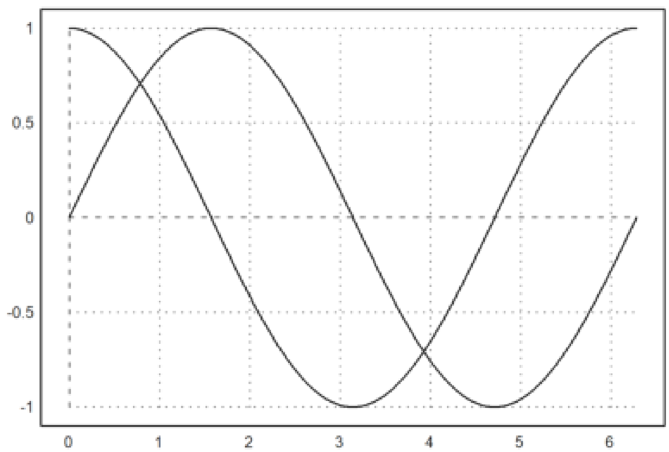


Metode lain adalah menggunakan vektor ekspresi (string). Anda kemudian dapat menggunakan larik warna, larik gaya, dan larik ketebalan dengan panjang yang sama.

```
>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
```

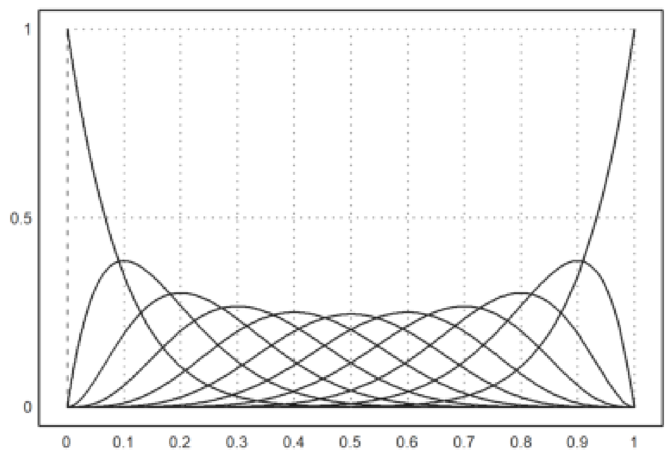
```
                 10               9                8  2                 7  3
        [(1 - x)  , 10 (1 - x)  x, 45 (1 - x)  x , 120 (1 - x)  x ,
             6  4              5  5              4  6              3  7
  210 (1 - x)  x , 252 (1 - x)  x , 210 (1 - x)  x , 120 (1 - x)  x ,
         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

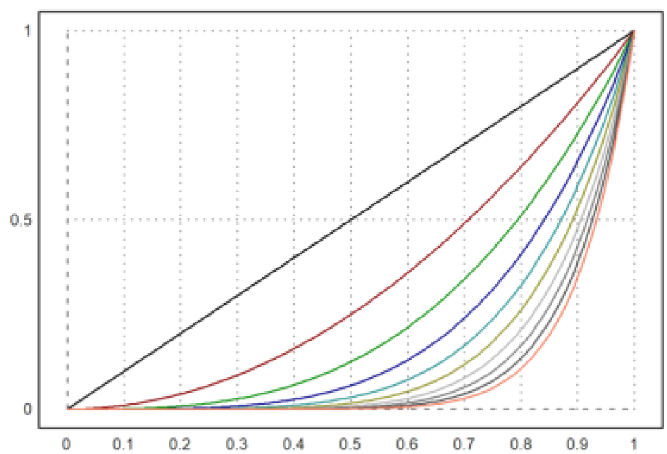
```
45*(1-x)^8*x^2
120*(1-x)^7*x^3
210*(1-x)^6*x^4
252*(1-x)^5*x^5
210*(1-x)^4*x^6
120*(1-x)^3*x^7
45*(1-x)^2*x^8
10*(1-x)*x^9
x^10
```

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



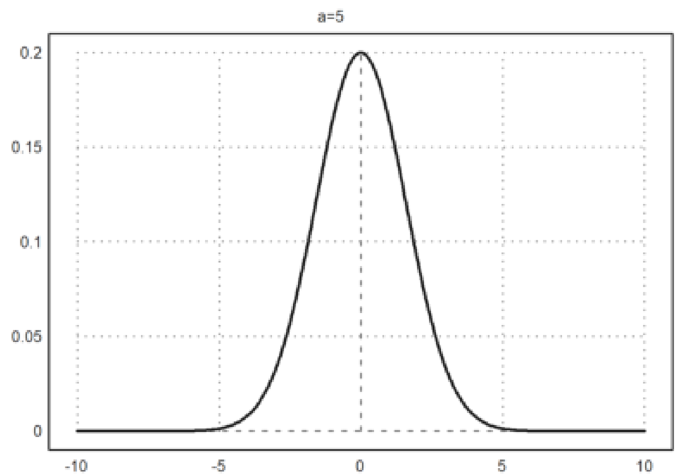
Alternatif lain adalah dengan menggunakan bahasa matriks Euler.
Jika ekspresi menghasilkan matriks fungsi, dengan satu fungsi di setiap baris, semua fungsi ini akan diplot ke dalam satu plot.
Untuk ini, gunakan vektor parameter dalam bentuk vektor kolom. Jika array warna ditambahkan, itu akan digunakan untuk setiap baris plot.

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



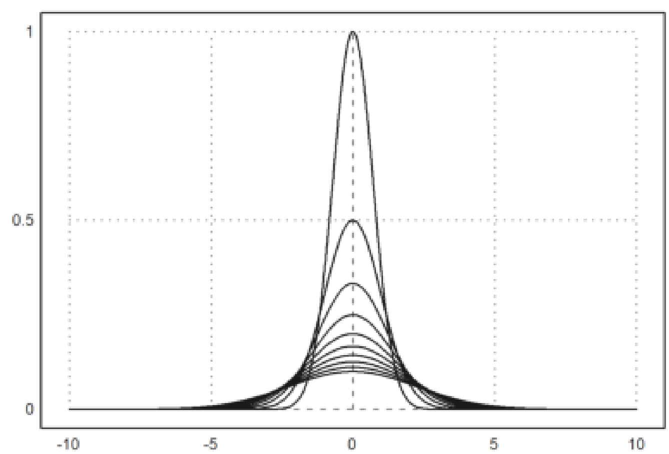
Ekspresi dan fungsi satu baris dapat melihat variabel global.
Jika Anda tidak dapat menggunakan variabel global, Anda perlu menggunakan fungsi dengan parameter tambahan, dan meneruskan parameter ini sebagai parameter titik koma.
Berhati-hatilah, untuk meletakkan semua parameter yang ditetapkan di akhir 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"):
```



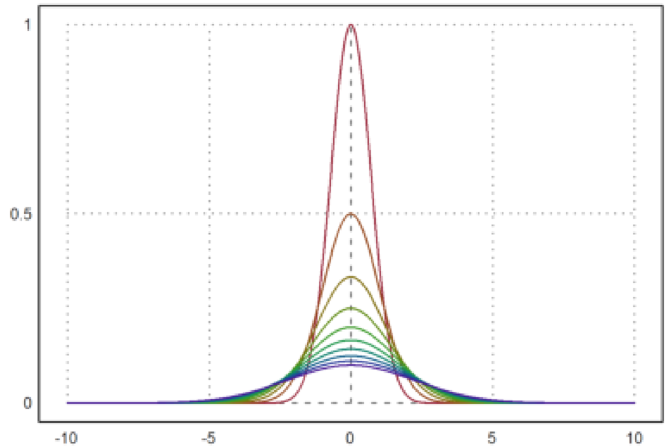
Atau, gunakan koleksi dengan nama fungsi dan semua parameter tambahan. Daftar khusus ini disebut koleksi panggilan, dan itu adalah cara yang lebih disukai untuk meneruskan argumen ke fungsi yang dengan sendirinya diteruskan sebagai argumen ke fungsi lain.
Dalam contoh berikut, kami menggunakan loop untuk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

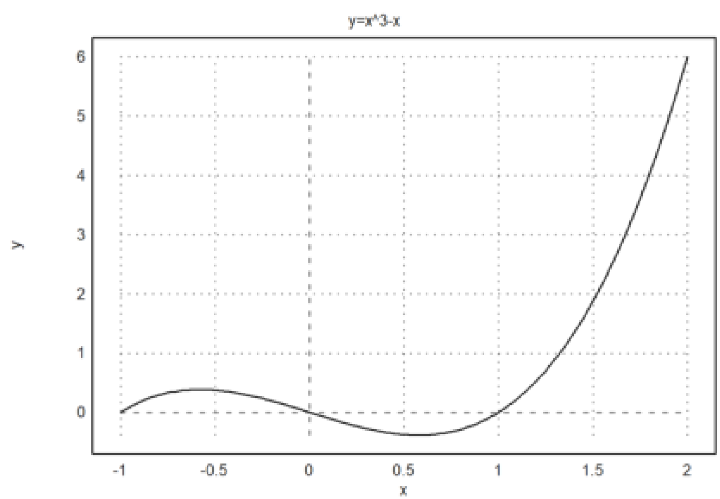
```
>x=-10:0.01:10; a=(1:10)'; plot2d(x,f(x,a),color=getspectral(a/10)):
```



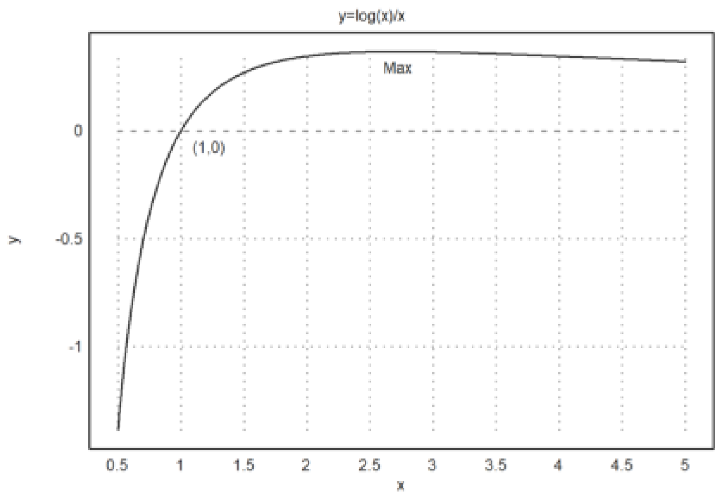
Label Teks

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"):
```

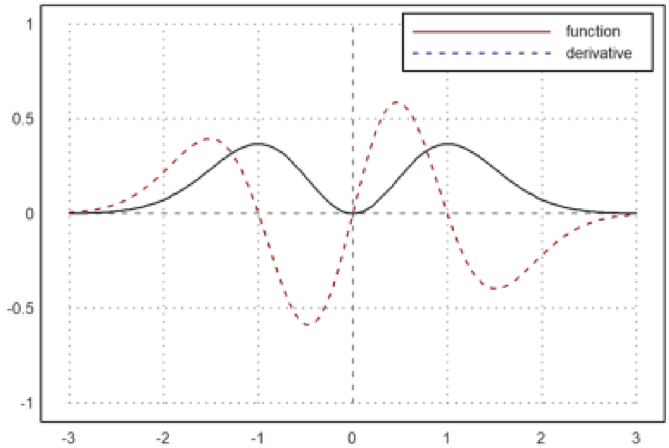


```
>expr := "log(x)/x"; ...
>  plot2d(expr,0.5,5,title="y="+expr,xl="x",yl="y"); ...
>  label("(1,0)",1,0); label("Max",E,expr(E),pos="lc"):
```



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=red,style="--"); ...
>labelbox(["function","derivative"],styles=["-","--"], ...
>   colors=[red,blue],w=0.4):
```

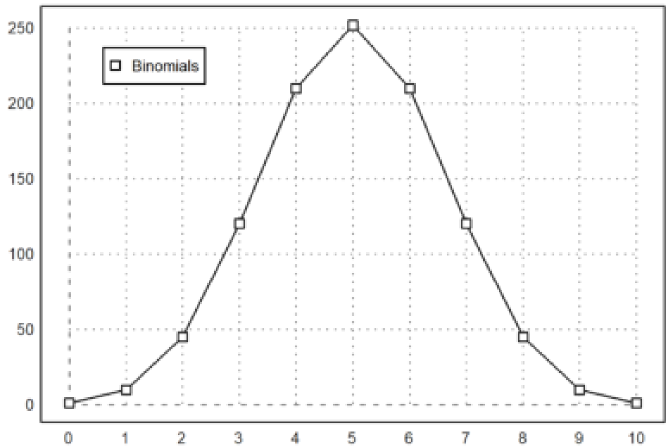


Kotak ditambatkan di kanan atas secara default, tetapi >kiri menambatkannya di kiri atas. Anda dapat memindahkannya ke tempat yang Anda suka. Posisi jangkar adalah sudut kanan atas kotak, dan angkanya adalah pecahan dari ukuran jendela grafik. Lebarnya otomatis.

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

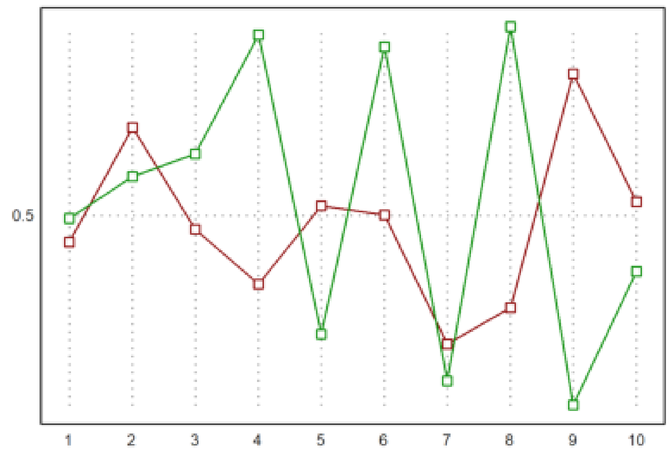
Dalam contoh berikut, hanya ada satu fungsi. Jadi kita bisa menggunakan string sebagai pengganti vektor string. Kami mengatur warna teks menjadi hitam untuk contoh ini.

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



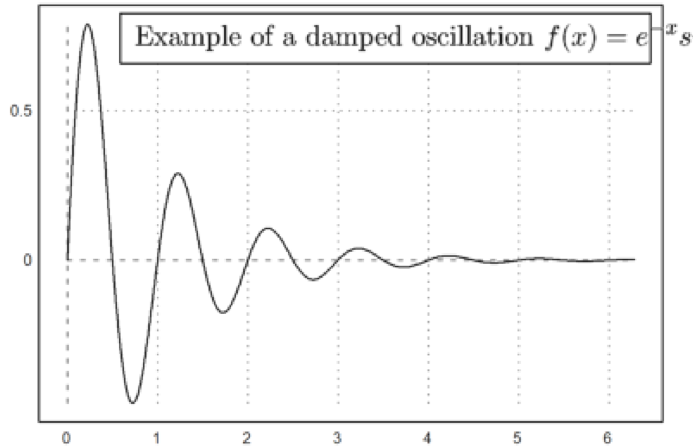
Gaya plot ini juga tersedia di statplot(). Seperti di plot2d() warna dapat diatur untuk setiap baris plot. Ada lebih banyak plot khusus untuk keperluan statistik (lihat tutorial tentang statistik).

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



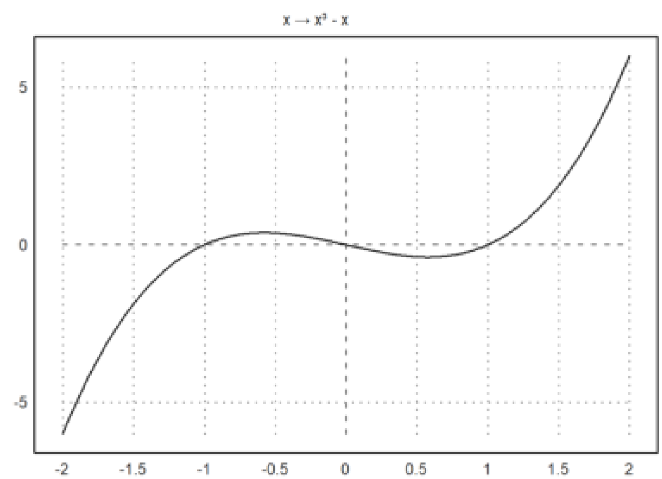
Fitur serupa adalah fungsi textbox().
Lebar secara default adalah lebar maksimal dari baris teks. Tapi itu bisa diatur oleh pengguna juga.

```
>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):
```



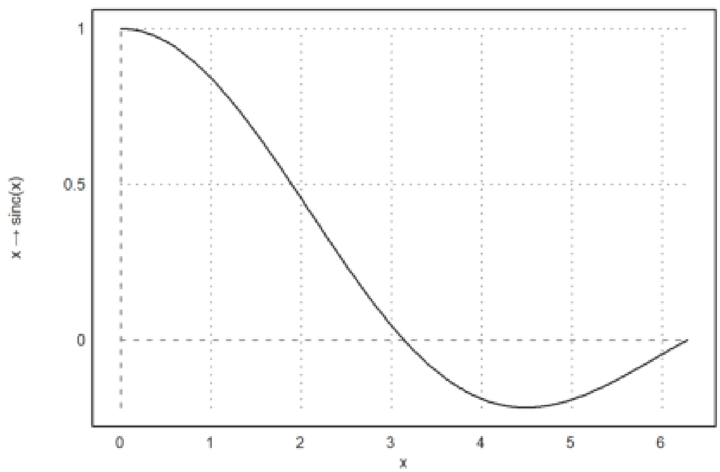
Label teks, judul, kotak label, dan teks lainnya dapat berisi string Unicode (lihat sintaks EMT untuk mengetahui lebih lanjut tentang string Unicode).

```
>plot2d("x^3-x",title=u"x &rarr; x&sup3; - x"):
```



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

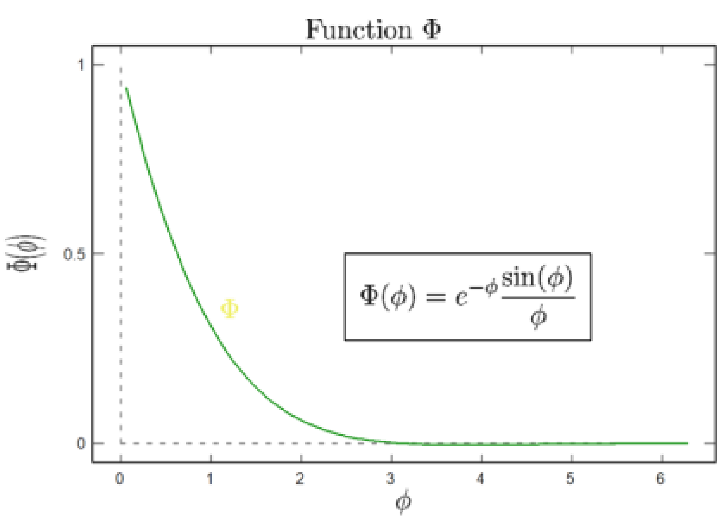
```
>plot2d("sinc(x)",0,2pi,xl="x",yl=u"x &rarr; sinc(x)",>vertical):
```



LaTeX

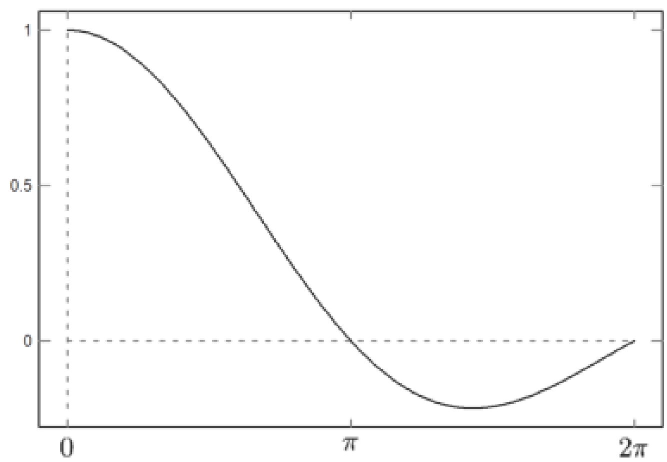
Anda juga dapat memplot rumus LaTeX jika Anda telah menginstal sistem LaTeX. Saya merekomendasikan MiKTeX. Jalur ke biner "lateks" dan "dvipng" harus berada di jalur sistem, atau Anda harus mengatur LaTeX di menu opsi.
Perhatikan, bahwa penguraian LaTeX lambat. Jika Anda ingin menggunakan LaTeX dalam 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=green, ...
>  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=yellow),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);  ...
>xtick([0,pi,2pi],["0","\pi","2\pi"],>latex):
```



Tentu saja, fungsi juga dapat digunakan.

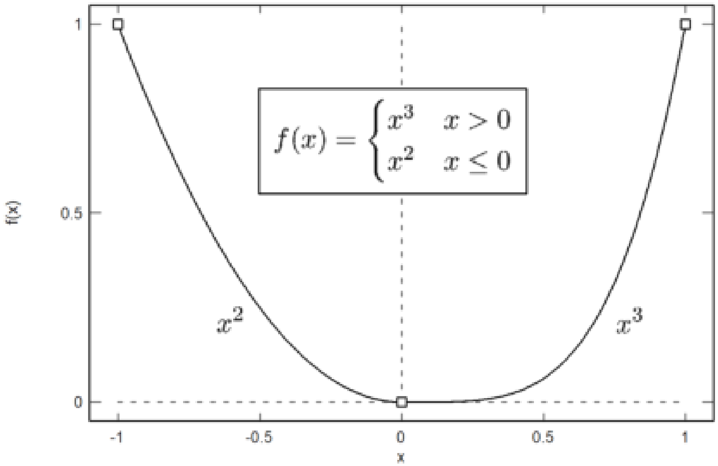
```
>function map f(x) ...
```

```
  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 \le 0\end{cases}"), ...
>  x=0.7,y=0.2):
```

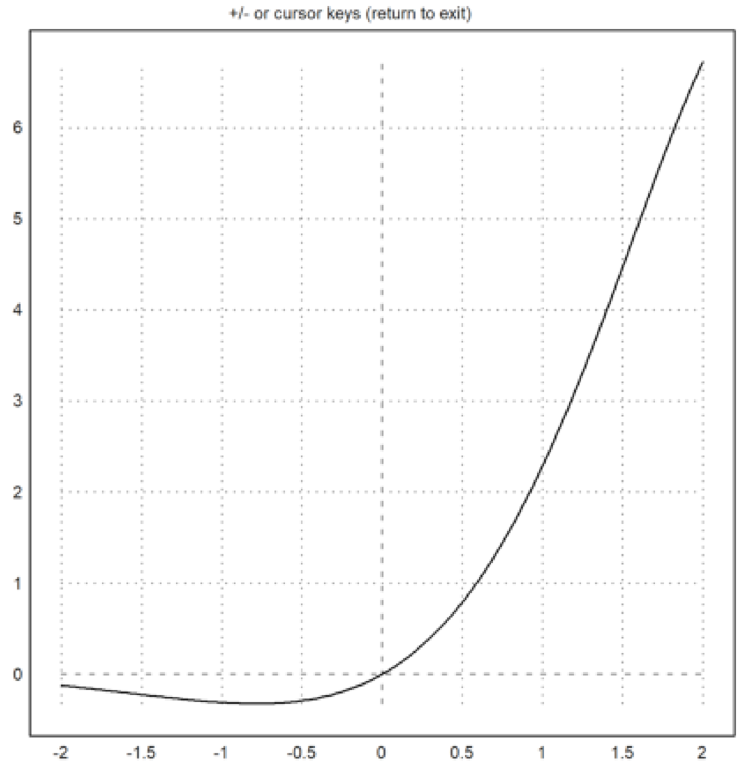


Interaksi pengguna

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 dragvalues() 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):
```

```
Variable plotf not found!
Use global variables or parameters for string evaluation.
Error in expression: plotf
Try "trace errors" to inspect local variables after errors.
dragvalues:
    f$(vv,args());
```

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;
```

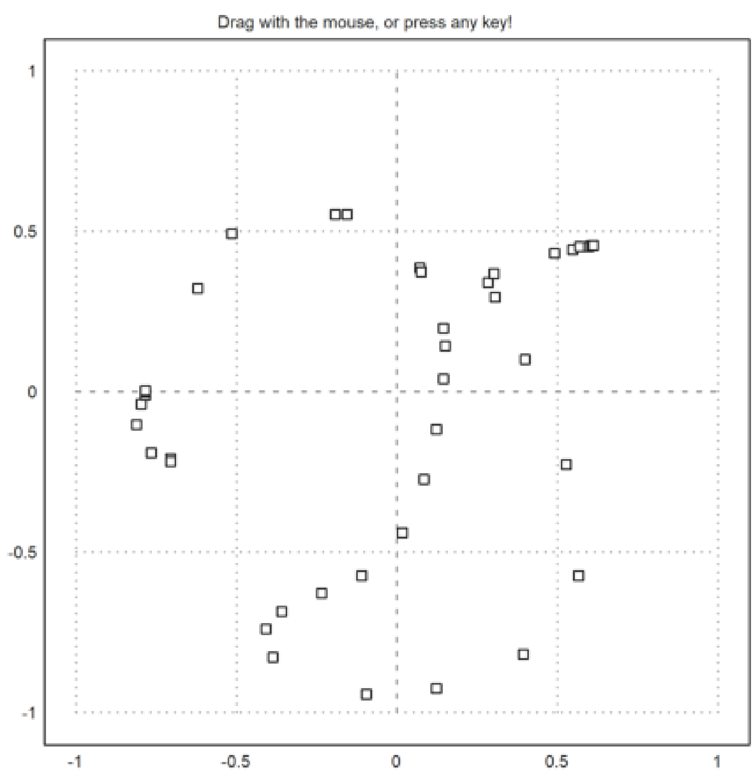
```
    endif;
  end
endfunction
```

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

Gaya Plot 2D

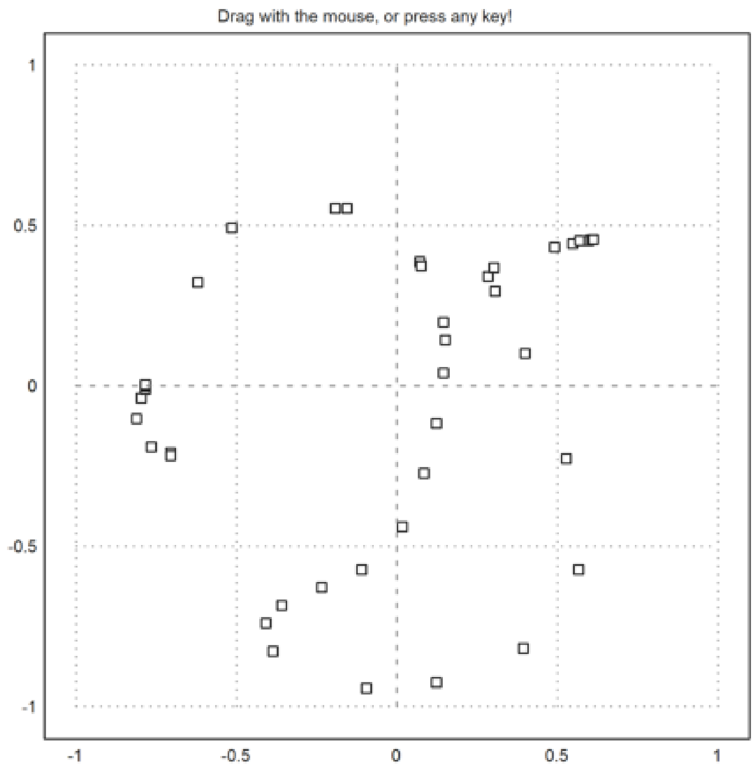
Secara default, EMT menghitung tick sumbu otomatis dan menambahkan label ke setiap tick. Ini dapat diubah dengan parameter grid. Gaya default sumbu dan label dapat dimodifikasi. Selain itu, label dan judul dapat ditambahkan secara manual. Untuk mengatur ulang ke gaya default, gunakan reset().

```
>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):
```



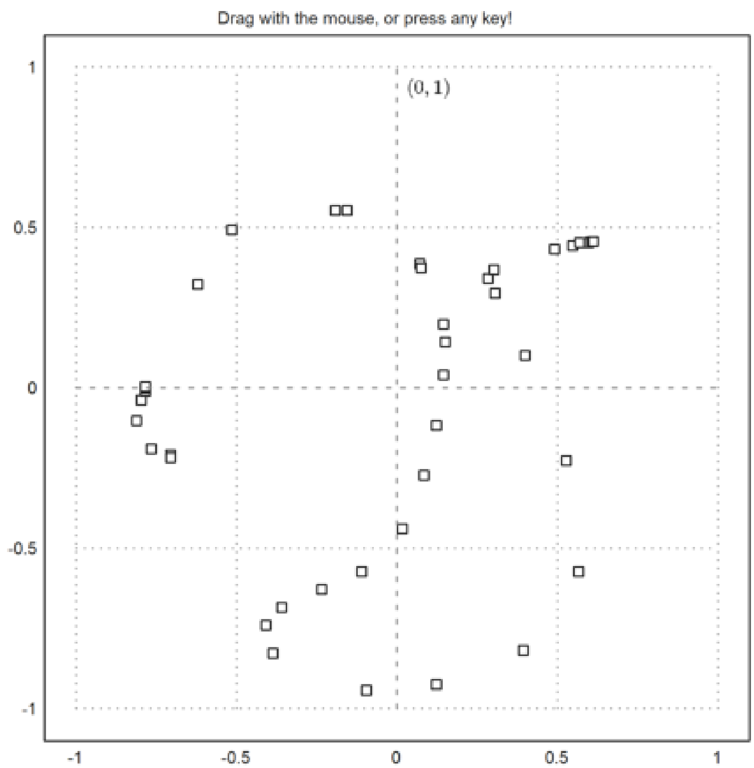
Parameter <frame mematikan frame, dan framecolor=blue mengatur frame ke warna biru.
Jika Anda ingin centang sendiri, Anda dapat menggunakan style=0, dan menambahkan semuanya nanti.

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



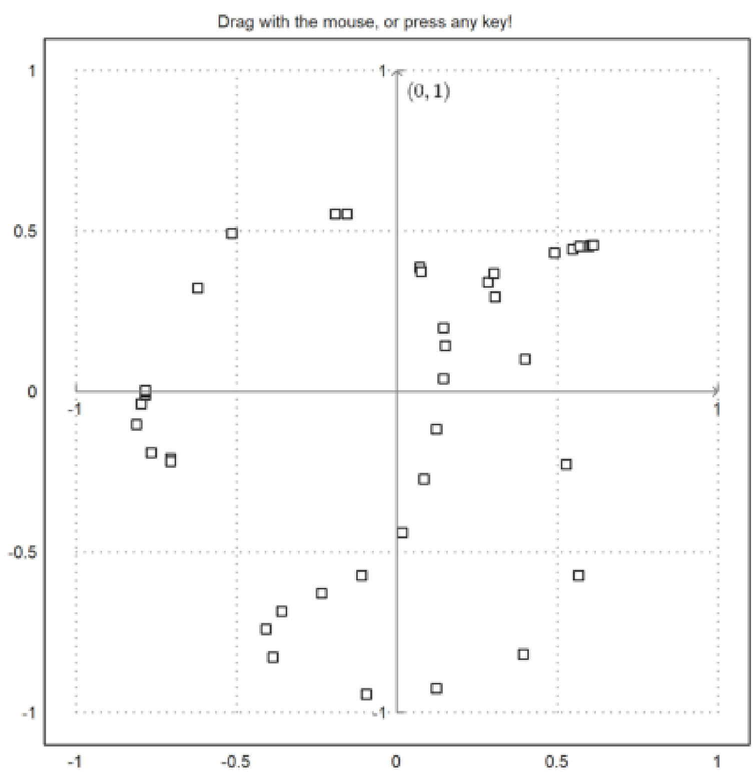
Untuk judul plot dan label sumbu, lihat contoh berikut.

```
>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
```

```
                                  3
                                 x  - x
```

```
>plot2d(f,-1,1,>square);
```

```
Variable or function f not found.
Error in:
plot2d(f,-1,1,>square); ...
        ^
```

```
>x0=fmin(f,0,1); // compute point of minimum
```

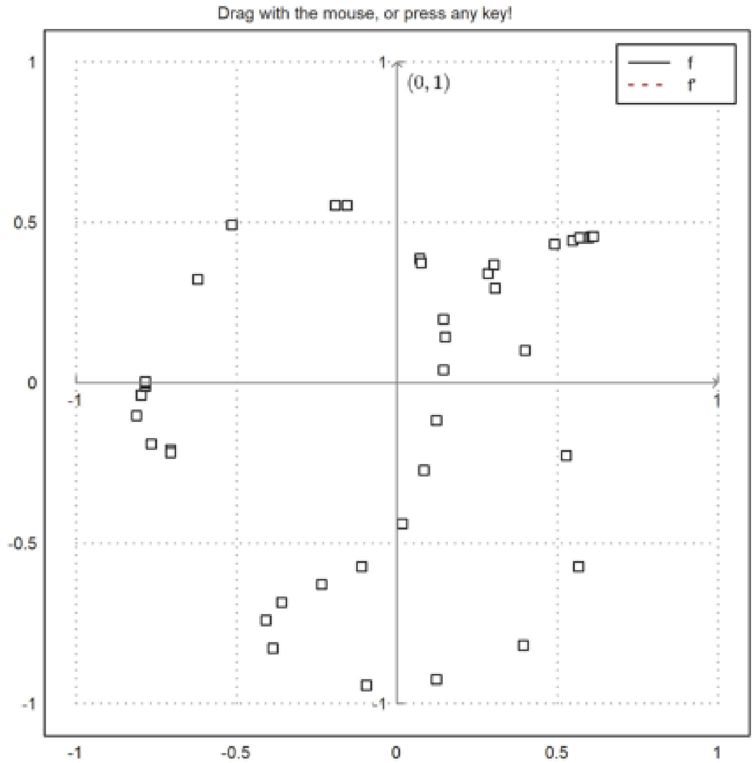
```
Variable or function f not found.
Error in:
x0=fmin(f,0,1); // compute point of minimum ...
         ^
```

```
>label("Rel. Min.",x0,f(x0),pos="lc"): // add a label there
```

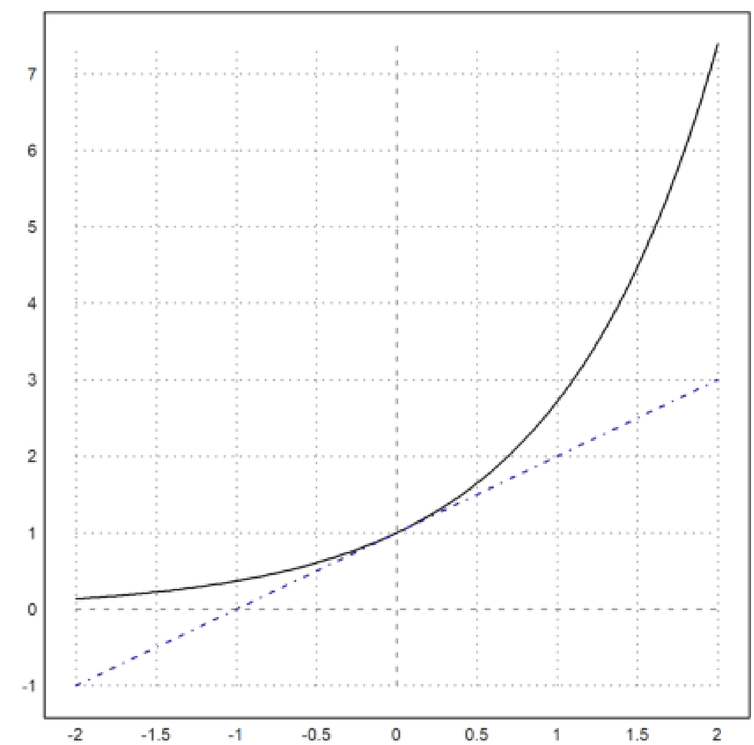
```
Variable or function x0 not found.
Error in:
label("Rel. Min.",x0f(x0),pos="lc"): // add a label there ...
                        ^
```

Ada juga kotak teks.

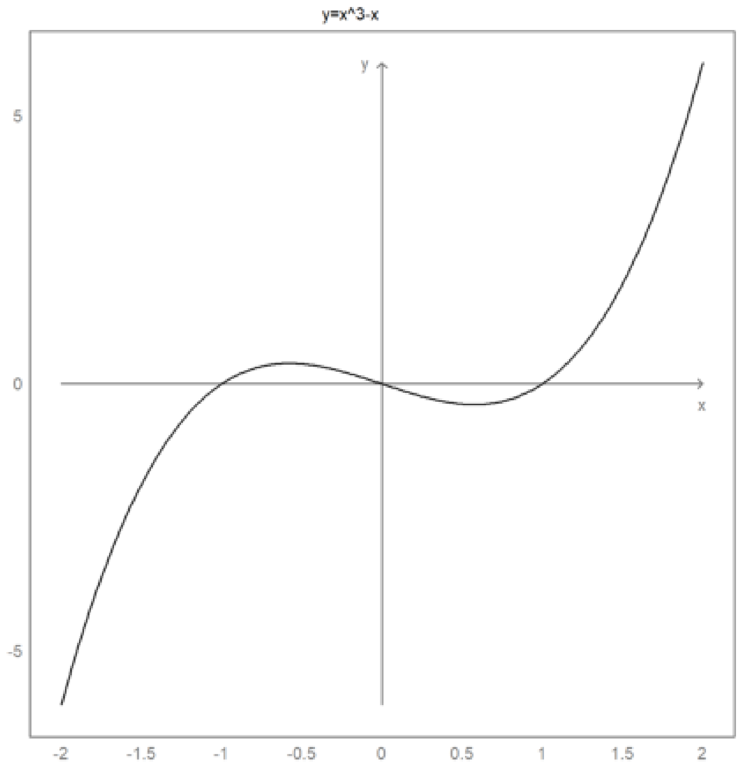
```
>plot2d(&f(x),-1,1,-2,2); // function
>plot2d(&diff(f(x),x),>add,style="--",color=red); // derivative
>labelbox(["f","f'"],["-","--"],[black,red]): // label box
```




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

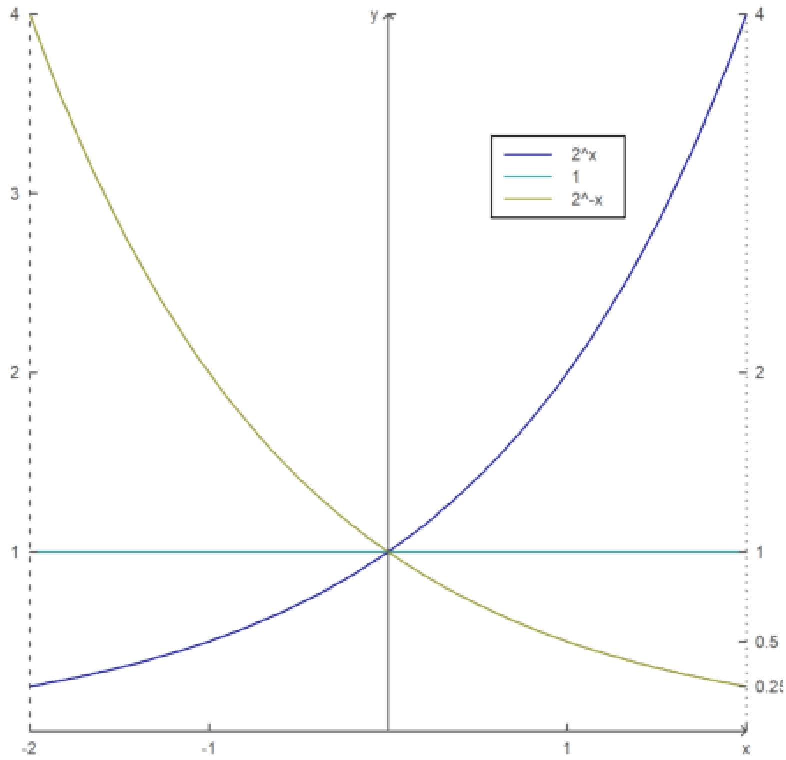


```
>gridstyle("->",color=gray,textcolor=gray,framecolor=gray);  ...
> plot2d("x^3-x",grid=1);   ...
> settitle("y=x^3-x",color=black); ...
> label("x",2,0,pos="bc",color=gray);  ...
> label("y",0,6,pos="cl",color=gray); ...
> reset():
```



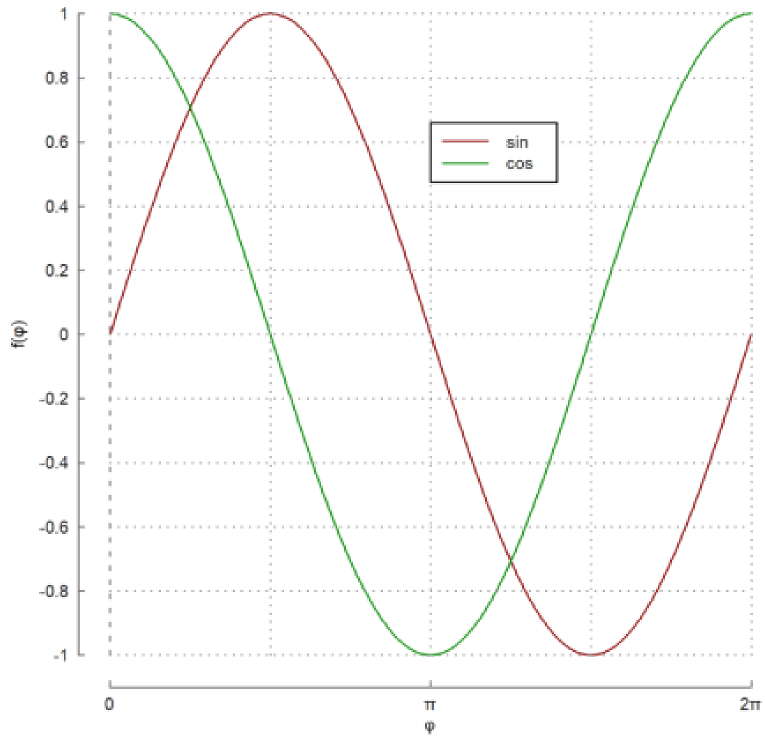
Untuk kontrol lebih, sumbu x dan sumbu y dapat dilakukan secara manual.
Perintah fullwindow() memperluas jendela plot karena kita tidak lagi membutuhkan tempat untuk label di luar jendela plot. Gunakan shrinkwindow() atau reset() untuk mengatur ulang ke default.

```
>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.

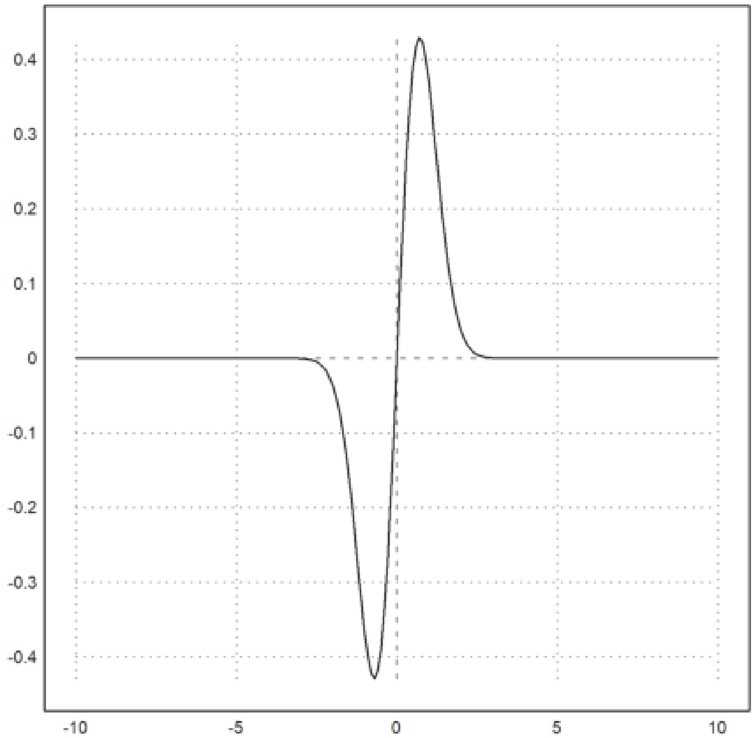
```
>aspect(1.5);
>plot2d(["sin(x)","cos(x)"],0,2pi,color=[red,green],<grid,<frame); ...
> xaxis(-1.1,(0:2)*pi,xt=["0",u"&pi;",u"2&pi;"],style="-",>ticks,>zero);  ...
> xgrid((0:0.5:2)*pi,<ticks); ...
> yaxis(-0.1*pi,-1:0.2:1,style="-",>zero,>grid); ...
> labelbox(["sin","cos"],colors=[red,green],x=0.5,y=0.2,>left); ...
> xlabel(u"&phi;"); ylabel(u"f(&phi;)"):
```



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.

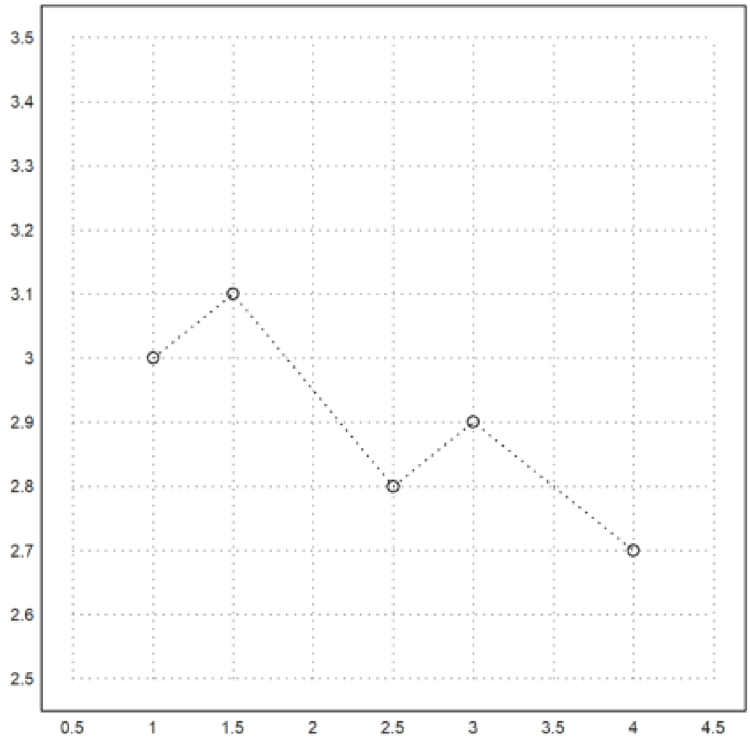
```
>x=-10:0.1:10; y=exp(-x^2)*x; plot2d(x,y):
```



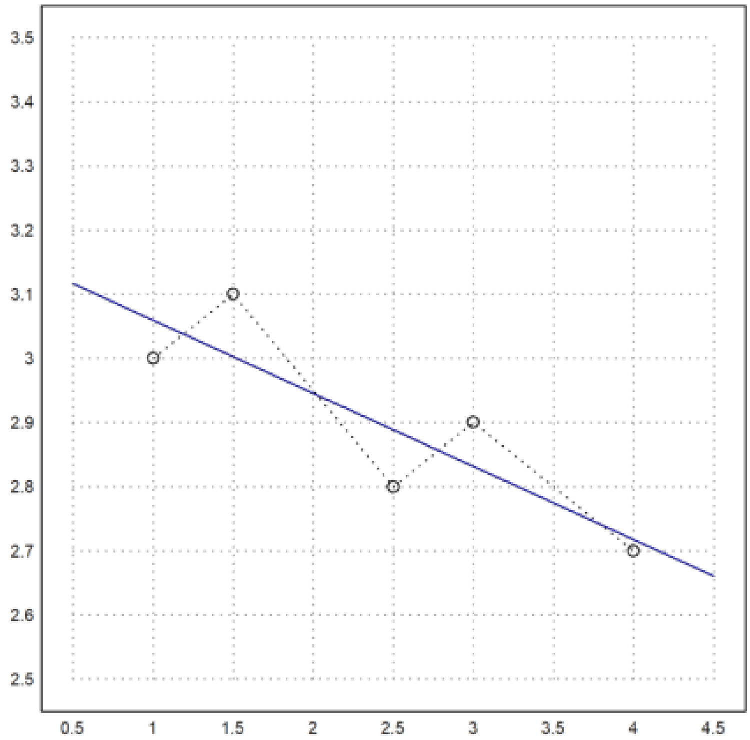
Data juga dapat diplot sebagai titik. Gunakan poin=true untuk ini. Plotnya bekerja seperti poligon, tetapi hanya menggambar sudut-sudutnya.
- style="...": Pilih dari "[]", "<>", "o", ".", "..", "+", "*", "[]", "< >", "o", "..", "", "|".

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
```



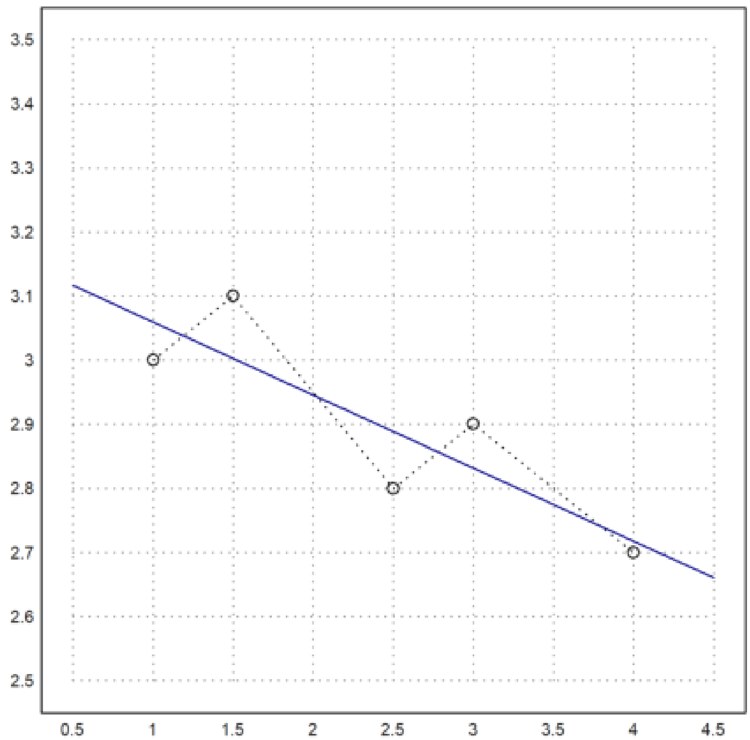

```
>p=polyfit(xdata,ydata,1); // get regression line
>plot2d("polyval(p,x)",>add,color=blue): // add plot of line
```



Menggambar Daerah Yang Dibatasi Kurva

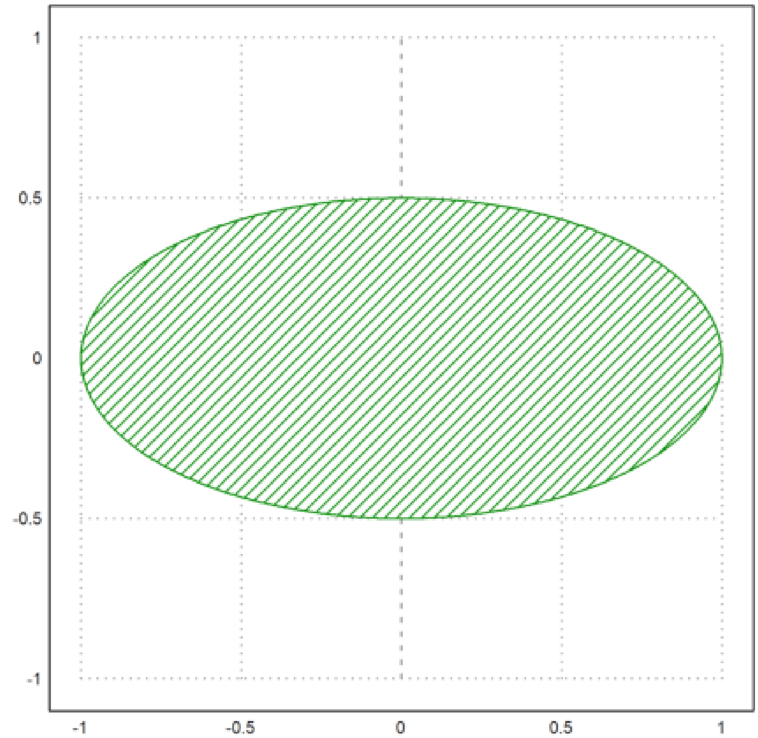
Plot data benar-benar poligon. Kita juga dapat memplot kurva atau kurva terisi.
- terisi=benar mengisi plot.
- style="...": Pilih dari "", "/", "\", "\/".
- fillcolor: Lihat di atas untuk warna yang tersedia.
Warna isian ditentukan oleh argumen "fillcolor", dan pada <outline opsional mencegah menggambar 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):
```

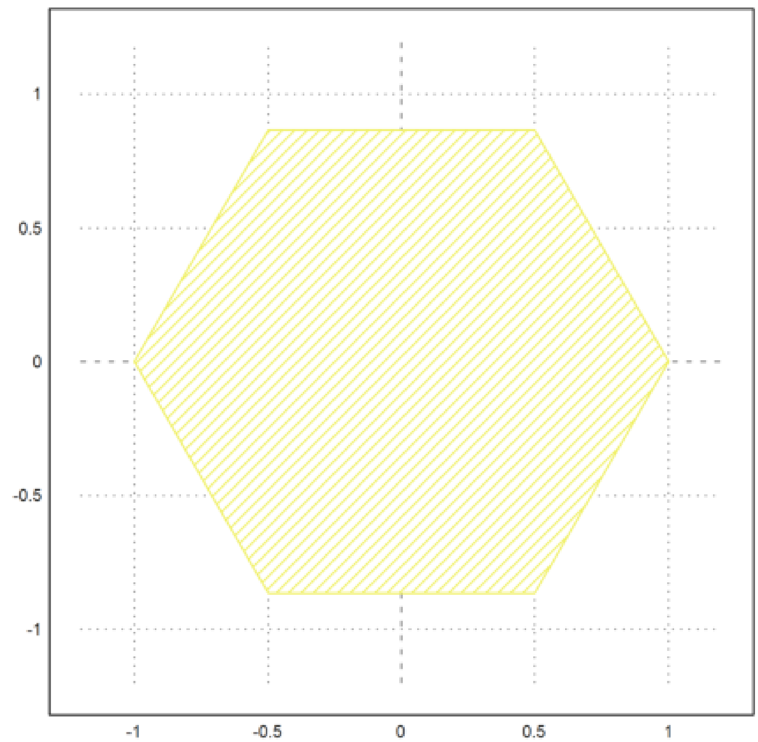


Dalam contoh berikut kami memplot elips terisi dan dua segi enam terisi menggunakan kurva tertutup dengan 6 titik dengan gaya isian berbeda.

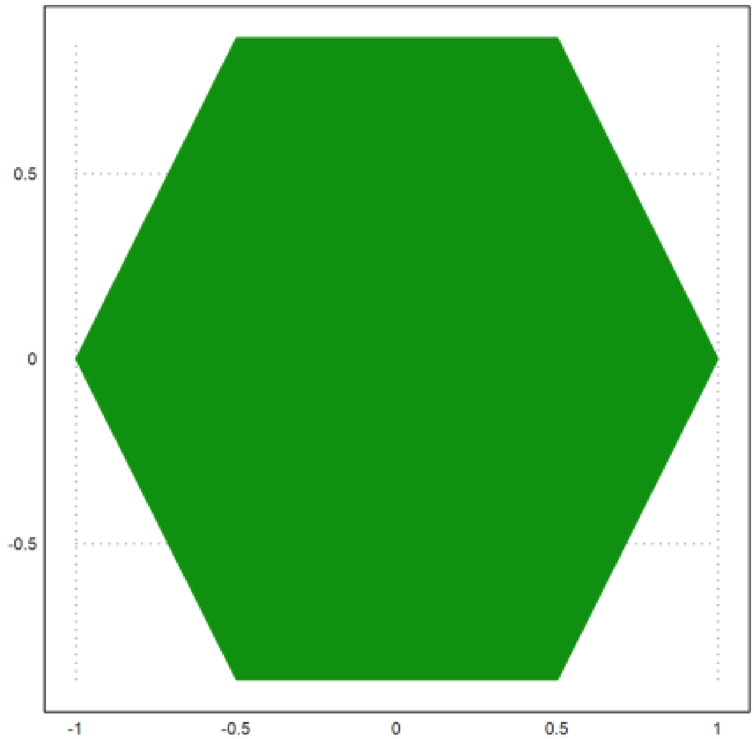
```
>x=linspace(0,2pi,1000); plot2d(sin(x),cos(x)*0.5,r=1,>filled,style="/"):
```




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

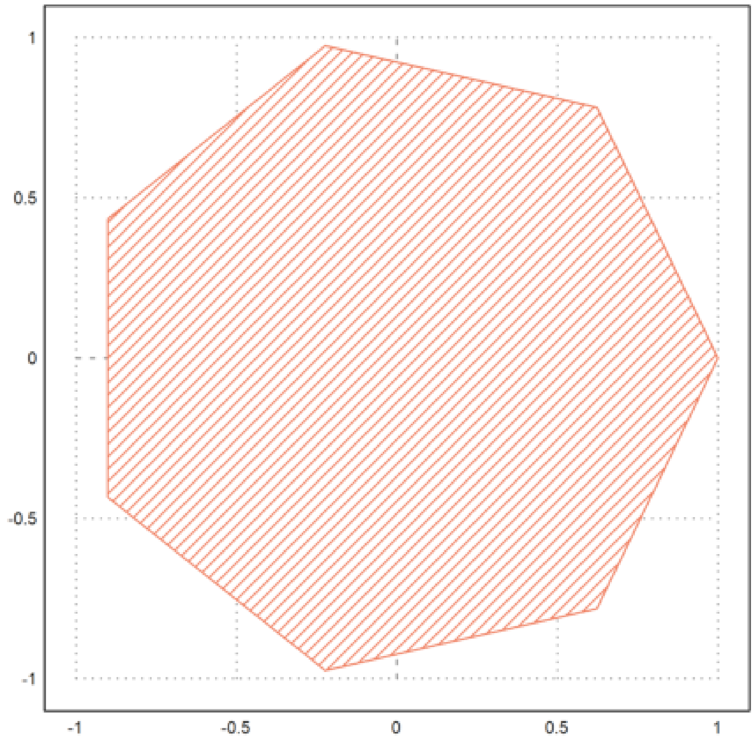


```
>t=linspace(0,2pi,6); plot2d(cos(t),sin(t),>filled,style="#"):
```



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=orange):
```



Berikut ini adalah himpunan nilai maksimal dari empat kondisi linier yang kurang dari atau sama dengan 3. Ini adalah A[k].v<=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):
```

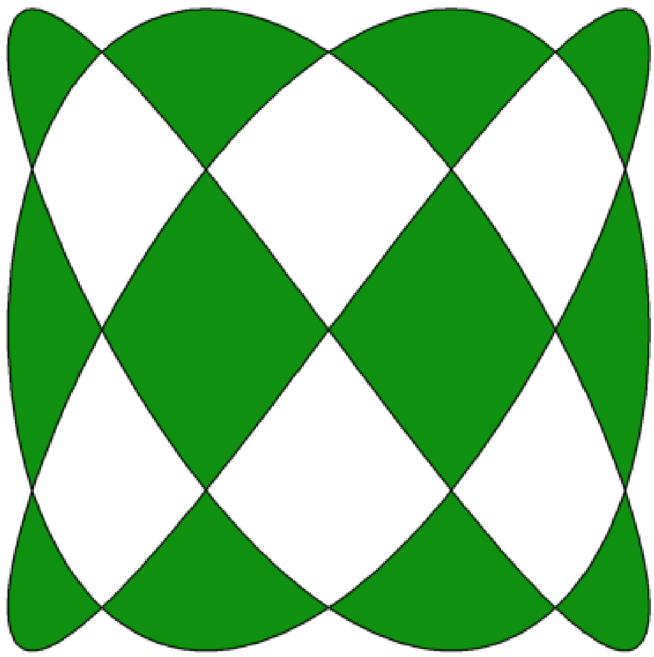
```
Variable f not found!
Use global variables or parameters for string evaluation.
Error in expression: f
 %ploteval2:
    if maps then return %mapexpression2(x,y,f$;args());
fcontour:
    Z=%ploteval2(f$,X,Y,maps;args());
Try "trace errors" to inspect local variables after errors.
plot2d:
    =style,=outline,=frame);
```

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);
```

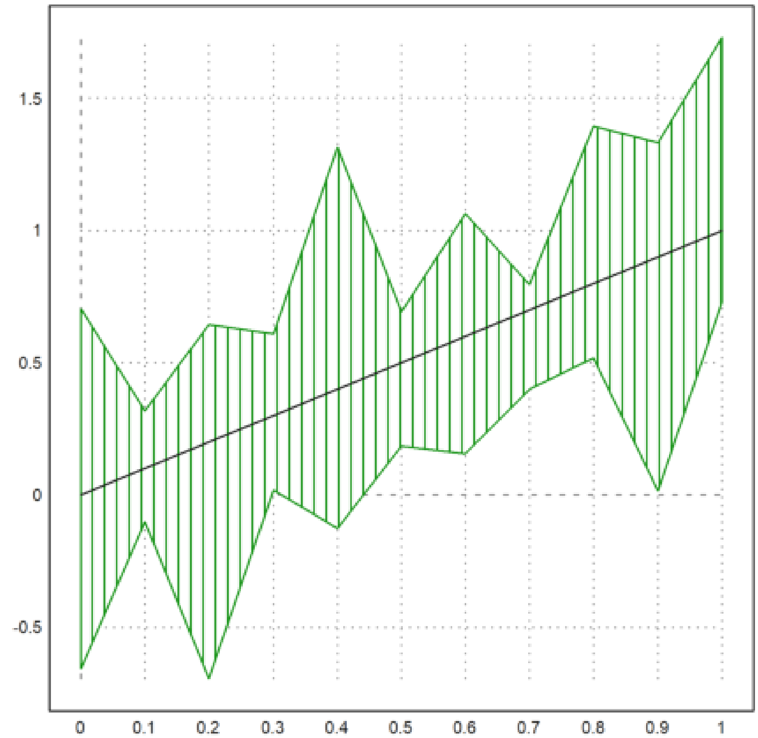
Kami sekarang memiliki vektor x dan y nilai. plot2d() dapat memplot nilai-nilai ini sebagai kurva yang menghubungkan titik-titik. Plotnya bisa diisi. Pada kasus ini ini menghasilkan hasil yang bagus karena aturan lilitan, yang digunakan untuk isi.

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



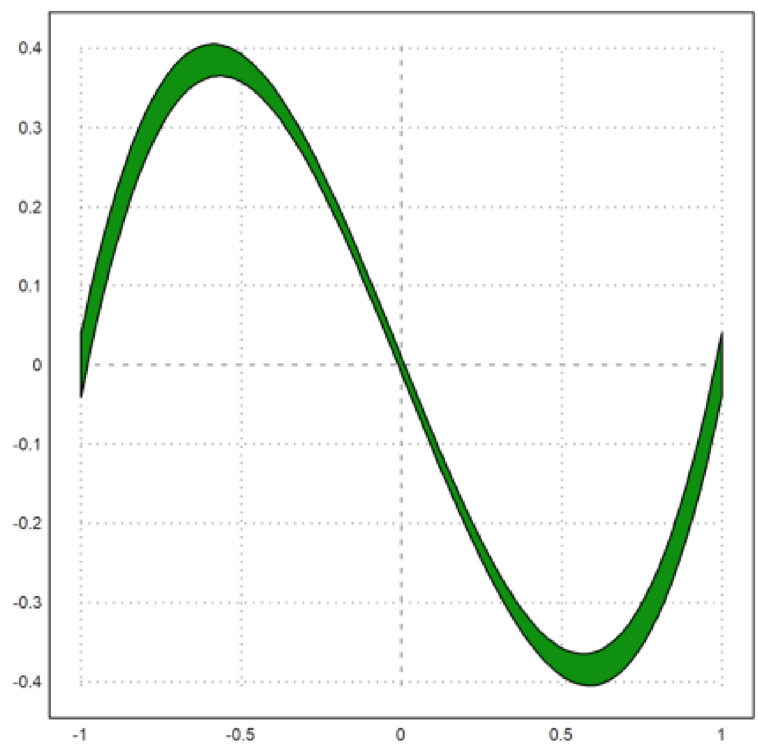
Sebuah vektor interval diplot terhadap nilai x sebagai daerah terisi
antara nilai interval bawah dan atas.
Hal ini dapat berguna untuk memplot kesalahan perhitungan. Tapi itu bisa
juga digunakan untuk memplot kesalahan statistik.

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



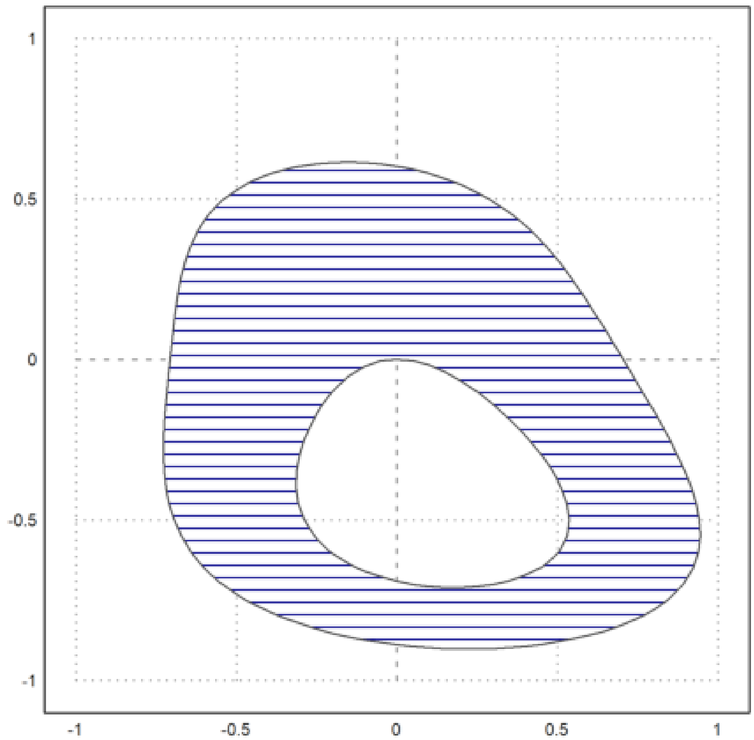
Jika x adalah vektor yang diurutkan, dan y adalah vektor interval, maka plot2d akan memplot rentang interval yang terisi dalam bidang. Gaya isian sama dengan gaya poligon.

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



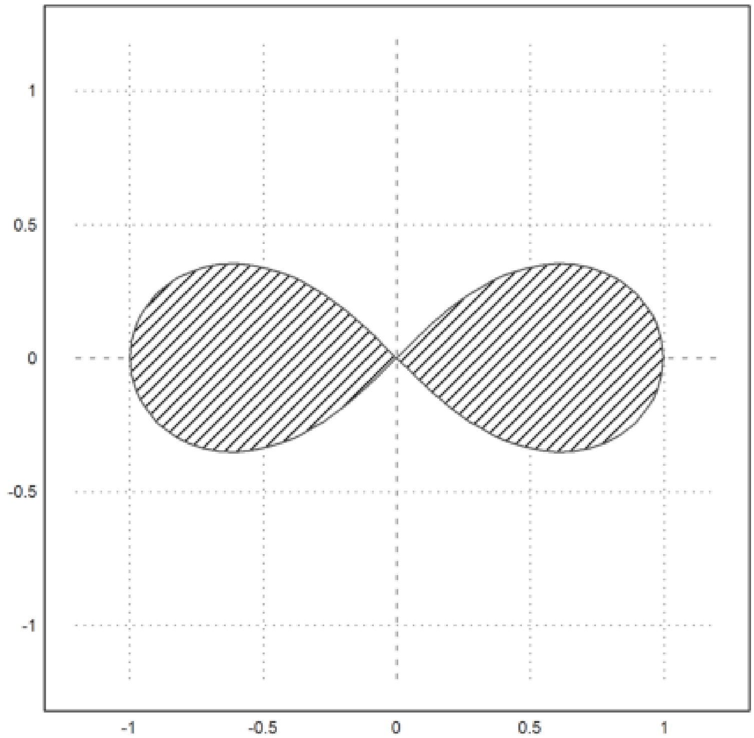
Jika x adalah vektor yang diurutkan, dan y adalah vektor interval, maka plot2d akan memplot rentang interval yang terisi dalam bidang. Gaya isian sama dengan gaya poligon.

```
>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
```

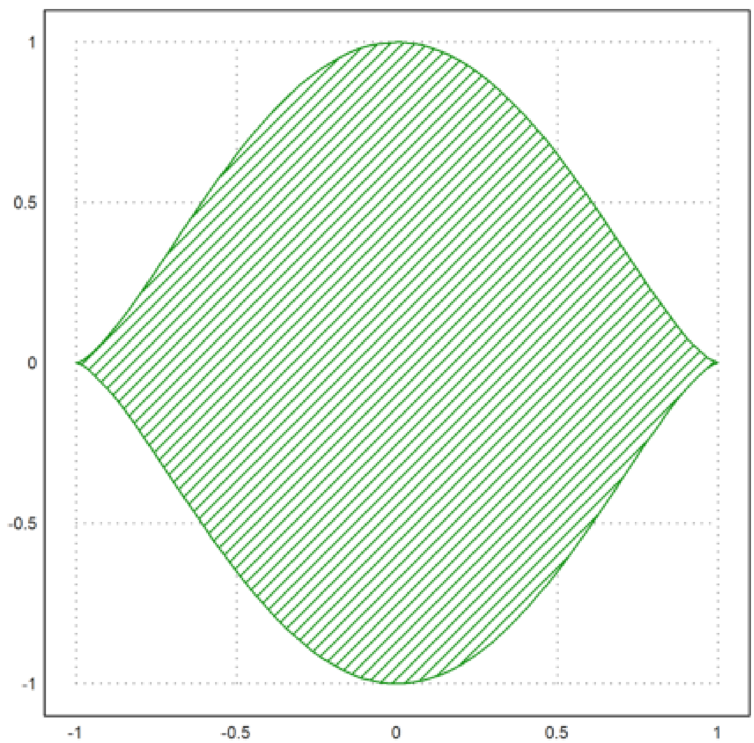


Kami juga dapat mengisi rentang nilai seperti
lateks: -1 \le (x^2+y^2)^2-x^2+y^2 \le 0.

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



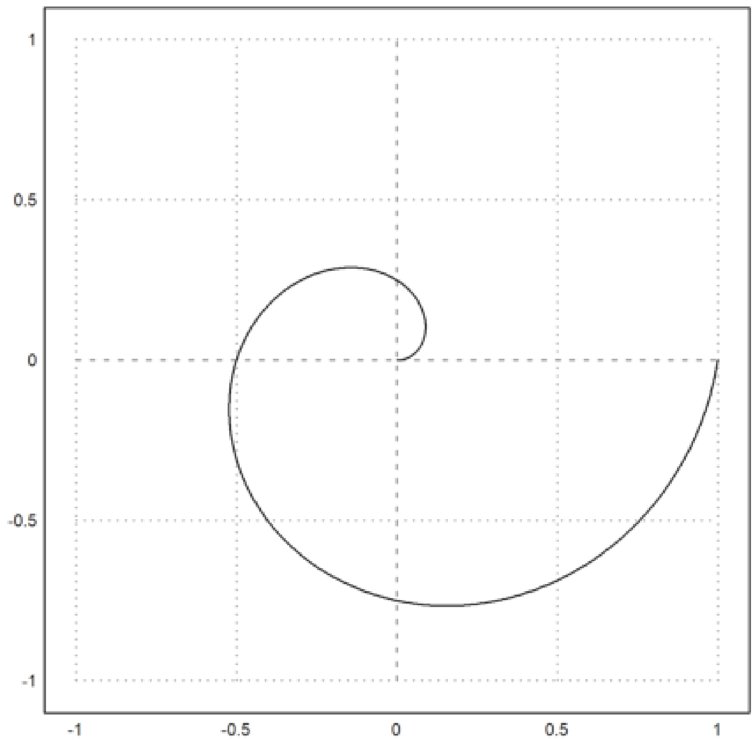

```
>plot2d("cos(x)","sin(x)^3",xmin=0,xmax=2pi,>filled,style="/"):
```



Grafik Fungsi Parametrik

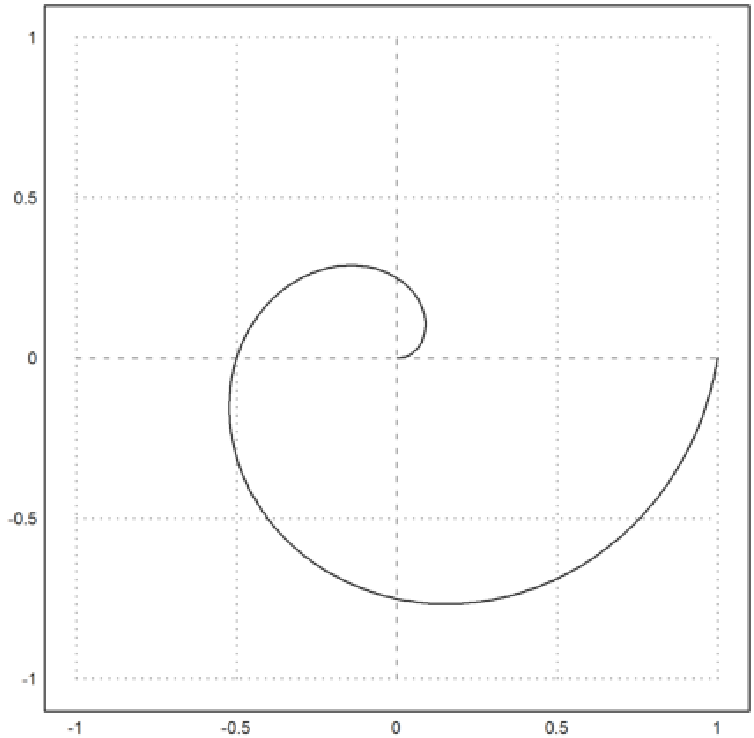
Nilai-x tidak perlu diurutkan. (x,y) hanya menggambarkan kurva. Jika x diurutkan, kurva tersebut merupakan grafik fungsi.
Dalam contoh berikut, kami memplot spiral
lateks: \gamma(t) = t \cdot (\cos(2\pi t),\sin(2\pi t))
Kita perlu menggunakan banyak titik untuk tampilan yang halus atau fungsi adaptif() 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):
```

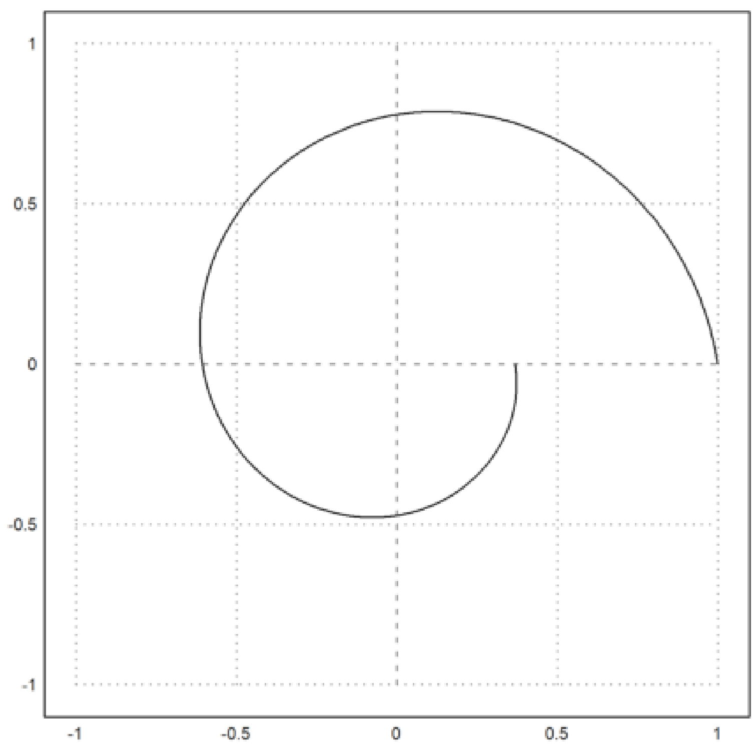


Atau, dimungkinkan untuk menggunakan dua ekspresi untuk kurva. Berikut ini plot kurva yang sama seperti di atas.

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

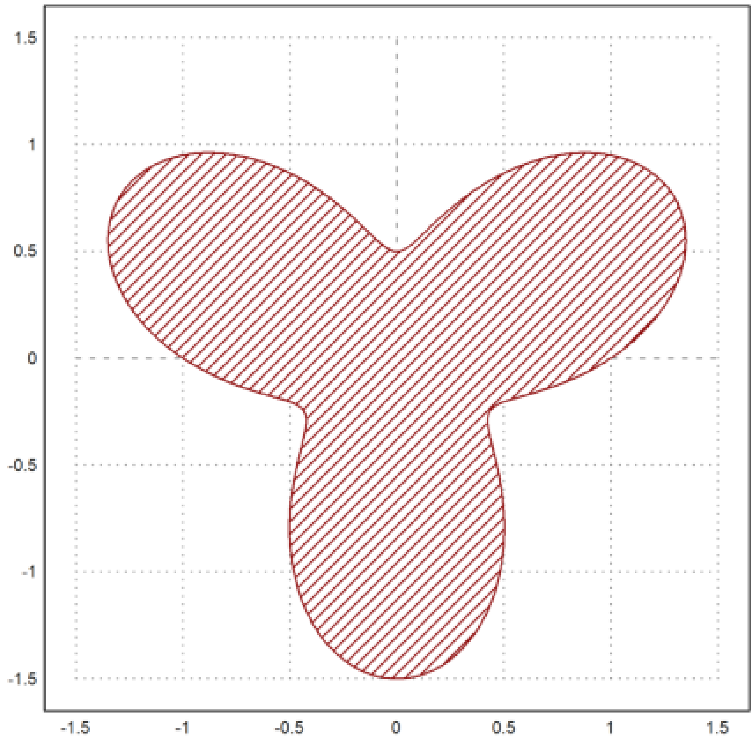



```
>t=linspace(0,1,1000); r=exp(-t); x=r*cos(2pi*t); y=r*sin(2pi*t);
>plot2d(x,y,r=1):
```



Dalam contoh berikutnya, kami memplot kurva
lateks: \gamma(t) = (r(t) \cos(t), r(t) \sin(t))
dengan
lateks: r(t) = 1 + \dfrac{\sin(3t)}{2}.

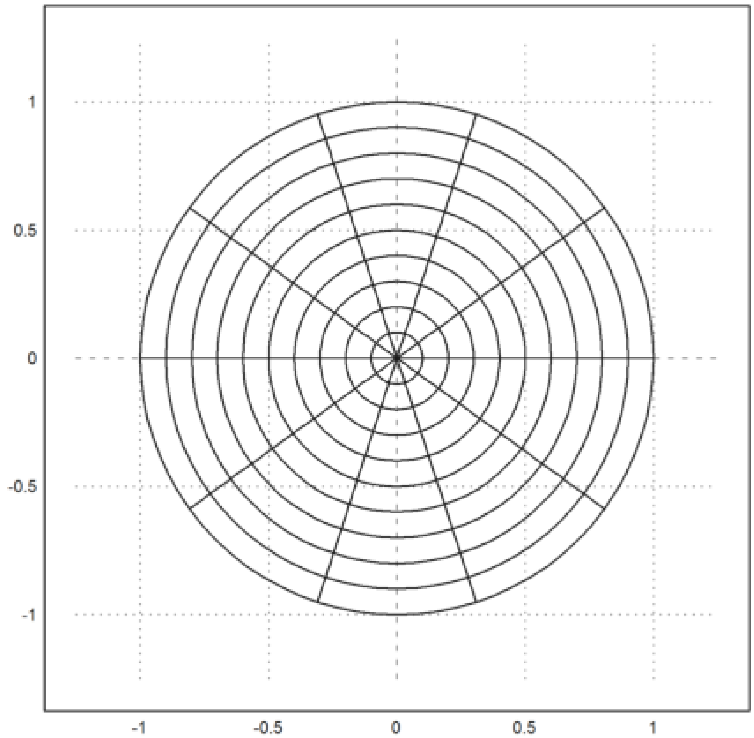
```
>t=linspace(0,2pi,1000); r=1+sin(3*t)/2; x=r*cos(t); y=r*sin(t); ...
>plot2d(x,y,>filled,fillcolor=red,style="/",r=1.5):
```



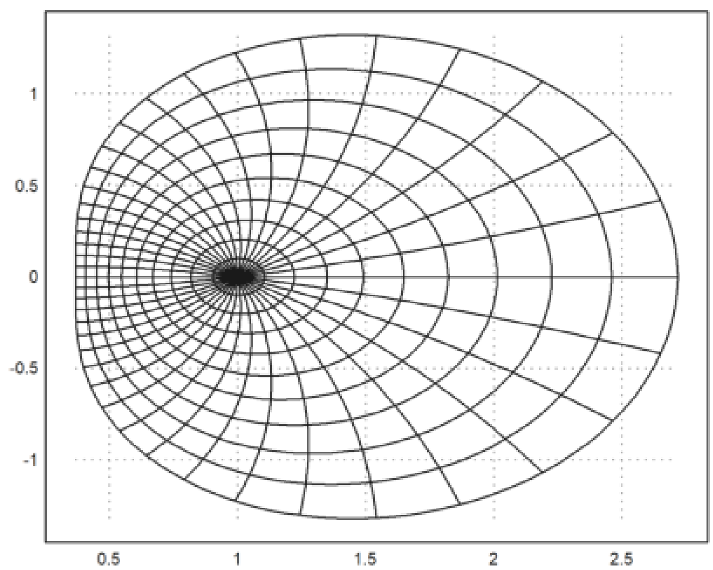
Menggambar Grafik Bilangan Kompleks

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.

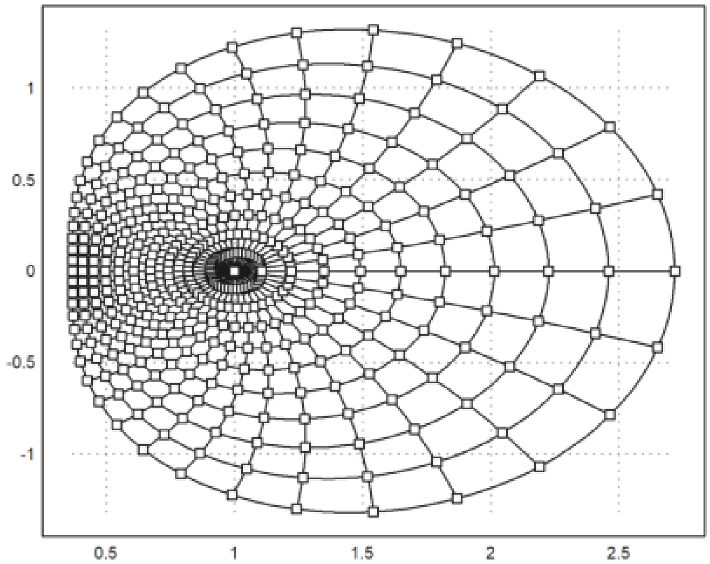
```
>aspect(); r=linspace(0,1,50); a=linspace(0,2pi,80)'; z=r*exp(I*a);...
>plot2d(z,a=-1.25,b=1.25,c=-1.25,d=1.25,cgrid=10):
```




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

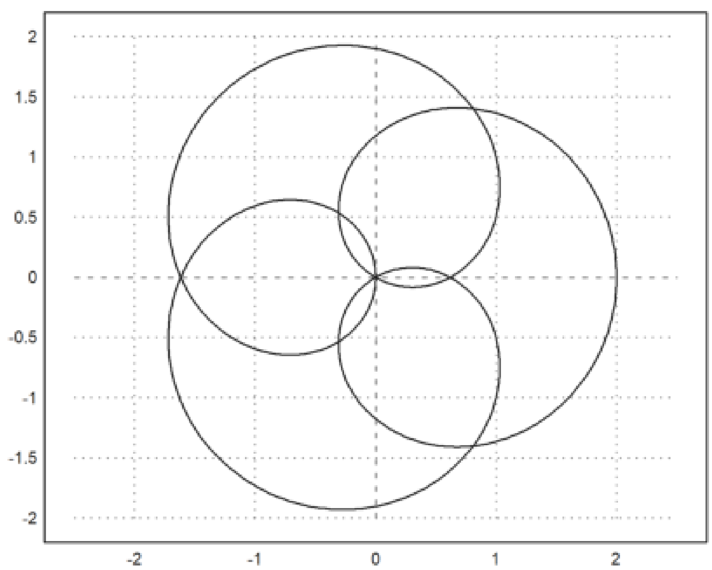



```
>r=linspace(0,1,10); a=linspace(0,2pi,40)'; z=r*exp(I*a);
>plot2d(exp(z),>points,>add):
```



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
lateks: \gamma(t) = e^{it}

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



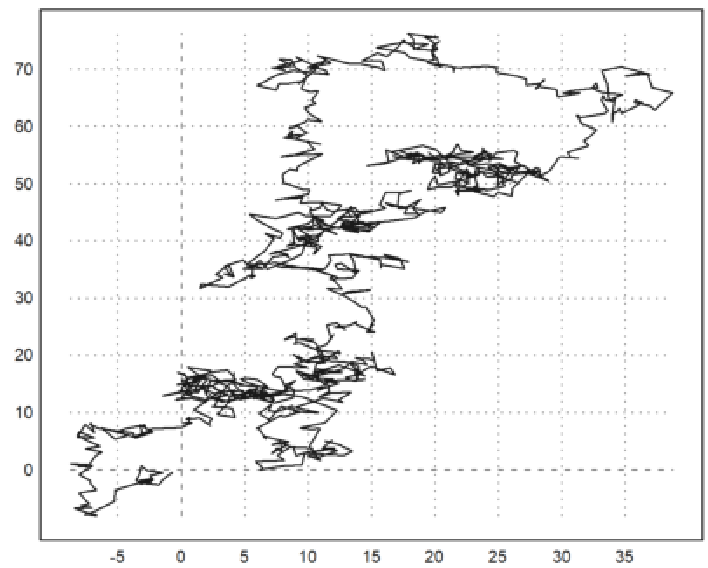
Plot Statistik

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.

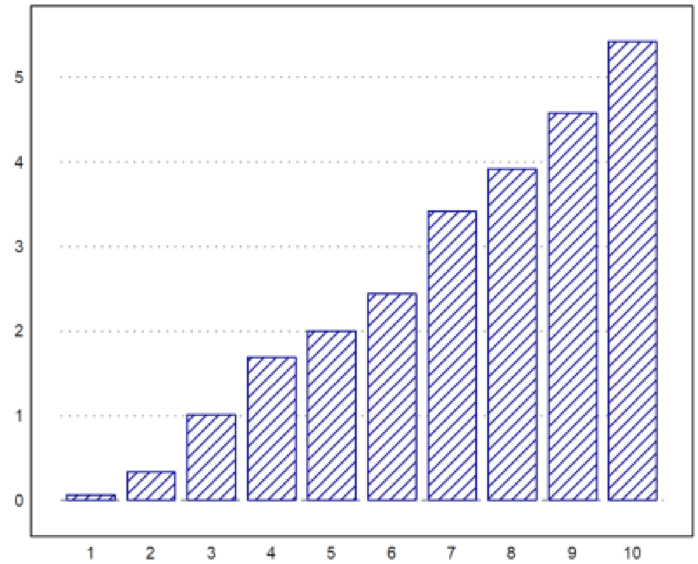
```
>plot2d(cumsum(randnormal(1,1000))):
```

Menggunakan dua baris menunjukkan jalan dalam dua dimensi.

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

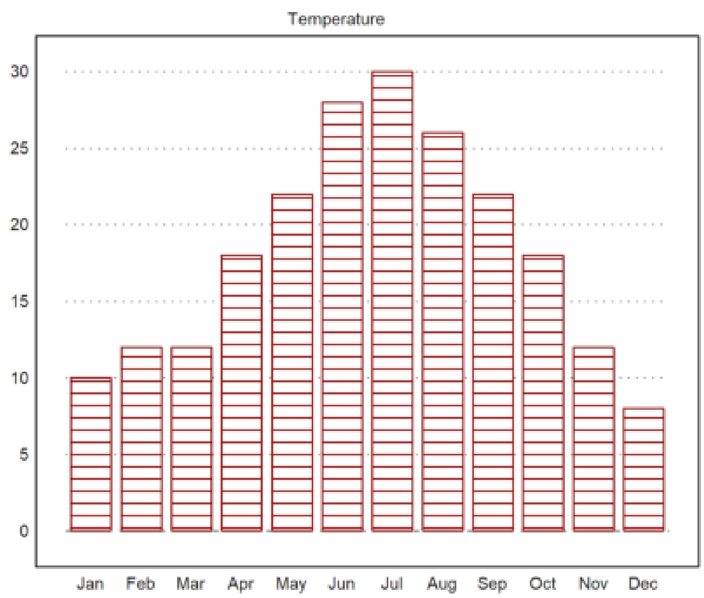



```
>columnsplot(cumsum(random(10)),style="/",color=blue):
```

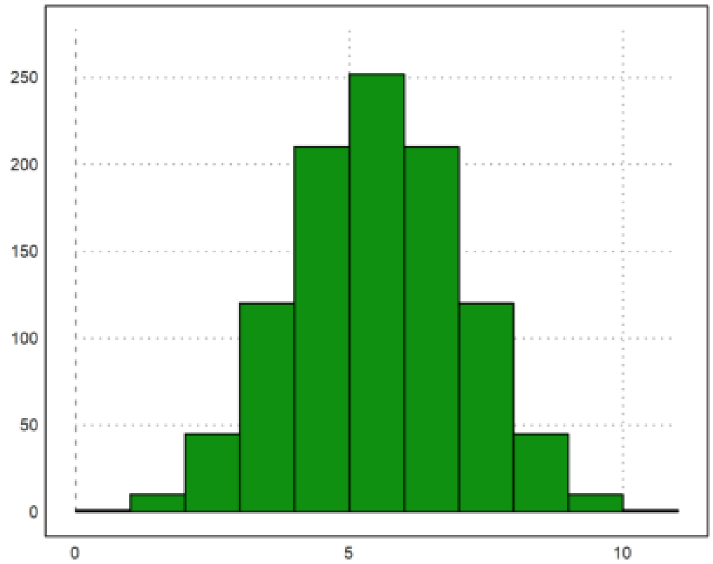


Itu juga dapat menampilkan string sebagai label.

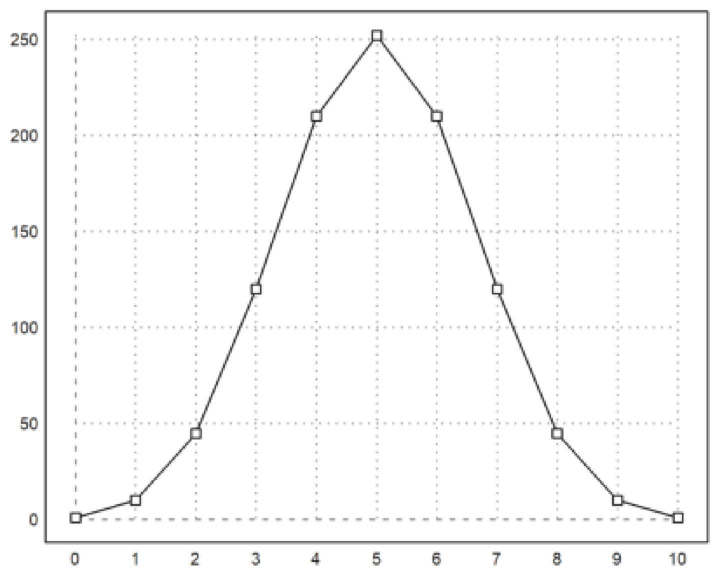
```
>months=["Jan","Feb","Mar","Apr","May","Jun", ...
>  "Jul","Aug","Sep","Oct","Nov","Dec"];
>values=[10,12,12,18,22,28,30,26,22,18,12,8];
>columnsplot(values,lab=months,color=red,style="-");
>title("Temperature"):
```



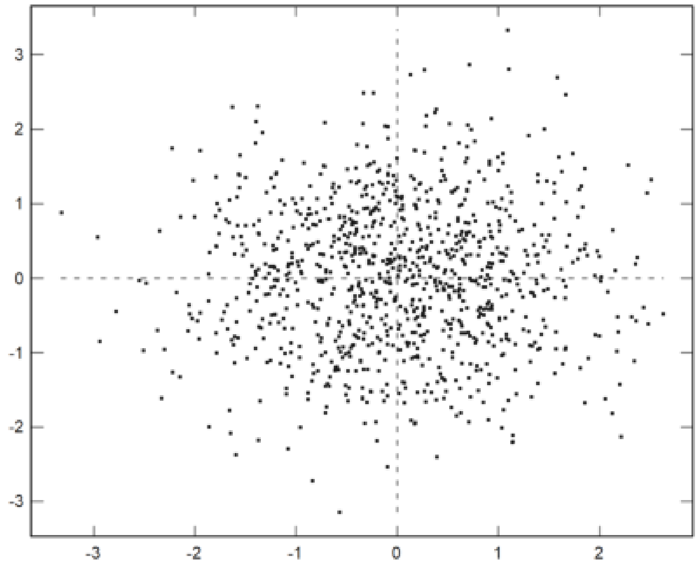

```
>k=0:10;
>plot2d(k,bin(10,k),>bar):
```



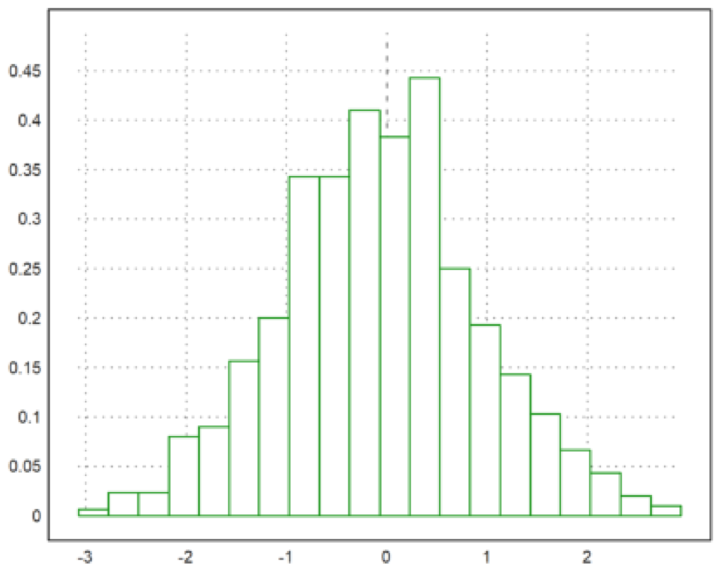

```
>plot2d(k,bin(10,k)); plot2d(k,bin(10,k),>points,>add):
```



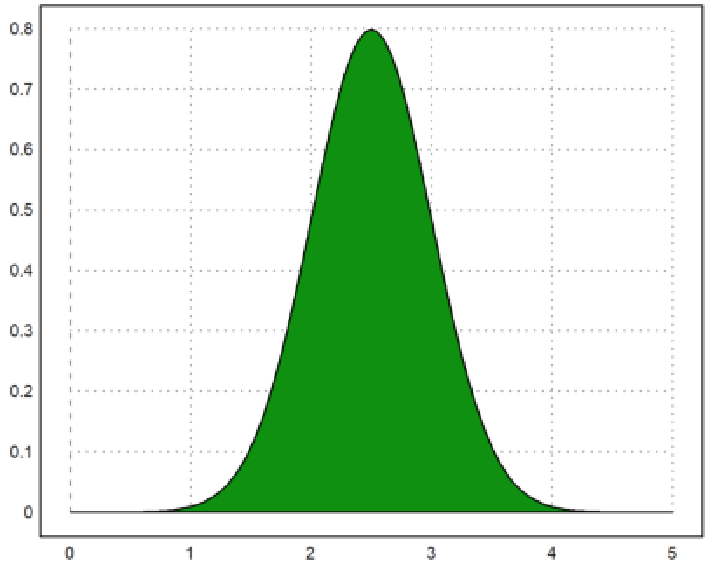

```
>plot2d(normal(1000),normal(1000),>points,grid=6,style=".."):
```




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

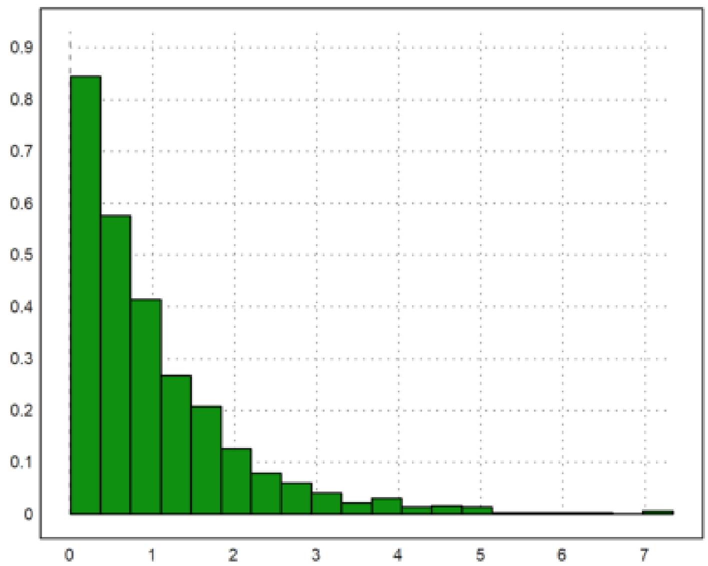



```
>plot2d("qnormal",0,5;2.5,0.5,>filled):
```



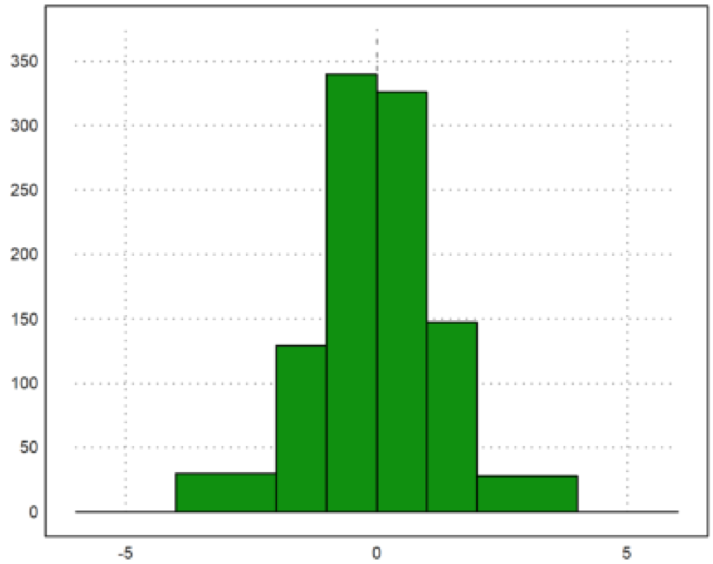
Untuk memplot distribusi statistik eksperimental, Anda dapat menggunakan distribution=n dengan plot2d.

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



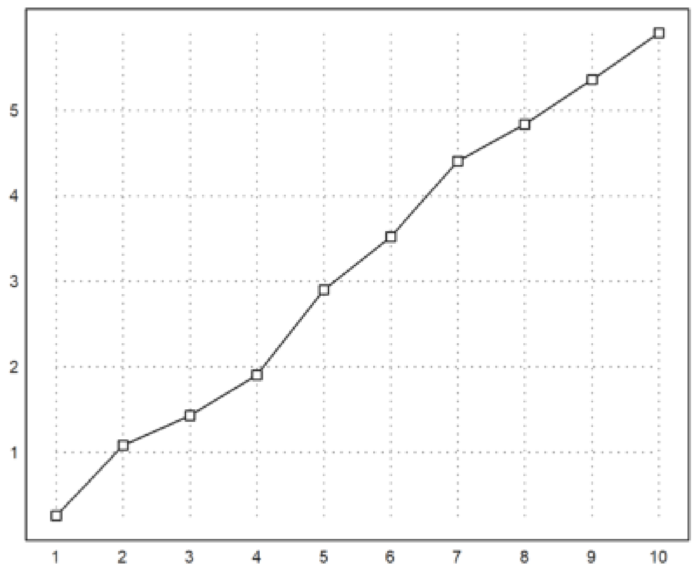
Atau Anda dapat menghitung distribusi dari data dan memplot hasilnya dengan >bar di plot3d, atau dengan plot kolom.

```
>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):
```

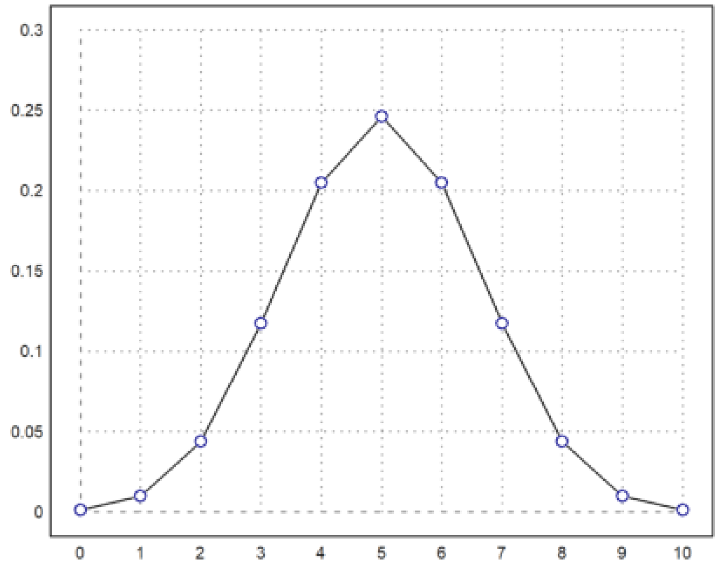


Fungsi statplot() menyetel gaya dengan string sederhana.

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

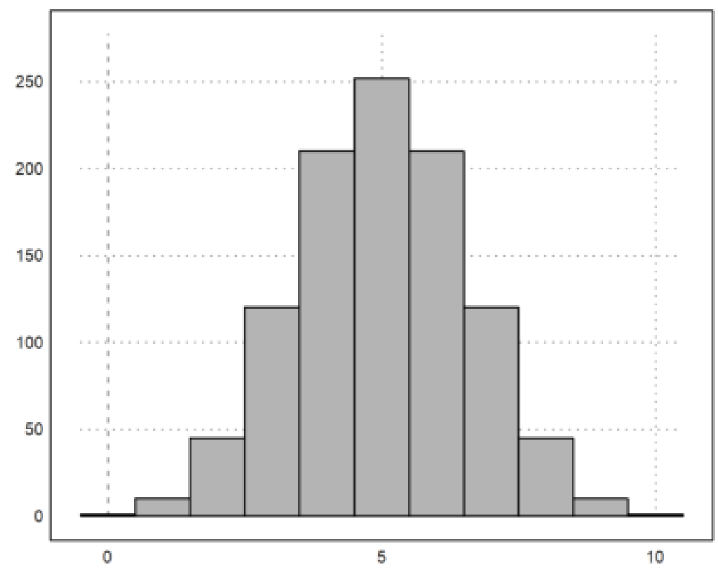



```
>n=10; i=0:n; ...
>plot2d(i,bin(n,i)/2^n,a=0,b=10,c=0,d=0.3); ...
>plot2d(i,bin(n,i)/2^n,points=true,style="ow",add=true,color=blue):
```



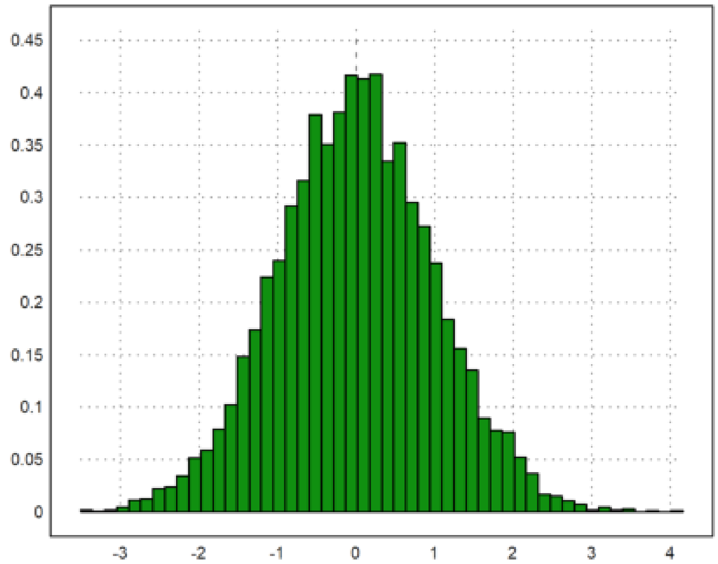
Selain itu, data dapat diplot sebagai batang. Dalam hal ini, x harus diurutkan dan satu elemen lebih panjang dari y. Bilah akan memanjang dari x[i] ke x[i+1] dengan nilai y[i]. Jika x memiliki ukuran yang sama dengan y, maka akan diperpanjang satu elemen dengan spasi terakhir.
Gaya isian dapat digunakan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

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

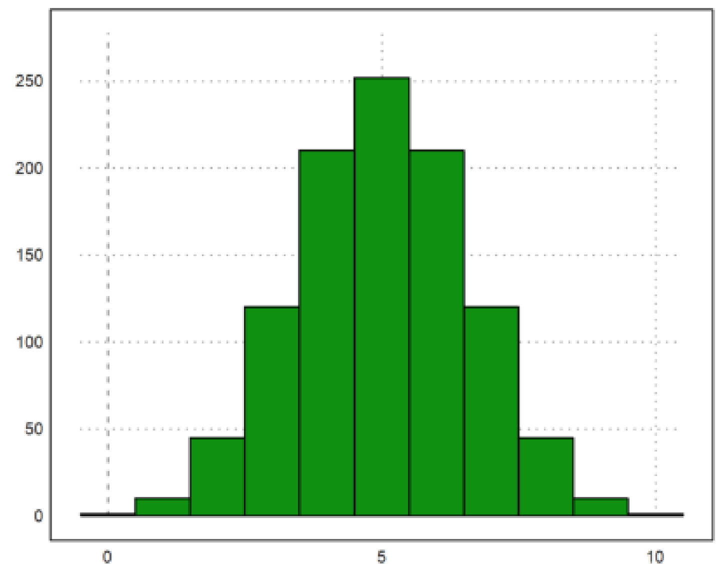


Data untuk plot batang (bar=1) dan histogram (histogram=1) dapat dinyatakan secara eksplisit dalam xv dan yv, atau dapat dihitung dari distribusi empiris dalam xv dengan >distribusi (atau distribusi=n). Histogram nilai xv akan dihitung secara otomatis dengan >histogram. Jika >genap ditentukan, nilai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

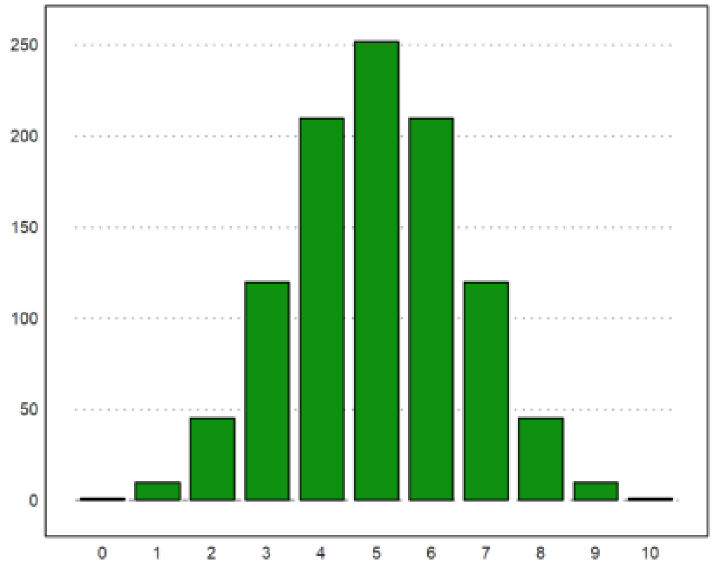
```
>plot2d(normal(10000),distribution=50):
```



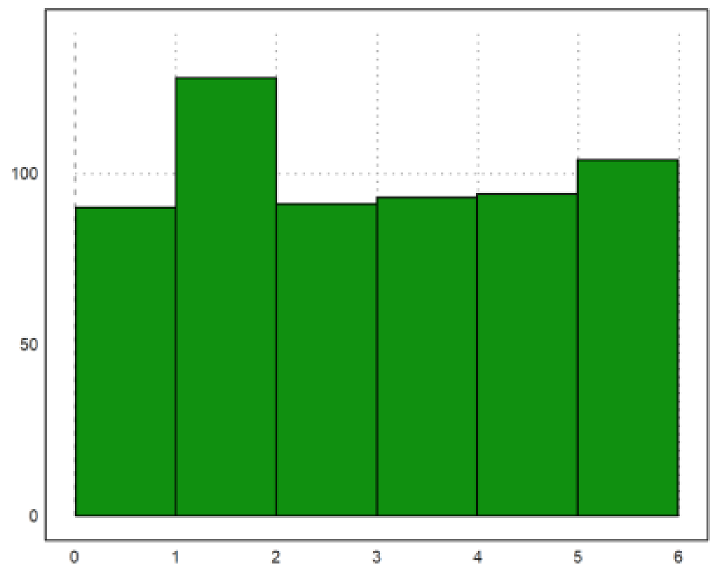

```
>k=0:10; m=bin(10,k); x=(0:11)-0.5; plot2d(x,m,>bar):
```




```
>columnsplot(m,k):
```

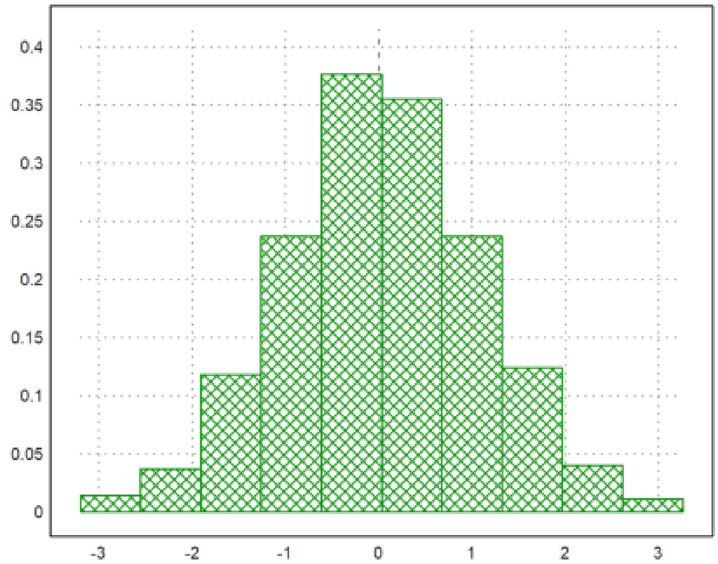



```
>plot2d(random(600)*6,histogram=6):
```



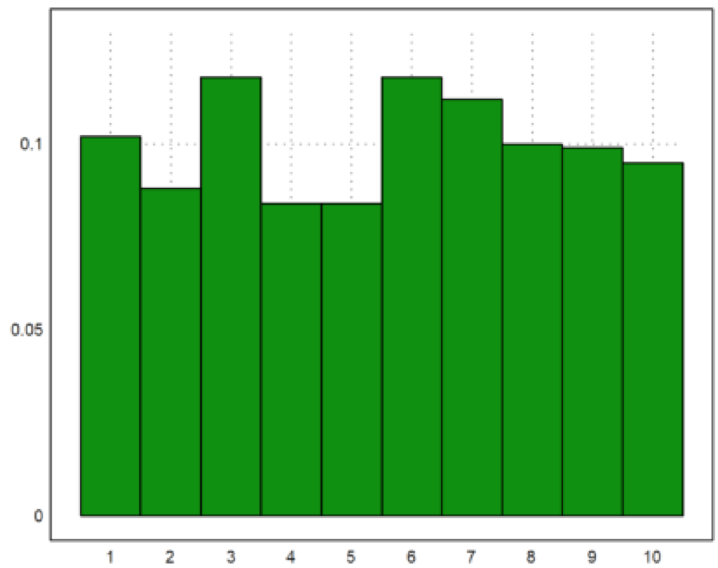
Untuk distribusi, ada parameter distribution=n, yang menghitung nilai secara otomatis dan mencetak distribusi relatif dengan n sub-interval.

```
>plot2d(normal(1,1000),distribution=10,style="\/"):
```



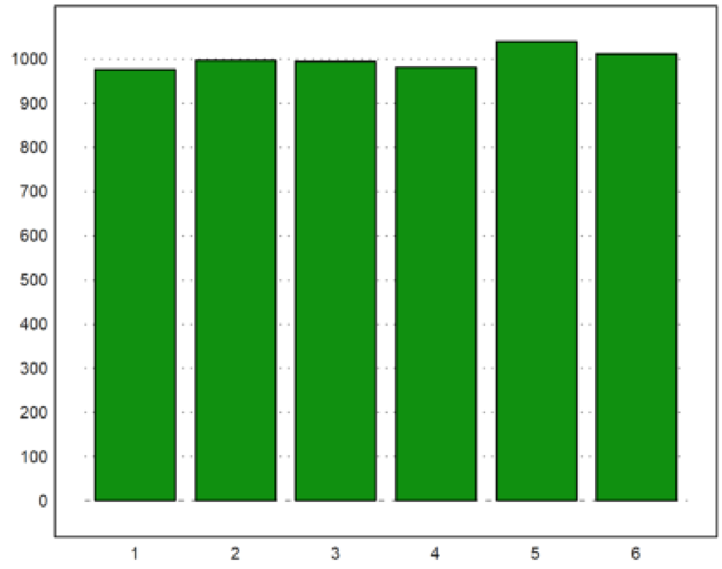
Dengan parameter even=true, ini akan menggunakan interval integer.

```
>plot2d(intrandom(1,1000,10),distribution=10,even=true):
```

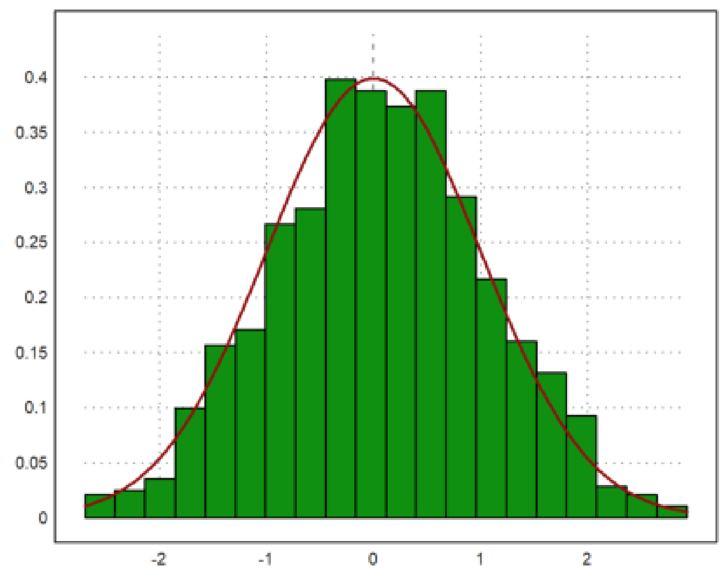


Perhatikan bahwa ada banyak plot statistik, yang mungkin berguna. Silahkan lihat tutorial tentang statistik.

```
>columnsplot(getmultiplicities(1:6,intrandom(1,6000,6))):
```

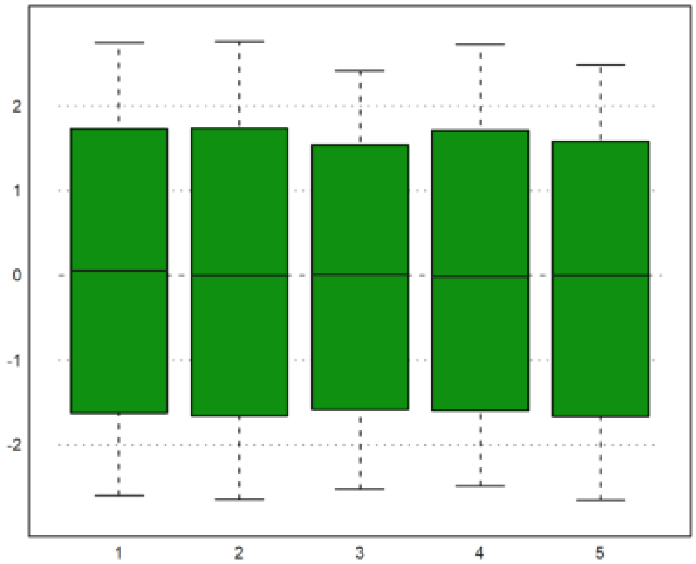



```
>plot2d(normal(1,1000),>distribution); ...
>  plot2d("qnormal(x)",color=red,thickness=2,>add):
```



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.

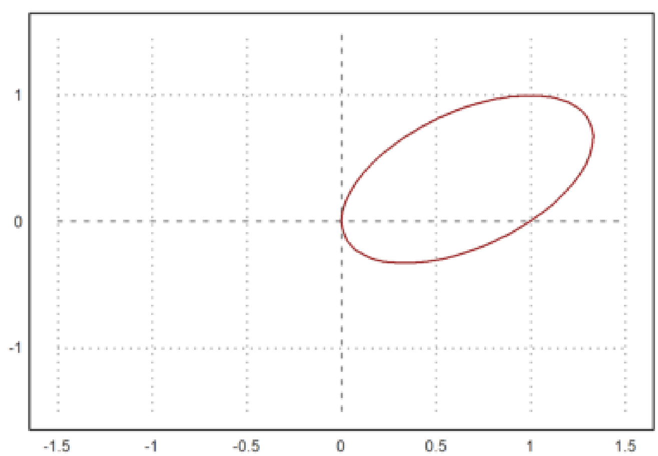
```
>M=normal(5,1000); boxplot(quartiles(M)):
```



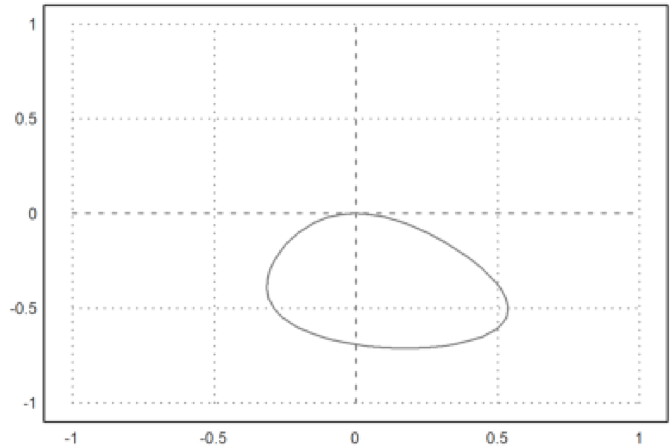
Fungsi Implisit

Plot implisit menunjukkan garis level yang menyelesaikan f(x,y)=level, di mana "level" dapat berupa nilai tunggal atau vektor nilai. Jika level="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 >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
lateks: 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 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.

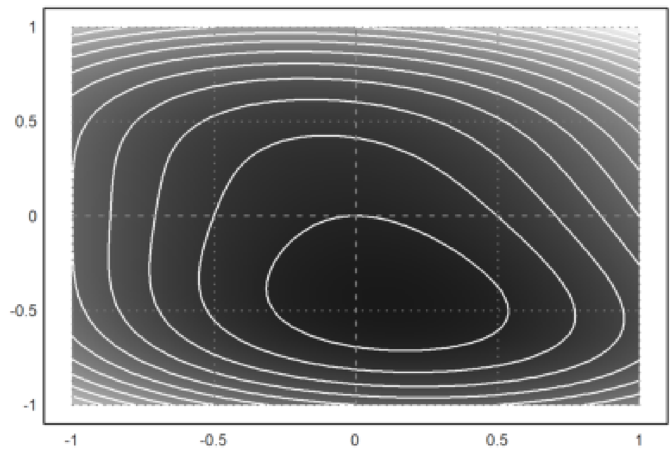
```
>aspect(1.5);
>plot2d("x^2+y^2-x*y-x",r=1.5,level=0,contourcolor=red):
```



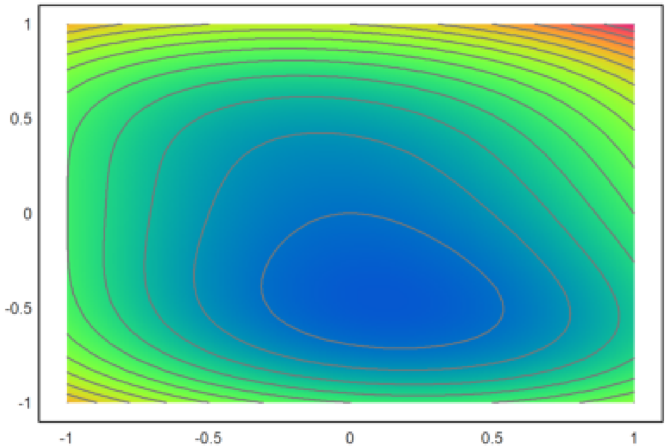
```
>expr := "2*x^2+x*y+3*y^4+y"; // define an expression f(x,y)
>plot2d(expr,level=0): // Solutions of f(x,y)=0
```




```
>plot2d(expr,level=0:0.5:20,>hue,contourcolor=white,n=200): // nice
```

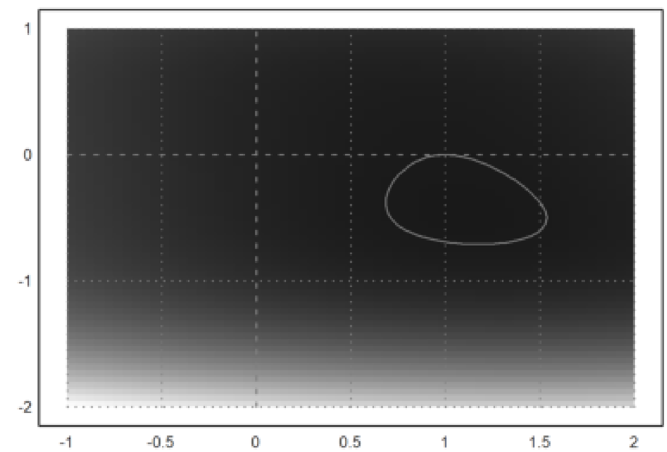



```
>plot2d(expr,level=0:0.5:20,>hue,>spectral,n=200,grid=4): // nicer
```

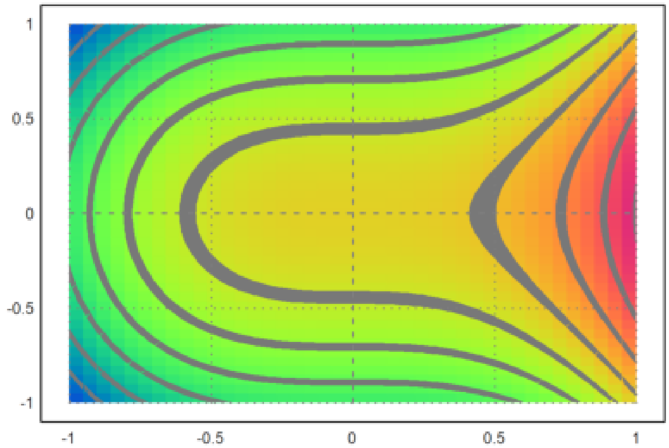


Ini berfungsi untuk plot data juga. Tetapi Anda harus menentukan rentangnya untuk label sumbu.

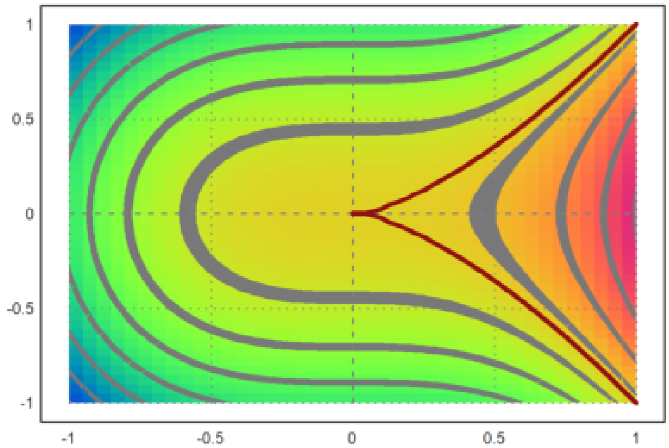
```
>x=-2:0.05:1; y=x'; z=expr(x,y);
>plot2d(z,level=0,a=-1,b=2,c=-2,d=1,>hue):
```



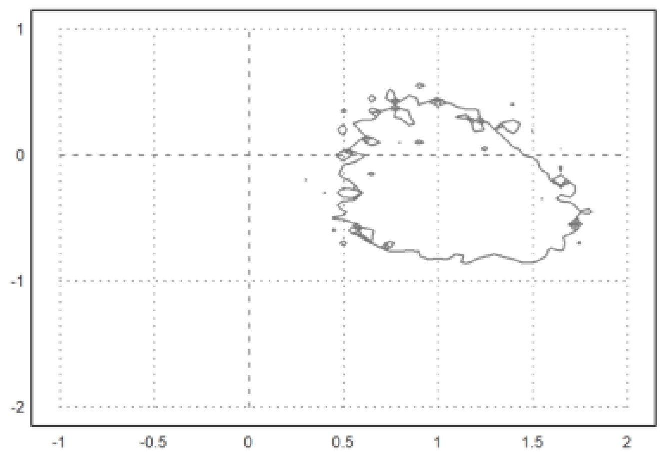

```
>plot2d("x^3-y^2",>contour,>hue,>spectral):
```



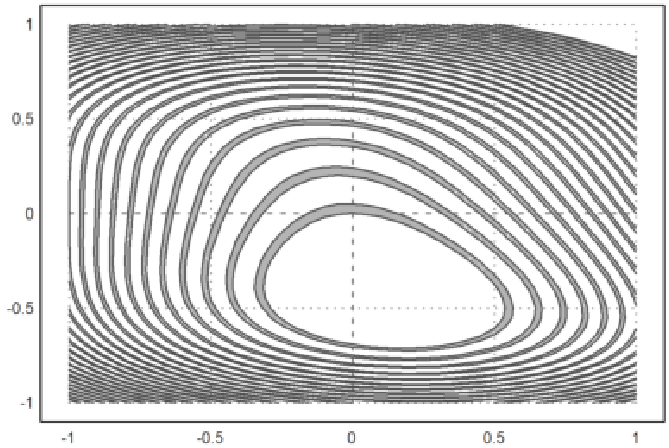

```
>plot2d("x^3-y^2",level=0,contourwidth=3,>add,contourcolor=red):
```



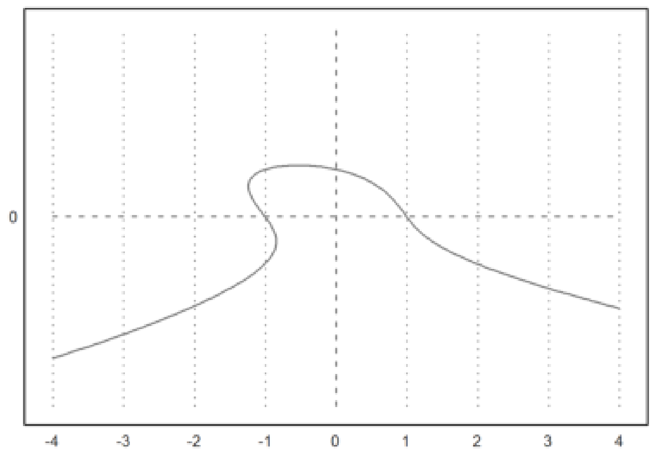

```
>z=z+normal(size(z))*0.2;
>plot2d(z,level=0.5,a=-1,b=2,c=-2,d=1):
```



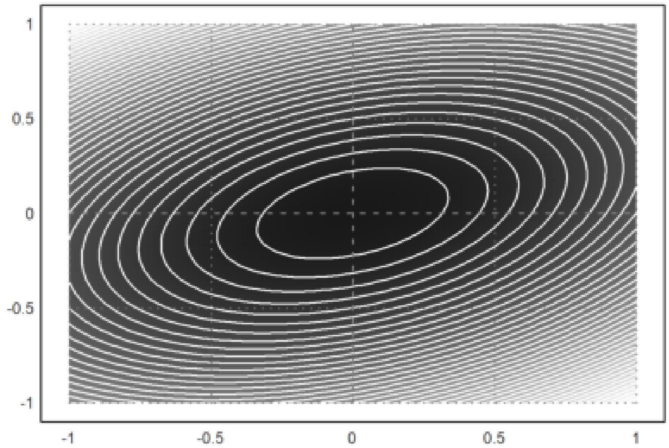
```
>plot2d(expr,level=[0:0.2:5;0.05:0.2:5.05],color=lightgray):
```



```
>plot2d("x^2+y^3+x*y",level=1,r=4,n=100):
```

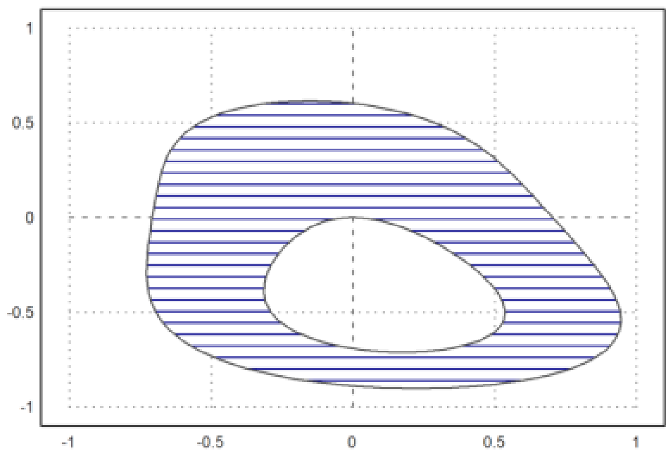


```
>plot2d("x^2+2*y^2-x*y",level=0:0.1:10,n=100,contourcolor=white,>hue):
```



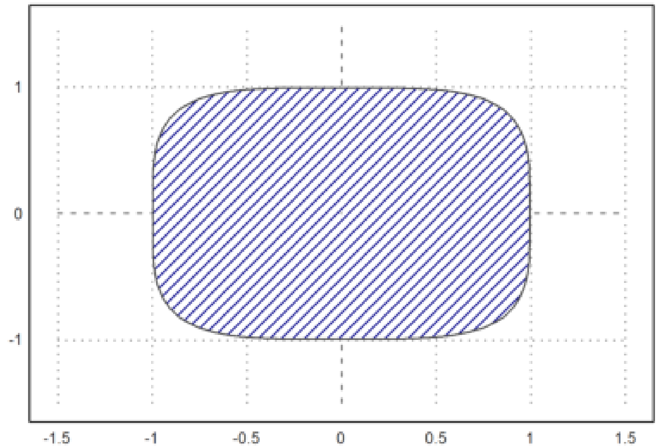
Juga dimungkinkan untuk mengisi set
lateks: a \le f(x,y) \le b
dengan rentang tingkat.
Dimungkinkan untuk mengisi wilayah nilai untuk fungsi tertentu. Untuk ini, level harus berupa matriks 2xn. Baris pertama adalah batas bawah dan baris kedua berisi batas atas.

```
>plot2d(expr,level=[0;1],style="-",color=blue): // 0 <= f(x,y) <= 1
```

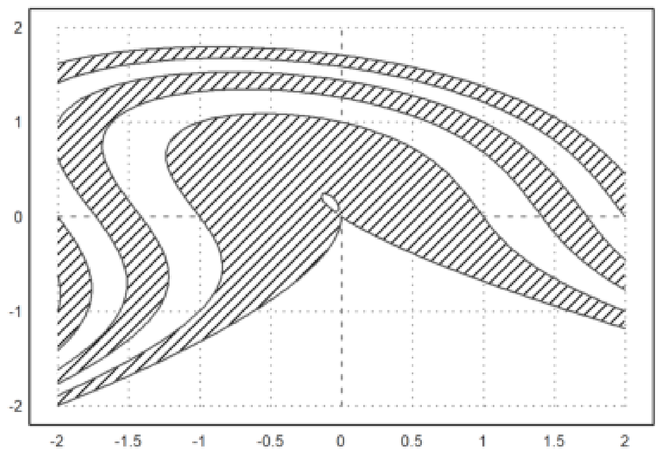


Plot implisit juga dapat menunjukkan rentang level. Kemudian level harus berupa matriks 2xn dari interval level, di mana baris pertama berisi awal dan baris kedua adalah akhir dari setiap interval. Atau, vektor baris sederhana dapat digunakan untuk level, dan parameter dl memperluas nilai level ke interval.

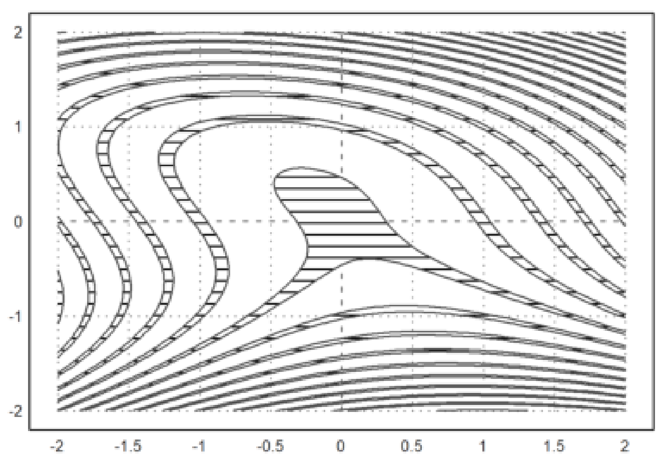
```
>plot2d("x^4+y^4",r=1.5,level=[0;1],color=blue,style="/"):
```



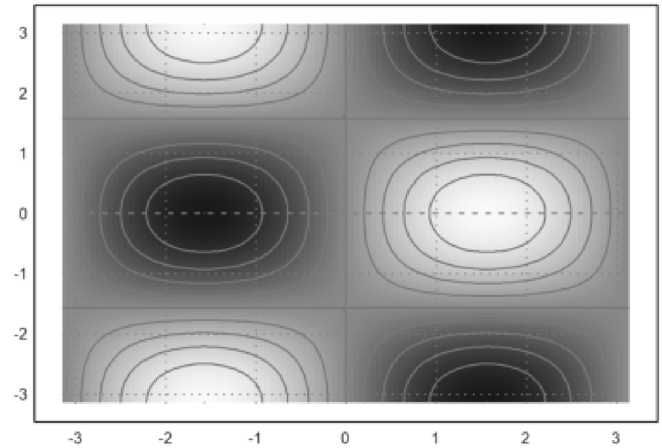

```
>plot2d("x^2+y^3+x*y",level=[0,2,4;1,3,5],style="/",r=2,n=100):
```




```
>plot2d("x^2+y^3+x*y",level=-10:20,r=2,style="-",dl=0.1,n=100):
```

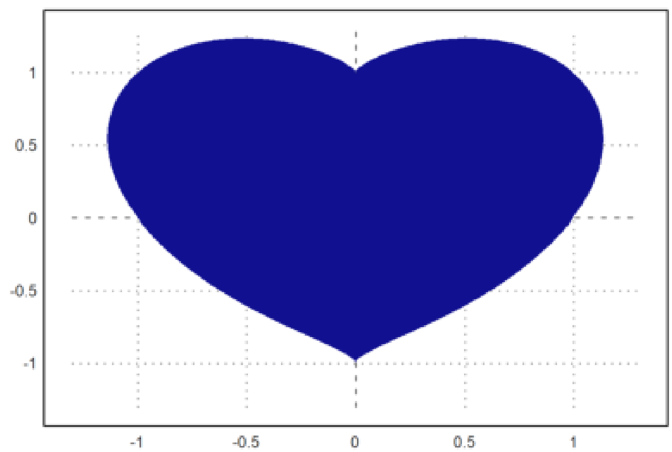


```
>plot2d("sin(x)*cos(y)",r=pi,>hue,>levels,n=100):
```



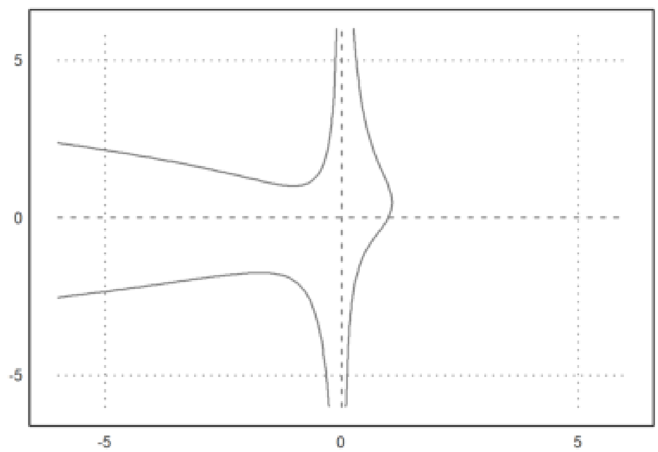
Dimungkinkan juga untuk menandai suatu wilayah
lateks: a \le f(x,y) \le b.
Ini dilakukan dengan menambahkan level dengan dua baris.

```
>plot2d("(x^2+y^2-1)^3-x^2*y^3",r=1.3, ...
>  style="#",color=blue,<outline, ...
>  level=[-2;0],n=100):
```



Dimungkinkan untuk menentukan level tertentu. Misalnya, kita dapat memplot solusi persamaan seperti
lateks: x^3-xy+x^2y^2=6

```
>plot2d("x^3-x*y+x^2*y^2",r=6,level=1,n=100):
```

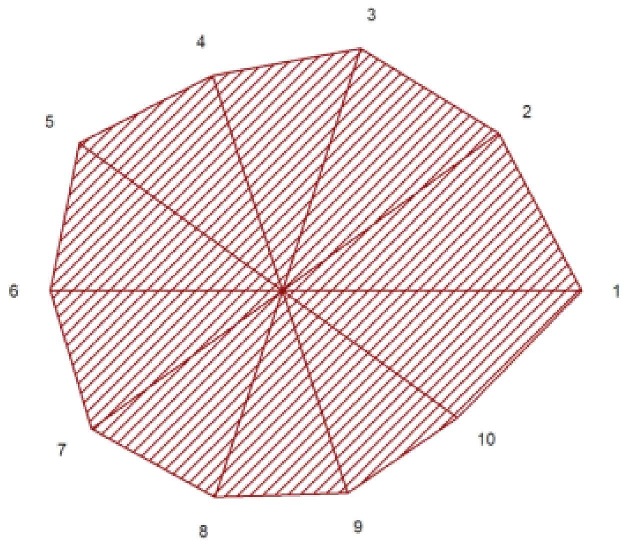



```
>function starplot1 (v, style="/", color=green, lab=none) ...
```

```
    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):
```



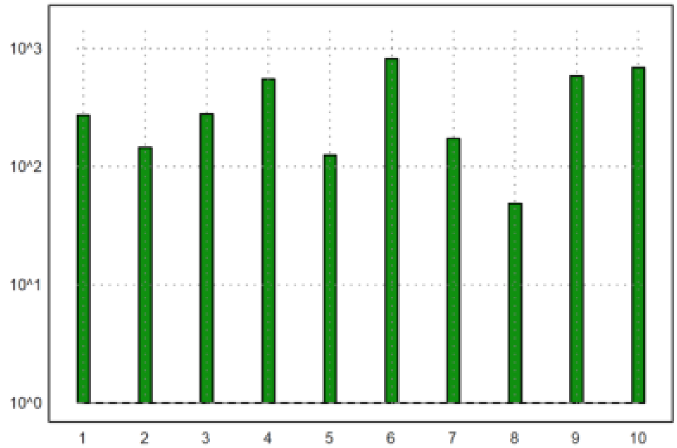
Terkadang, Anda mungkin ingin merencanakan sesuatu yang tidak dapat dilakukan plot2d, tetapi hampir. Dalam fungsi berikut, kami melakukan plot impuls logaritmik. plot2d dapat melakukan plot logaritmik, tetapi tidak untuk batang impuls.

```
>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.

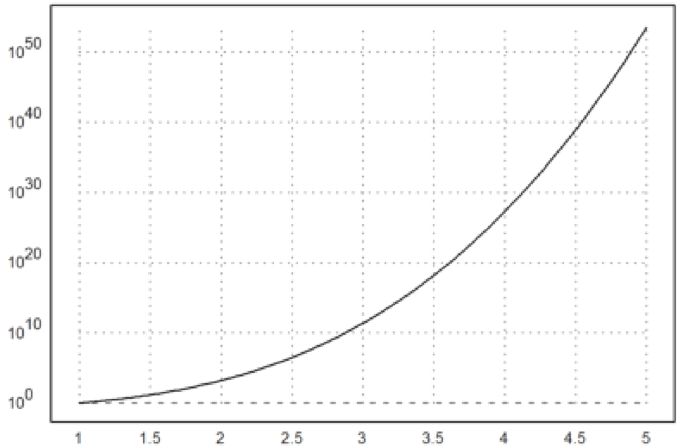
```
>animliss(2,3); // lihat hasilnya, jika sudah puas, tekan ENTER
```

Plot Logaritmik

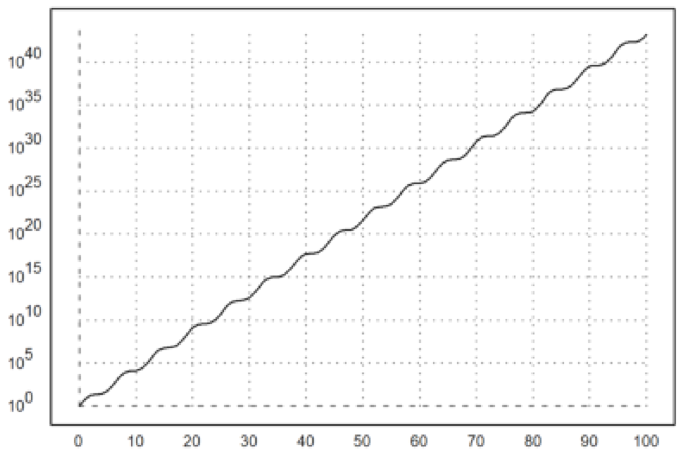
EMT menggunakan parameter "logplot" untuk skala logaritmik.
Plot logaritma dapat diplot baik menggunakan skala logaritma dalam y dengan logplot=1, atau menggunakan skala logaritma dalam x dan y dengan logplot=2, atau dalam x dengan logplot=3.

```
- logplot=1: y-logaritma
- logplot=2: x-y-logaritma
- logplot=3: x-logaritma
```

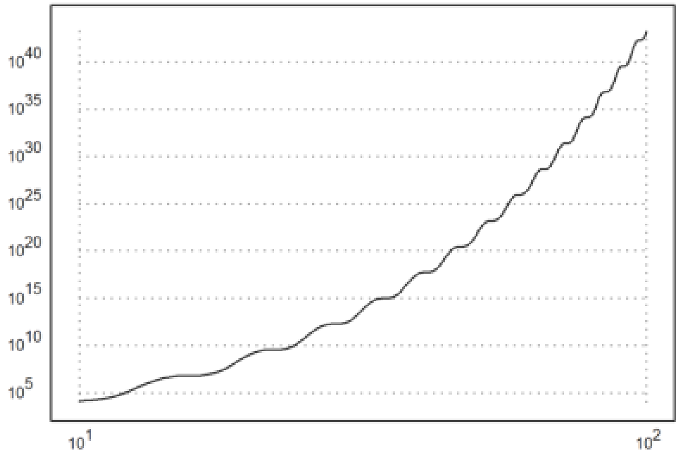
```
>plot2d("exp(x^3-x)*x^2",1,5,logplot=1):
```



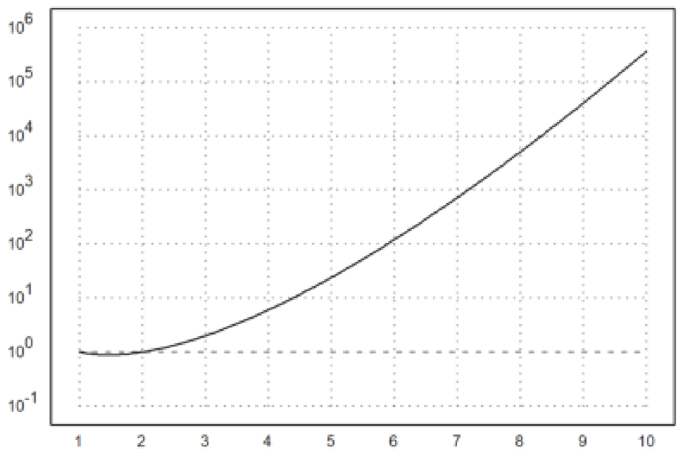

```
>plot2d("exp(x+sin(x))",0,100,logplot=1):
```



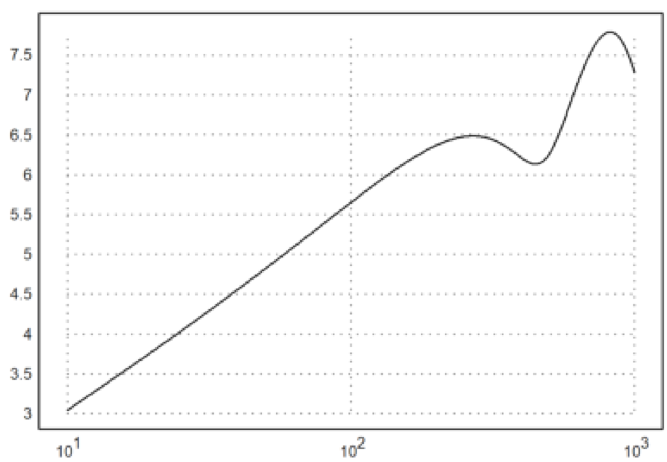

```
>plot2d("exp(x+sin(x))",10,100,logplot=2):
```




```
>plot2d("gamma(x)",1,10,logplot=1):
```

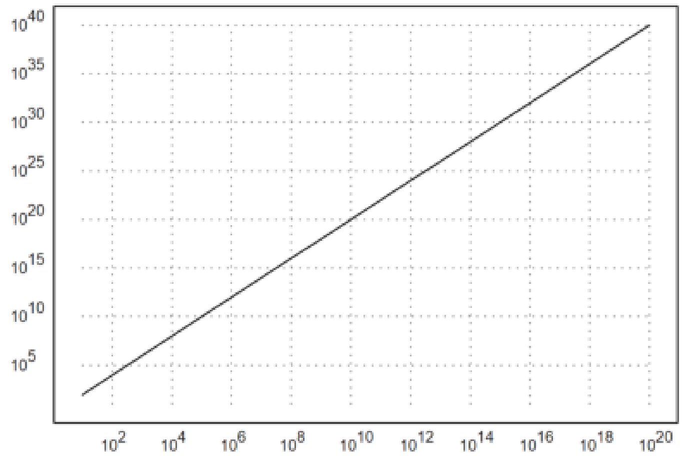



```
>plot2d("log(x*(2+sin(x/100)))",10,1000,logplot=3):
```



Ini juga berfungsi dengan plot data.

```
>x=10^(1:20); y=x^2-x;
>plot2d(x,y,logplot=2):
```

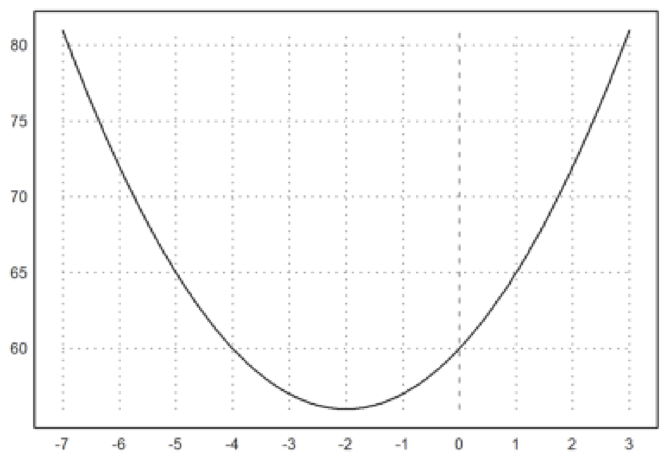


Latihan Soal

1. Tentukan titik puncak dari persamaan

$$x^2 + 4x + 60$$

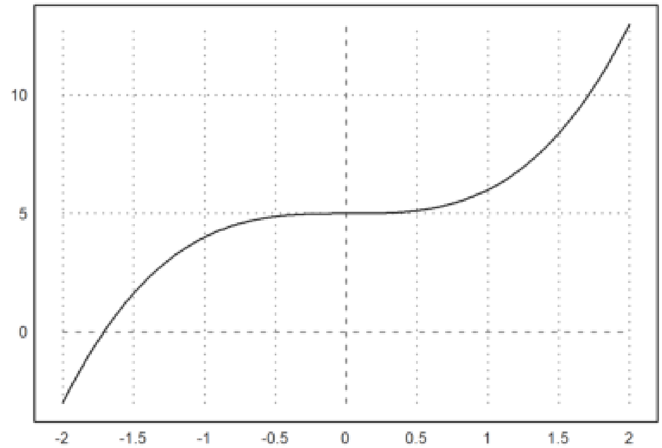
```
>plot2d("x^2+4*x+60",-7,3):
```



Jadi, titik puncaknya berada di titik (-2,0)
2. Gambarkan kurva fungsi

$$f(x) = x^3 + 5$$

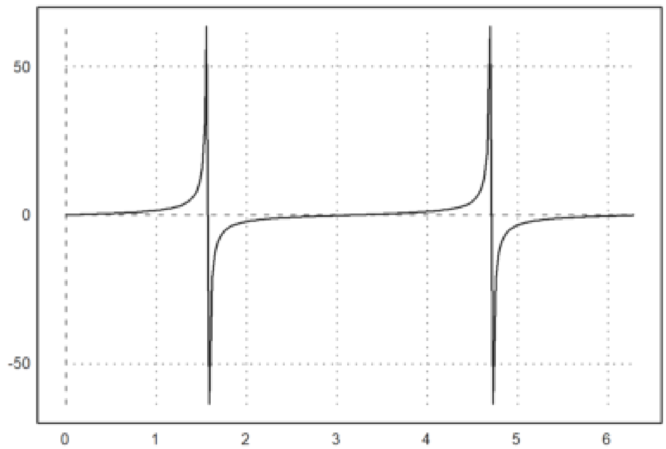
```
>plot2d("x^3+5"):
```



3. Gambarkan kurva

$$f(x) = tanx$$

```
>plot2d("tan(x)",0,2pi):
```

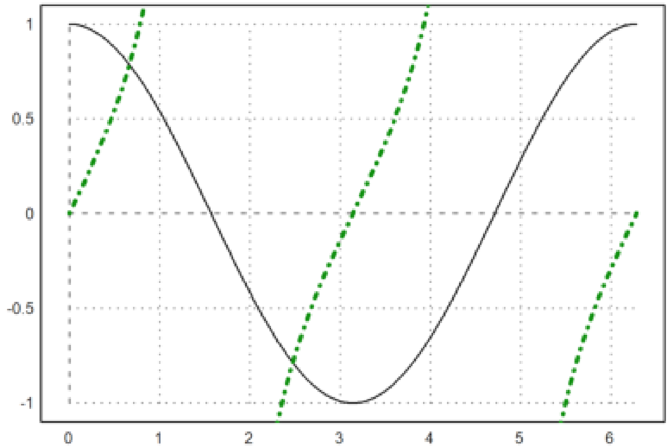


4. Buatlah grafik dari

$$f(x) = cos(x)$$

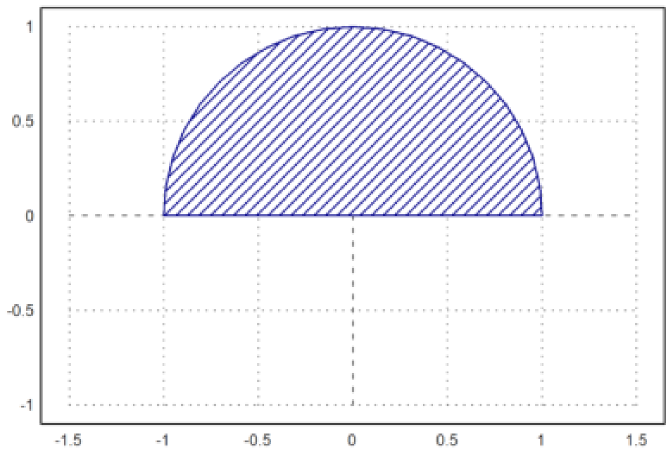
$$f(x) = tan(x)$$

```
>aspect(1.5); plot2d("cos(x)",0,2pi); plot2d("tan(x)",color=green,thickness=3,style="-.",>
```



5. Gambarkan setengah lingkaran

```
>t=linspace(0,pi,20); ...
>plot2d(cos(t),sin(t),>filled,style="/",fillcolor=blue,r=1):
```



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
"#", "#O", "O", "/", "\", "\/",
"+", "|", "-", "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.

BAB 3

KB PEKAN 5: MENGGUNAKAN EMT UNTUK MENGAMBAR GRAFIK 3 DIMENSI (3D)

[a4paper,10pt]article eumat

Nama : Fransisca Renita Pejoresa
Kelas: Matematika E 2022
NIM : 22305144012

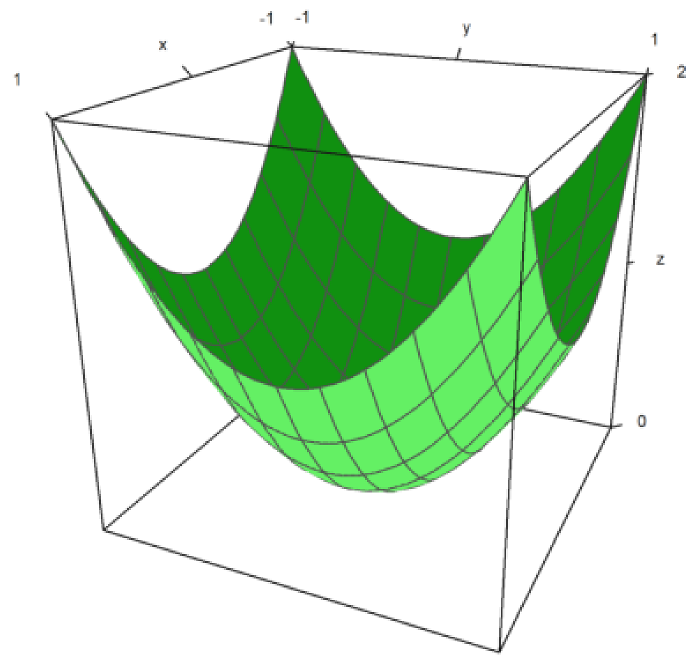
Menggambar Plot 3D dengan EMT

1. Menggambar Grafik Fungsi Dua Variabel * dalam Bentuk Ekspresi

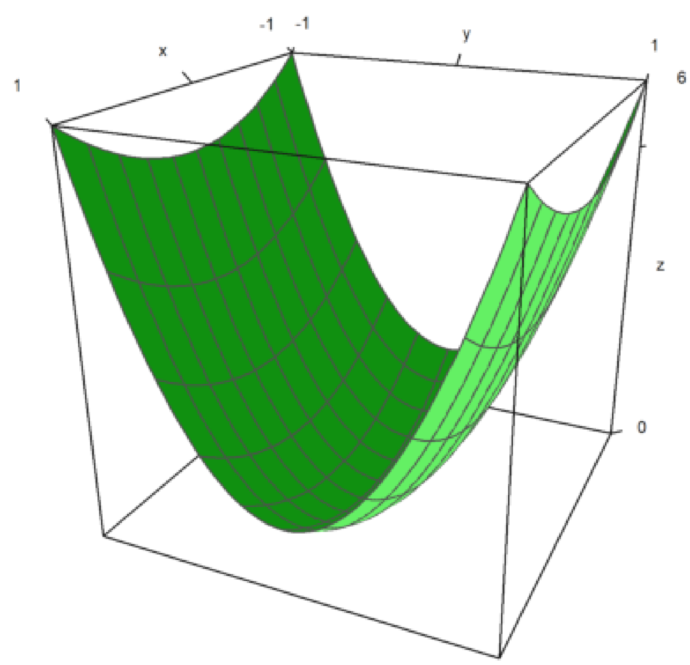
Langsung
Fungsi Dua Variabel didefinisikan sebagai sebuah fungsi bernilai real dari dua variabel real, yakni fungsi f yang memadankan setiap pasangan
terurut (x,y) pada suatu himpunan D dari bidang dengan bilangan real
tunggal f (x,y).
Di dalam program numerik EMT, ekspresi adalah string. Jika ditandai sebagai simbolis, mereka akan mencetak melalui Maxima, jika tidak melalui EMT. Ekspresi dalam string digunakan untuk membuat plot dan banyak fungsi numerik. Untuk ini, variabel dalam ekspresi harus "x" dan "y".
Untuk grafik suatu fungsi, gunakan
- ekspresi sederhana dalam x dan y,
- nama fungsi dari dua variabel
- atau matriks data.

contoh:

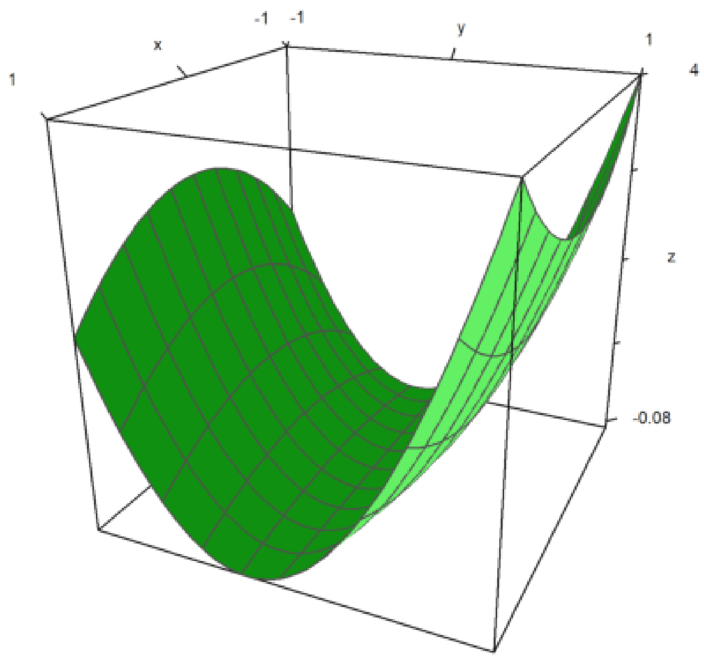
```
>plot3d("x^2+y^2"):
```



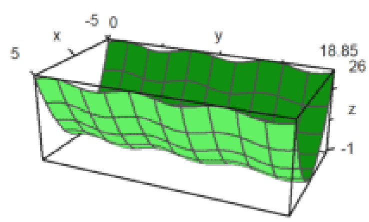

```
>plot3d("x^2+5*y^2"):
```



```
>plot3d("x^2*y+3*y^2"):
```




```
>aspect(1.5); plot3d("x^2+sin(y)",-5,5,0,6*pi):
```



1. aspect(1.5) mengatur aspek rasio pada grafik 3D.
2. plot3d("x^2+sin(y)",-5,5,0,6*pi) adalah fungsi matematika yang digunakan untuk membuat grafik 3D.
3. -5,5 mengatur rentang sumbu x yang akan ditampilkan pada grafik.
4. 0,6*pi mengatur rentang sumbu y yang akan ditampilkan pada grafik.

Fungsi umum untuk plot 3D.

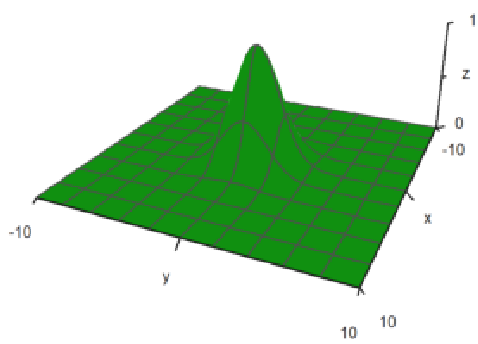
Fungsi plot3d (x, y, z, xmin, xmax, ymin, ymax, n, a, ..
b, c, d, r, scale, fscale, frame, angle, height, zoom, distance, ..)
Rentang plot untuk fungsi dapat ditentukan dengan
- a,b: rentang x
- c,d: rentang y
- r : persegi simetris di sekitar (0,0).
- n : jumlah subinterval untuk plot.
Ada beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.
- fscale: menskalakan ke nilai fungsi (defaultnya adalah <fscale).
- scale: angka atau vektor 1x2 untuk menskalakan ke arah x dan y.
- frame: jenis bingkai (default 1).
Tampilan dapat diubah dengan berbagai cara.
- distance: jarak pandang ke plot.
- zoom: nilai zoom.
- angle: sudut terhadap sumbu y negatif dalam radian.
- height: ketinggian pandangan 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.
contoh:

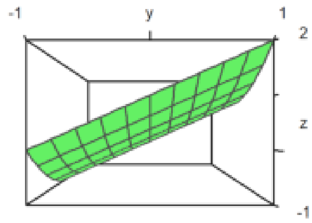
```
>plot3d("exp(-(x^2+y^2)/5)",r=10,n=80,fscale=4,scale=1.2,frame=3,>user):
```



1. exp(-(x^2+y^2)/5) adalah fungsi matematika yang digunakan untuk membuat grafik 3D.
2. r=10 mengatur jarak maksimum dari pusat grafik ke tepi grafik.
3. n=80 mengatur jumlah titik yang digunakan untuk membuat grafik.
4. fscale=4 mengatur faktor skala untuk warna.
5. scale=1.2 mengatur faktor skala untuk ukuran grafik.
6. frame=3 mengatur jenis bingkai yang digunakan untuk grafik.

Pada contoh berikut, sudut=0 dan tinggi=0 dilihat dari sumbu y negatif. Label sumbu untuk y disembunyikan dalam kasus ini.

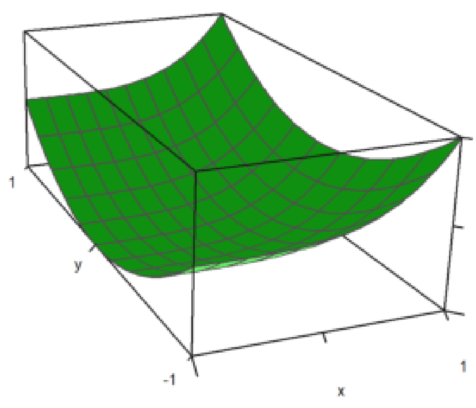
```
>plot3d("x^2+y",distance=3,zoom=1,angle=pi/2,height=0):
```



1. x^2+y adalah fungsi matematika yang digunakan untuk membuat grafik 3D.
2. distance=3 mengatur jarak pandang dari grafik.
3. zoom=1 mengatur faktor perbesaran grafik.
4. angle=pi/2 mengatur sudut pandang grafik dalam radian.
5. height=0 mengatur ketinggian pandangan dari grafik.

Plot selalu terlihat berada di tengah kubus plot. Anda dapat memindahkan bagian tengah 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):
```



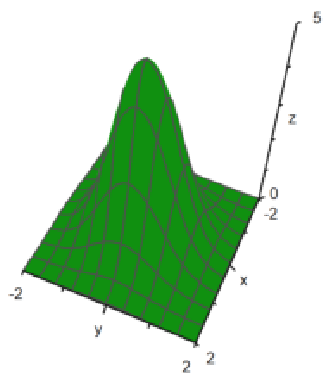
1. x^4+y^2 adalah fungsi matematika yang digunakan untuk membuat grafik 3D.
2. a=0,b=1,c=-1,d=1 mengatur rentang sumbu x dan y yang akan ditampilkan pada grafik.
3. angle=-20° mengatur sudut pandang grafik dalam derajat.
4. height=20° mengatur ketinggian pandangan dari grafik dalam derajat.

5. center=[0.4,0,0] mengatur pusat pandangan dari grafik.
6. zoom=5 mengatur faktor perbesaran grafik.

Plotnya diskalakan agar sesuai dengan unit kubus untuk dilihat. Jadi tidak perlu mengubah jarak atau zoom tergantung ukuran plot. Namun labelnya mengacu pada ukuran sebenarnya.

Jika Anda mematikannya dengan scale=false, Anda harus berhati-hati agar plot tetap masuk ke dalam jendela plotting, dengan mengubah jarak pandang atau zoom, dan memindahkan bagian tengah.

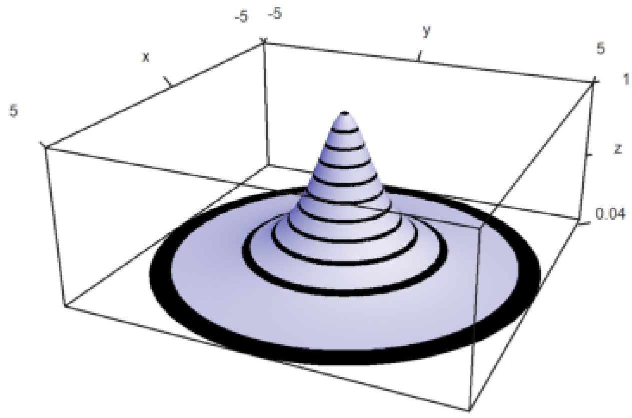
```
>plot3d("5*exp(-x^2-y^2)",r=2,<fscale,<scale,distance=13,height=50°, ...
>  center=[0,0,-2],frame=3):
```



1. 5*exp(-x^2-y^2) adalah fungsi matematika yang digunakan untuk membuat grafik 3D.
2. r=2 mengatur jarak maksimum dari pusat grafik ke tepi grafik.
3. <fscale mengatur faktor skala untuk warna.
4. <scale mengatur faktor skala untuk ukuran grafik.
5. distance=13 mengatur jarak pandang dari grafik.
6. height=50° mengatur ketinggian pandangan dari grafik dalam derajat.
7. center=[0,0,-2] mengatur pusat pandangan dari grafik.
8. frame=3 mengatur jenis bingkai yang digunakan untuk grafik.

Plot kutub juga tersedia. Parameter polar=true menggambar plot kutub. Fungsi tersebut harus tetap merupakan fungsi dari x dan y. Parameter "fscale" menskalakan fungsi dengan skalanya sendiri. Kalau tidak, fungsinya akan diskalakan agar sesuai dengan kubus.

```
>plot3d("1/(x^2+y^2+1)",r=5,>polar, ...
>fscale=2,>hue,n=100,zoom=4,>contour,color=blue):
```

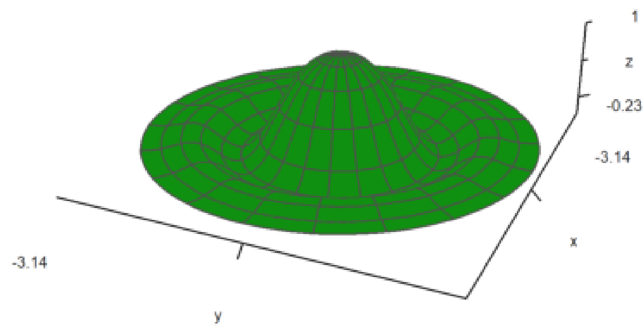


1. 1/(x^2+y^2+1) adalah fungsi matematika yang digunakan untuk membuat grafik 3D.
2. r=5 mengatur jarak maksimum dari pusat grafik ke tepi grafik.
3. polar mengatur tampilan grafik dalam koordinat polar.
4. fscale=2 mengatur faktor skala untuk warna.
5. hue mengatur skala warna yang digunakan pada grafik.
6. n=100 mengatur jumlah titik yang digunakan untuk membuat grafik.
7. zoom=4 mengatur faktor perbesaran grafik.
8. contour mengatur tampilan garis kontur pada grafik.
9. color=blue mengatur warna garis kontur pada grafik.

```
>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):
```

```
Function plot3d needs at least one argument!
Use: plot3d (x {, y: none, z: none, xmin: none, xmax: none, ...})
Error in:
plot3d"f(x^2+y^2)",>polar,scale=[1,1,0.4],r=pi,frame=3,zoom=4) ...
      ^
```

```
>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):
```

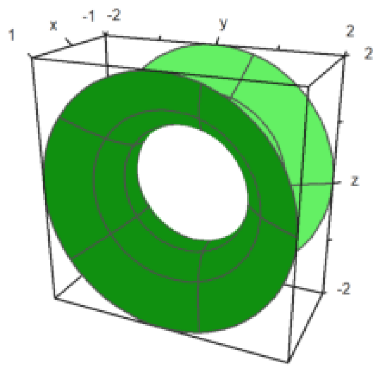


1. function f(r) := exp(-r/2)*cos(r) adalah fungsi matematika yang didefinisikan sebagai f(r) = e^(-r/2) * cos(r).
2. plot3d("f(x^2+y^2)",polar,scale=[1,1,0.4],r=pi,frame=3,zoom=4) adalah perintah untuk membuat grafik 3D dari fungsi f(x^2+y^2).
3. polar mengatur tampilan grafik dalam koordinat polar.
4. scale=[1,1,0.4] mengatur faktor skala untuk ukuran grafik.
5. r=pi mengatur jarak maksimum dari pusat grafik ke tepi grafik.
6. frame=3 mengatur jenis bingkai yang digunakan untuk grafik.
7. zoom=4 mengatur faktor perbesaran grafik.

Parameter memutar memutar fungsi di x di sekitar sumbu x.

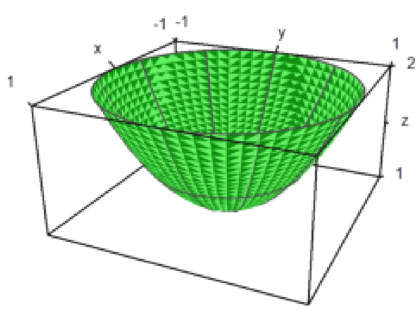
- rotate=1: Menggunakan sumbu x
- rotate=2: Menggunakan sumbu z

```
>plot3d("x^2+1",a=-1,b=1,rotate=true,grid=5):
```



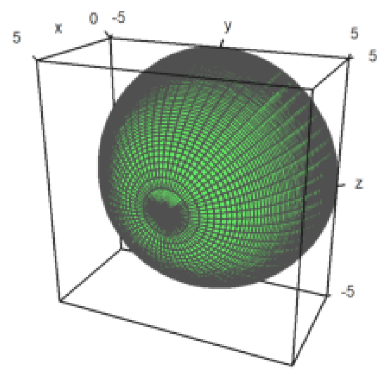
1. x^2+1 adalah fungsi matematika yang digunakan untuk membuat grafik 3D.
2. a=-1,b=1 mengatur rentang sumbu x yang akan ditampilkan pada grafik.
3. rotate=true mengatur grafik agar dapat diputar secara interaktif.
4. grid=5 mengatur jumlah garis koordinat yang ditampilkan pada grafik.

```
>plot3d("x^2+1",a=-1,b=1,rotate=2,grid=5):
```



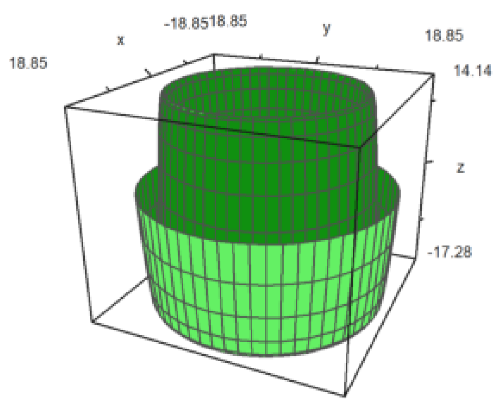
1. x^2+1 adalah fungsi matematika yang digunakan untuk membuat grafik 3D.
2. a=-1,b=1 mengatur rentang sumbu x yang akan ditampilkan pada grafik.
3. rotate=2 mengatur grafik agar dapat diputar secara interaktif dengan menggunakan mouse.
4. grid=5 mengatur jumlah garis koordinat yang ditampilkan pada grafik.

```
>plot3d("sqrt(25-x^2)",a=0,b=5,rotate=1):
```



1. sqrt(25-x^2) adalah fungsi matematika yang digunakan untuk membuat grafik 3D.
2. a=0,b=5 mengatur rentang sumbu x yang akan ditampilkan pada grafik.
3. rotate=1 mengatur grafik agar dapat diputar secara interaktif.

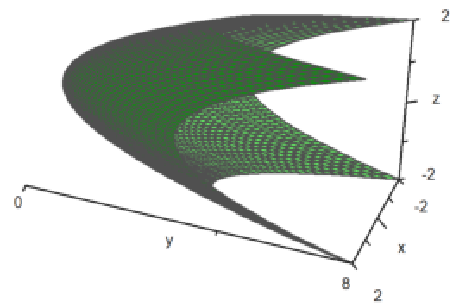
```
>plot3d("x*sin(x)",a=0,b=6pi,rotate=2):
```



1. x*sin(x) adalah fungsi matematika yang digunakan untuk membuat grafik 3D.
2. a=0,b=6pi mengatur rentang sumbu x yang akan ditampilkan pada grafik.
3. rotate=2 mengatur grafik agar dapat diputar secara interaktif.

Berikut adalah plot dengan tiga fungsi.

```
>plot3d("x","x^2+y^2","y",r=2,zoom=3.5,frame=3):
```



1. x adalah fungsi matematika yang digunakan untuk menentukan nilai sumbu x pada grafik.
2. x^2+y^2 adalah fungsi matematika yang digunakan untuk menentukan nilai sumbu z pada grafik.
3. y adalah fungsi matematika yang digunakan untuk menentukan nilai sumbu y pada grafik.
4. r=2 mengatur jarak maksimum dari pusat grafik ke tepi grafik.
5. zoom=3.5 mengatur faktor perbesaran grafik.
6. frame=3 mengatur jenis bingkai yang digunakan untuk grafik.

2. Menggambar Grafik Fungsi Dua Variabel yang

* Rumusnya Disimpan dalam Variabel Ekspresi
Fungsi ini dapat memplot plot 3D dengan grafik fungsi dua variabel, permukaan berparameter, kurva ruang, awan titik, penyelesaian persamaan tiga variabel. Semua plot 3D bisa ditampilkan sebagai anaglyph.
fungsi plot3d (x, y, z, xmin, xmax, ymin, ymax, n, a
Parameter
x : ekspresi dalam x dan y
x,y,z : matriks koordinat suatu permukaan
x,y,z : ekspresi dalam x dan y untuk permukaan parametrik
x,y,z : ekspresi dalam x untuk memplot kurva ruang
xmin,xmax,ymin,ymax :

```
x,y batas ekspresi
```

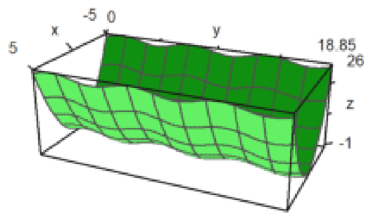
contoh:
ekspresi dalam string

```
>expr := "x^2+sin(y)"
```

```
x^2+sin(y)
```

plot ekspresi

```
>plot3d(expr,-5,5,0,6*pi):
```



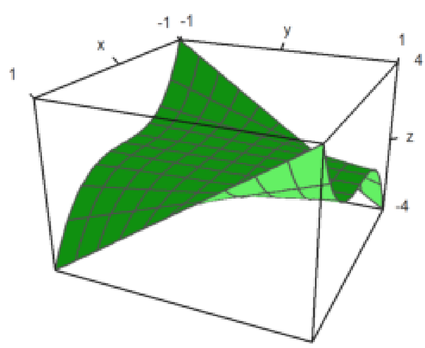
1. x^2+sin(y) adalah fungsi matematika yang digunakan untuk membuat grafik 3D.
2. -5,5 mengatur rentang sumbu x yang akan ditampilkan pada grafik.
3. 0,6*pi mengatur rentang sumbu y yang akan ditampilkan pada grafik.

contoh 1:

```
>
>expr := "4*x^3*y"
```

```
4*x^3*y
```

```
>aspect(1.5); plot3d(expr):
```



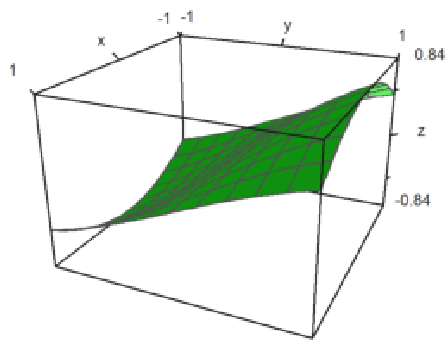
1. aspect(2) mengatur aspek rasio pada grafik 3D.
2. plot3d(expr) adalah fungsi matematika yang digunakan untuk membuat grafik 3D.

contoh 2:

```
>expr := "cos(x)*sin(y)"
```

```
cos(x)*sin(y)
```

```
>plot3d(expr):
```

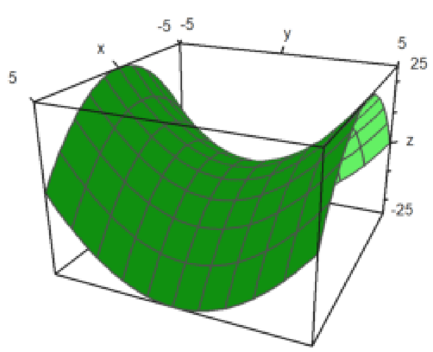


contoh 3:

```
>expr := "y^2-x^2"
```

```
y^2-x^2
```

```
>aspect(1.5); plot3d(expr,-5,5,-5,5):
```



3. Menggambar Grafik Fungsi Dua Variabel yang

* Fungsinya Didefinisikan sebagai Fungsi Numerik

Fungsi Dua Variabel

Fungsi dua variabel adalah sebuah fungsi yang bernilai real dari dua variabel real. Fungsi ini memadankan setiap pasangan terurut (x,y) pada suatu himpunan D dari bidang dengan bilangan real tunggal f(x,y). Dalam matematika, fungsi dua variabel atau lebih digunakan untuk menggambarkan hubungan antara dua atau lebih variabel.

Fungsi Numerik

Fungsi numerik adalah suatu fungsi matematika yang menghasilkan nilai numerik sebagai output-nya. Fungsi ini dapat dinyatakan dalam bentuk persamaan matematika atau algoritma komputasi.
Contoh:
Fungsi

$$f(x, y) = 5x + y$$

Misal input nilai x=2 dan y=3, maka akan dihasilkan nilai z yaitu

$$z = f(x, y) = 5(2) + 3 = 10 + 3 = 13$$

Gambar Grafik Fungsi

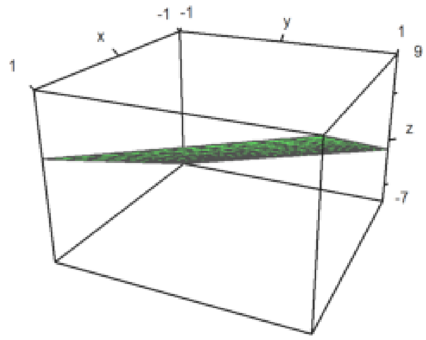
Fungsi satu baris numerik didefinisikan oleh ":=".
Langkah-langkah untuk memvisualisasikan grafik fungsi dua variabel yang fungsi nya didefinisikan sebagai fungsi numerik dalam plot3d:
1. Buat fungsi numerik yang akan digunakan untuk memvisualisasikan data.
function f(x,y):=ax+by
dimana a dan b adalah konstanta
2. Gunakan fungsi plot3d() untuk membuat grafik tiga dimensi dari fungsi numerik.
plot3d("f"):

Contoh

Fungsi matematika f(x,y) dapat digambarkan dalam bentuk grafik tiga dimensi menggunakan perintah plot3d. Berikut adalah contoh penggunaan perintah plot3d untuk menggambarkan fungsi tersebut:
1. Fungsi Linear Dua Variabel

$$f(x, y) = 5x + 3y + 1$$

```
>function f(x,y):= 5*x+3*y+1
>plot3d("f"):
```



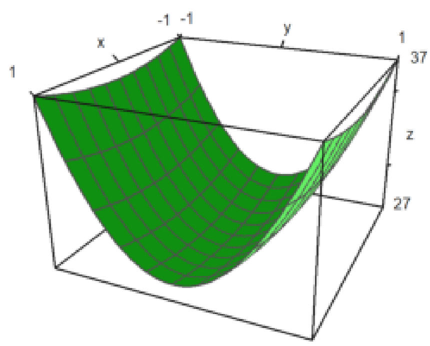
- Fungsi f(x,y) didefinisikan sebagai 5x+3y+1.
- Perintah "plot3d("f")" digunakan untuk memplot grafik 3D dari fungsi f(x,y) menggunakan fungsi plot3d di EMT.
- Grafik yang dihasilkan akan menampilkan fungsi dalam tiga dimensi, dengan sumbu x dan y mewakili variabel masukan dan sumbu z mewakili nilai keluaran fungsi. Grafik akan menunjukkan bentuk fungsi dan perubahannya seiring dengan perubahan variabel masukan.

---

2. Fungsi Kuadrat Dua Variabel

$$f(x, y) = x^2 + 3y^2 + 21$$

```
>function f(x,y):= x^2+(3*y)^2+27
>plot3d("f"):
```



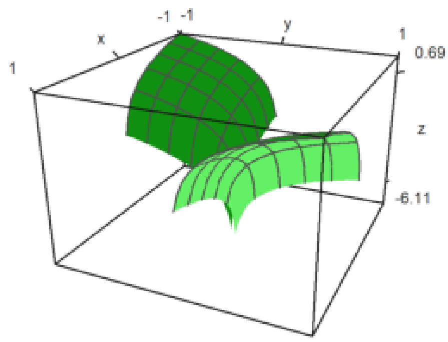
- Perintah "function f(x,y):= x^2+(3*y)^2+27" berarti mendefinisikan fungsi matematika f(x,y) sebagai x pangkat 2 ditambah 3 kali y pangkat 2 ditambah 27.
- Perintah "plot3d("f")" berarti membuat grafik tiga dimensi dari fungsi f(x,y) yang telah didefinisikan sebelumnya.

---

3. Fungsi Logaritma Dua Variabel

$$f(x, y) = \log(2xy)$$

```
>function f(x,y):= log((2*x)*y)
>plot3d("f"):
```



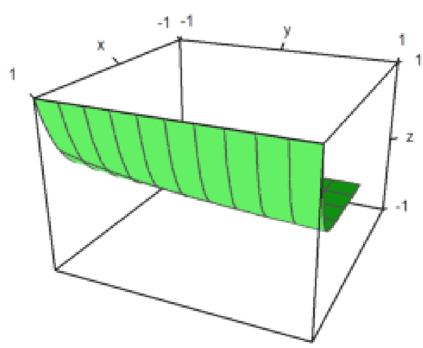
- Input yang diberikan adalah fungsi matematika dua variabel, f(x,y), yang didefinisikan sebagai logaritma hasil kali 2x dan y.
- Perintah "plot3d("f")" digunakan untuk memplot grafik fungsi f(x,y) dalam ruang tiga dimensi.

---

4. Fungsi Eksponen Dua Variabel

$$f(x, y) = x^{5y+10}$$

```
>function f(x,y):= x^(5*y+10)
>plot3d("f"):
```



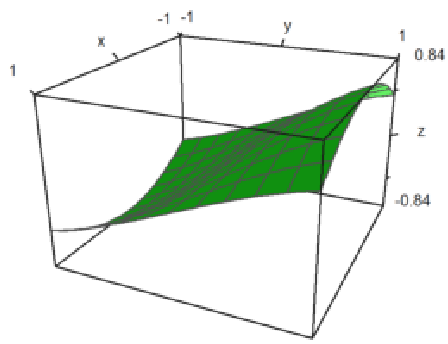
- Perintah ‘fungsi f(x,y):= x^(5y+10)‘ adalah fungsi matematika dua variabel ‘x‘ dan ‘y‘ dan dengan rumus x^(5y+10)
- Perintah ‘plot3d("f")‘ digunakan untuk memplot fungsi dalam ruang tiga dimensi. Plot yang dihasilkan akan menampilkan nilai fungsi sebagai permukaan pada bidang x-y, dengan tinggi permukaan mewakili nilai fungsi pada titik tersebut.

---

5. Fungsi Trigonometri Dua Variabel

$$f(x,y) = cos(x) + sin(y)$$

```
>function f(x,y):= cos(x)*sin(y)
>plot3d("f"):
```



- Perintah "function f(x,y):= cos(x)*sin(y)" adalah perintah untuk mendefinisikan fungsi matematika f(x,y) yang menghasilkan nilai cosinus dari x dikalikan dengan sinus dari y.
- Perintah "plot3d("f")" adalah perintah untuk membuat grafik tiga dimensi dari fungsi f(x,y) yang telah didefinisikan sebelumnya.

4. Menggambar Grafik Fungsi Dua Variabel yang

* Fungsinya Didefinisikan sebagai Fungsi Simbolik

Fungsi Simbolik

Fungsi simbolik adalah fungsi yang dinyatakan dengan menggunakan simbol-simbol matematika, seperti huruf dan operasi matematika, daripada menggunakan angka konkret atau ekspresi numerik. Fungsi simbolik sering digunakan untuk menggambarkan hubungan antara variabel-variabel matematika dalam bentuk yang lebih umum dan abstrak.
Contoh fungsi simbolik yang umum adalah:

$$g(x,y) = 2x + y$$

Dalam contoh di atas, g(x) adalah fungsi simbolik yang mengaitkan setiap nilai x dengan hasil dari ekspresi matematika 2x + 3. Fungsi ini dapat digunakan untuk menghitung nilai fungsi untuk berbagai nilai x.

Perbedaan Fungsi Numerik dan Fungsi Simbolik

1. Fungsi Numerik
Fungsi numerik dinyatakan dalam bentuk yang lebih konkret menggunakan angka-angka nyata.
Contoh fungsi numerik adalah

$$g(x,y) = 2x + y + 3$$

dimana kita memberikan nilai numerik kepada "x dan y"
misalnya, x = 5 dan y = 2, maka hasilnya adalah angka konkret yaitu g(5,2) = 15
2. Fungsi Simbolik
Fungsi simbolik dinyatakan menggunakan simbol-simbol matematika seperti huruf (variabel) dan operasi matematika.
Contoh fungsi simbolik adalah

$$g(x,y) = 2x + y + 3$$

"g" adalah simbol fungsi
"x,y" adalah variabel,
2x + 3 adalah ekspresi matematika yang menggambarkan hubungan antara "x,y" dan hasil fungsi.

Gambar Grafik Fungsi

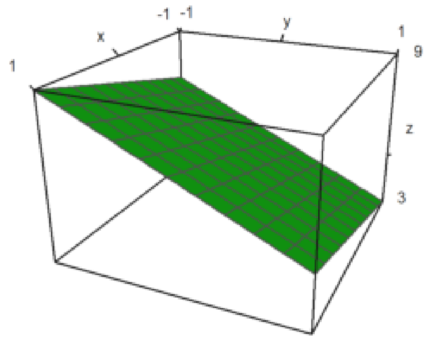
Fungsi satu baris simbolik didefinisikan oleh "&=".
Langkah-langkah untuk memvisualisasikan grafik fungsi dua variabel yang fungsi nya didefinisikan sebagai fungsi simbolikdalam plot3d:
1. Buat fungsi simbolik yang akan digunakan untuk memvisualisasikan data.
function g(x,y):= ax+by;
dimana a dan b adalah konstanta
2. Gunakan fungsi plot3d() untuk membuat grafik tiga dimensi dari fungsi numerik.
plot3d("g"):
3. Menentukan rentang variabel
misal
plot3d("g(x,y)",-10,10,-5,5):
dengan batasan x dari -10 hingga 10 dan batasan y dari -5 hingga 5

Contoh

1. Fungsi Linear Dua Variabel

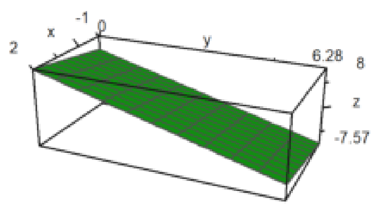
$$g(x,y) = x - 2y + 6$$

```
>function g(x,y)&= x-2*y+6;
>plot3d("g(x,y)"):
```



- Fungsi g(x,y) adalah fungsi matematika yang mengambil dua variabel, x dan y, dan menghasilkan sebuah nilai berdasarkan rumus x - 2y + 6.
- Perintah "plot3d" digunakan untuk menghasilkan grafik tiga dimensi dari fungsi tersebut.

```
>plot3d("g(x,y)",-1,2,0,2*pi):
```



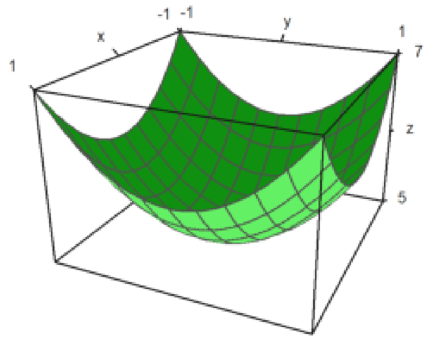
- Perintah "plot3d("g(x,y)",-1,2,0,2*pi)" adalah perintah untuk menggambar grafik fungsi tiga dimensi "g(x,y)" pada rentang x dari -1 hingga 2 dan rentang y dari 0 hingga 2pi.

---

2. Fungsi Kuadrat Dua Variabel

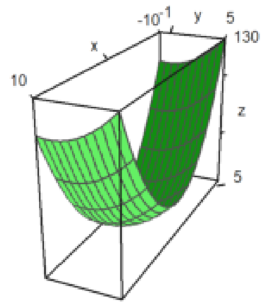
$$g(x, y) = x^2 + y^2 + 5$$

```
>function g(x,y)&= x^2+y^2+5;
>plot3d("g(x,y)"):
```



- Fungsi g(x,y) adalah fungsi matematika yang mengambil dua variabel, x dan y, dan menghasilkan sebuah nilai berdasarkan rumus x^2+y^2+5
- Perintah "plot3d" digunakan untuk menghasilkan grafik tiga dimensi dari fungsi tersebut.

```
>plot3d("g(x,y)",-10,10,-1,5):
```



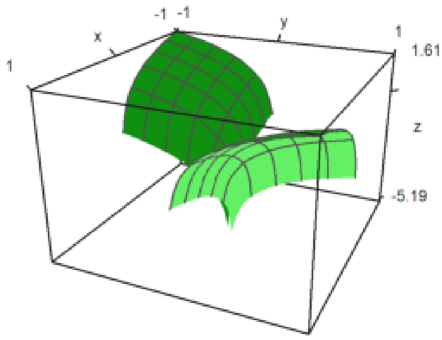
- Perintah "plot3d("g(x,y)",-10,10,-1,5)" adalah perintah untuk menggambar grafik fungsi tiga dimensi g(x,y) pada rentang x dari -10 hingga 10 dan rentang y dari -1 hingga 5

---

3. Fungsi Logaritma Dua Variabel

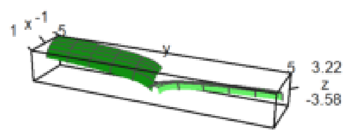
$$g(x,y) = \log(xy5)$$

```
>function g(x,y)&= log(x*y*5);
>plot3d("g(x,y)"):
```



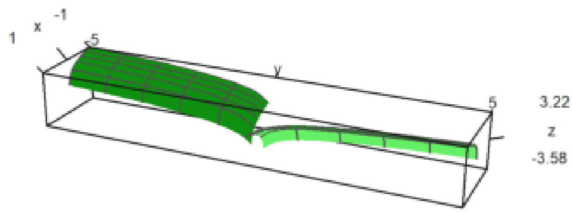
- Fungsi g(x,y) adalah fungsi matematika yang mengambil dua variabel, x dan y, dan menghasilkan sebuah nilai berdasarkan rumus logaritma x dikalikan y dikalikan 5
- Perintah "plot3d" digunakan untuk menghasilkan grafik tiga dimensi dari fungsi tersebut.

```
>plot3d("g(x,y)",-1,1,-5,5):
```



- Perintah "plot3d("g(x,y)",-1,1,-5,5)" adalah perintah untuk menggambar grafik fungsi tiga dimensi g(x,y) pada rentang x dari -1 hingga 1 dan rentang y dari -5 hingga 5

```
>plot3d("g(x,y)",-1,1,-5,5,zoom=4.5):
```



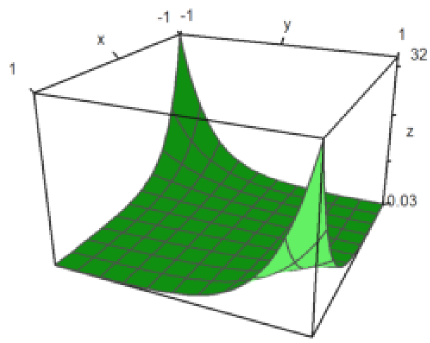
- plot3d: perintah untuk membuat grafik 3D.
- "g(x,y)": fungsi matematika yang akan digunakan untuk membuat grafik.
- -1,1: rentang nilai variabel x yang akan digunakan dalam grafik.
- -5,5: rentang nilai variabel y yang akan digunakan dalam grafik.
- zoom=4.5: perintah untuk memperbesar tampilan grafik dengan faktor 4.5.

---

4. Fungsi Eksponen Dua Variabel

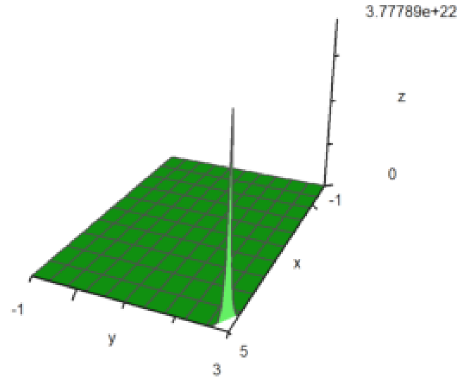
$$g(x,y) = 2^{xy5}$$

```
>function g(x,y)&= 2^(x*y*5);
>plot3d("g(x,y)"):
```



- Fungsi g(x,y) adalah fungsi matematika yang mengambil dua variabel, x dan y, dan menghasilkan sebuah nilai berdasarkan rumus 2^(xy5)
- Perintah "plot3d" digunakan untuk menghasilkan grafik tiga dimensi dari fungsi tersebut.

```
>plot3d("g(x,y)",-1,5,-1,3,frame=3,zoom=3):
```



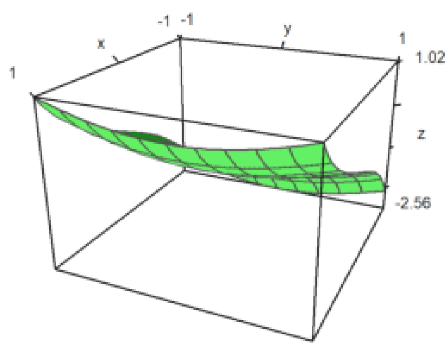
- Peintah plot3d("g(x,y)",-1,5,-1,3,frame=3,zoom=3) adalah perintah untuk membuat plot tiga dimensi dari fungsi ‘g(x,y)‘ dengan batas ‘x‘ dari ‘-1‘ hingga ‘5‘ dan batas ‘y‘ dari ‘-1‘ hingga ‘3‘.
- plot3d: perintah untuk membuat plot tiga dimensi.
- "g(x,y)"‘: fungsi yang akan diplot.
- (-1,5): batas ‘x‘ dari ‘-1‘ hingga ‘5‘.
- (-1,3): batas ‘y‘ dari ‘-1‘ hingga ‘3‘.
- frame=3: menampilkan frame nomor 3.
- zoom=3: memperbesar tampilan plot sebanyak 3 kali.

---

5. Fungsi Trigonometri Dua Variabel

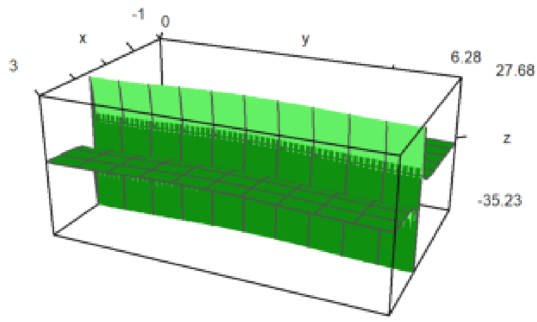
$$g(x,y) = tan(x) - cot(y)$$

```
>function g(x,y)&= tan(x)-cos(y);
>plot3d("g(x,y)"):
```



- Fungsi g(x,y) adalah fungsi matematika yang mengambil dua variabel, x dan y, dan menghasilkan sebuah nilai berdasarkan rumus tan(x)-cos(y)
- Perintah "plot3d" digunakan untuk menghasilkan grafik tiga dimensi dari fungsi tersebut.

```
>plot3d("g(x,y)",-1,3,0,2*pi,frame=1,zoom=3.5):
```



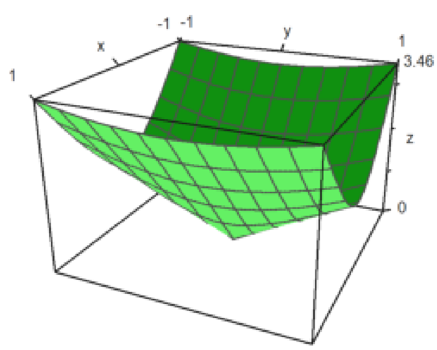
- Peintah plot3d("g(x,y)",-1,3,0,2*pi,frame=5,zoom=3) adalah perintah untuk membuat plot tiga dimensi dari fungsi ‘g(x,y)’ dengan batas ‘x‘ dari ‘-1‘ hingga ‘3‘ dan batas ‘y‘ dari ‘0‘ hingga ‘2pi‘.
- plot3d: perintah untuk membuat plot tiga dimensi.
- "g(x,y)"‘: fungsi yang akan diplot.
- (-1,3): batas ‘x‘ dari ‘-1‘ hingga ‘3‘.
- (0,2pi): batas ‘y‘ dari ‘0‘ hingga ‘2pi‘.
- frame=1: menampilkan frame nomor 1.
- zoom=3.5: memperbesar tampilan plot sebanyak 3.5 kali.

---

6. Fungsi Akar Kuadrat

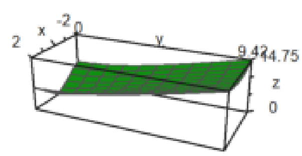
$$P(x,y) = \sqrt{10x^2 + 2y^2}$$

```
>function P(x,y) &= sqrt(10*x^2+2*y^2);
>plot3d("P(x,y)"):
```



- Fungsi P(x,y) adalah fungsi matematika yang mengambil dua variabel, x dan y, dan menghasilkan sebuah nilai berdasarkan rumus akar kuadrat dari 10x^2+2y^2
- Perintah "plot3d" digunakan untuk menghasilkan grafik tiga dimensi dari fungsi tersebut.

```
>plot3d("P(x,y)",-2,2,0,3*pi,frame=5,zoom=2,scale=1):
```



- P(x,y): Merupakan fungsi yang akan digambarkan dalam grafik tiga dimensi.
- (-2,2): Merupakan rentang nilai dari sumbu x yang akan digunakan dalam grafik.
- (0,3pi): Merupakan rentang nilai dari sumbu y yang akan digunakan dalam grafik. Nilai pi dikalikan dengan 3 agar rentang nilai y mencakup tiga putaran lingkaran penuh.
- frame=5: Menentukan nomor bingkai (frame) yang akan digunakan dalam animasi grafik.
- zoom=2: Menentukan faktor pembesaran grafik. Dengan memperbesar tampilan, kita dapat melihat detail yang lebih kecil pada plot.
- scale=1: Menentukan skala grafik. Dengan mengatur skala, kita dapat mengubah jarak antara titik-titik pada sumbu tersebut.

1. aspect(1.5) mengatur aspek rasio pada grafik 3D.
2. plot3d(expr,-5,5,-5,5) adalah fungsi matematika yang digunakan untuk membuat grafik 3D.
3. -5,5 mengatur rentang sumbu x yang akan ditampilkan pada grafik.
4. -5,5 mengatur rentang sumbu y yang akan ditampilkan pada grafik.

Menggambar Data x, y, z

* pada ruang Tiga Dimensi (3D)
Definisi

```
Menggambar data pada ruang tiga dimensi (3D) adalah proses
```

visualisasi data yang mengubah informasi dalam tiga dimensi, yaitu panjang, lebar, dan tinggi, menjadi representasi visual yang dapat dipahami dan dianalisis.
Tujuan:

```
Tujuan dari menggambar data 3D adalah untuk membantu pemahaman dan
```

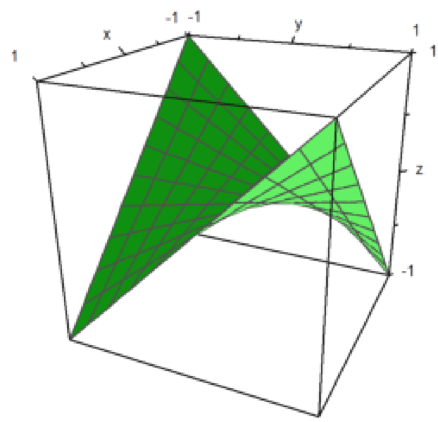
interpretasi data yang lebih baik, terutama ketika data tersebut memiliki komponen yang tidak dapat direpresentasikan dengan baik dalam dua dimensi.
Sama seperti plot2d, plot3d menerima data. Untuk objek 3D, Anda perlu menyediakan matriks nilai x-, y- dan z, atau tiga fungsi atau ekspresi fx(x,y), fy(x,y), 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.
Kita 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.

Contoh 1

```
>t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,t*s,grid=10):
```

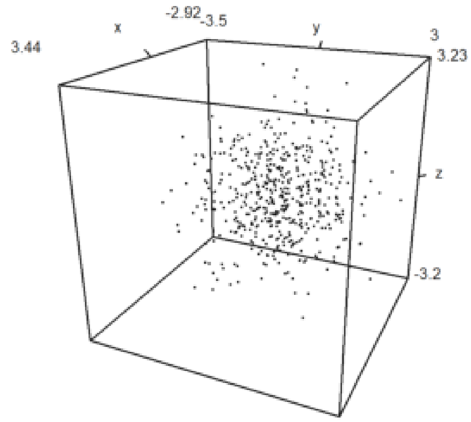


Baris pertama kode "t=-1:0.1:1" membuat vektor baris t yang berisi nilai dari -1 hingga 1 dengan interval 0.1. Baris kedua "s=(-1:0.1:1)′" membuat vektor kolom s yang berisi nilai dari -1 hingga 1 dengan interval 0.1. Operator transpose ′ digunakan untuk mengubah vektor baris t menjadi vektor kolom.
Baris ketiga "plot3d(t,s,ts,grid=10)" membuat plot tiga dimensi dari fungsi f(x,y) = xy pada domain [-1,1] x [-1,1]. Plot dibuat menggunakan fungsi plot3d, yang mengambil tiga argumen: koordinat x, y, dan z dari titik-titik yang akan diplot. Dalam hal ini, koordinat x diberikan oleh vektor t, koordinat y diberikan oleh vektor s, dan koordinat z diberikan oleh hasil perkalian t dan s, yaitu ts. Parameter grid diatur menjadi 10, yang menunjukkan jumlah garis grid yang akan ditampilkan pada plot.

Contoh 2

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="."):
```



Kode "n=500; plot3d(normal(1,n),normal(1,n),normal(1,n),points=true,style=".")" digunakan untuk membuat plot tiga dimensi dari tiga vektor normal yang dihasilkan secara acak dengan menggunakan fungsi "normal()" pada Euler Math Toolbox (EMT). Parameter "n=500" menunjukkan bahwa setiap vektor normal memiliki 500 elemen. Parameter "points=true" digunakan untuk menampilkan titik-titik pada plot, sedangkan parameter "style='.'" digunakan untuk mengatur gaya titik pada plot menjadi titik bulat.

Contoh 3

```
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):
```

Kode ini terdiri dari beberapa baris. Baris pertama "t=linspace(0,2pi,320)" membuat vektor t yang berisi 320 nilai yang sama terdistribusi secara merata antara 0 dan 2p. Baris kedua "s=linspace(-pi/2,pi/2,160)'" membuat vektor s yang berisi 160 nilai yang sama terdistribusi secara merata antara -p/2 dan p/2. Operator transpose ' digunakan untuk mengubah vektor baris s menjadi vektor kolom.

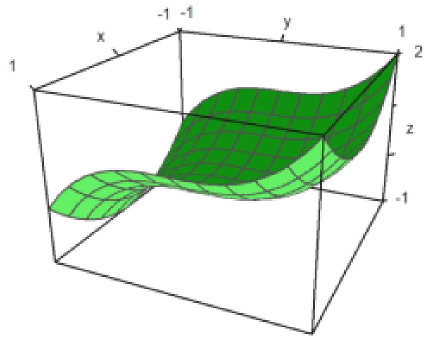
Baris ketiga "d=1+0.2*(cos(4t)+cos(8s))" membuat vektor d yang berisi nilai dari 1 + 0.2 * (cos(4t) + cos(8s)). Baris keempat "plot3d(cos(t)*cos(s)*d,sin(t)*cos(s)*d,sin(s)*d,hue=1,light=[1,0,1],frame=0,zoom=5)" membuat plot tiga dimensi dari fungsi f(x,y) = 2x^2 + y^3. Plot dibuat menggunakan fungsi plot3d, yang mengambil empat argumen: koordinat x, y, dan z dari titik-titik yang akan diplot, serta beberapa parameter lainnya. Dalam hal ini, koordinat x diberikan oleh ekspresi cos(t)*cos(s)*d, koordinat y diberikan oleh ekspresi sin(t)*cos(s)*d, dan koordinat z diberikan oleh ekspresi sin(s)*d. Parameter "hue=1" digunakan untuk mengatur warna pada plot berdasarkan nilai fungsinya. Parameter "light=[1,0,1]" digunakan untuk mengatur pencahayaan pada plot. Parameter "frame=0" digunakan untuk menghilangkan frame pada plot. Parameter "zoom=5" digunakan untuk mengatur level zoom pada plot. **Grafik Tiga Dimensi yang**

---

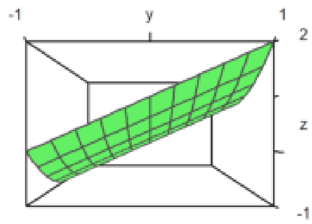
* Bersifat Interaktif dan animasi grafik 3D

Membuat gambar grafik tiga dimensi (3D) yang bersifat interaktif dan animasi grafik 3D adalah proses menciptakan visualisasi tiga dimensi yang memungkinkan pengguna berinteraksi dengan objek-objek 3D. Interaktivitas dalam gambar 3D memungkinkan pengguna untuk melakukan tindakan seperti mengubah sudut pandang, memindahkan objek, atau berinteraksi dengan elemen-elemen dalam adegan 3D. Animasi grafik 3D dapat mencakup pergerakan, tetapi juga dapat berarti perubahan dalam tampilan atau atribut objek tanpa pergerakan fisik yang mencolok.
CONTOH GAMBAR

```
>function testplot () := plot3d("x^2+y^3"); ...
>rotate("testplot"); testplot():
```

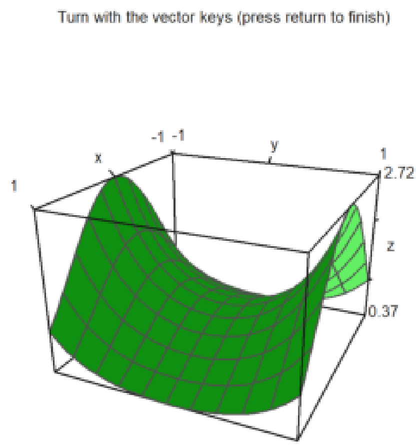



```
>function testplot () := plot3d("x^2+y",distance=3,zoom=1,angle=pi/2,height=0); ...
>rotate("testplot"); testplot():
```



Hilangkan command angle untuk bisa merotasikan grafik,dan height = 0 untuk membuat posisi sejajar dengan mata jadi tidak mempengaruhi pergerakan hanya berbeda sudut pandang saja

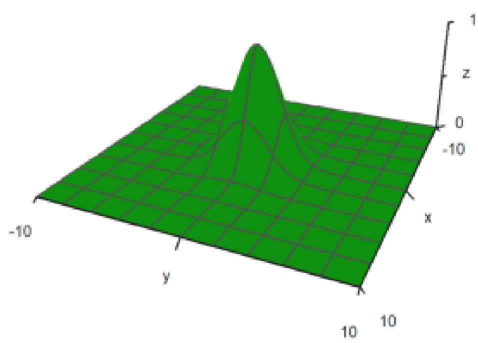
```
>plot3d("exp(-x^2+y^2)",>user, ...
>   title="Turn with the vector keys (press return to finish)"):
```




```
>plot3d("exp(x^2+y^2)",>user, ...
>title="Coba gerakan)")
```

Interaksi pengguna dimungkinkan dengan parameter. Pengguna dapat menekan tombol berikut.
1. kiri, kanan, atas, bawah: memutar sudut pandang
2. +,-: memperbesar atau memperkecil
3. a: menghasilkan anaglyph (lihat di bawah)
4. l: beralih memutar sumber cahaya (lihat di bawah)
5. spasi: disetel ulang ke default
6. enter: akhiri interaksi

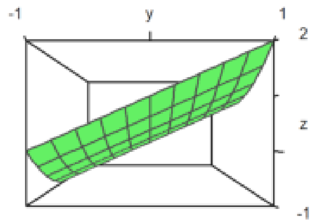
```
>plot3d("exp(-(x^2+y^2)/5)",r=10,n=80,fscale=4,scale=1.2,frame=3,>user):
```



Parameter "r=10" menunjukkan jari-jari bola yang digunakan untuk membuat plot tiga dimensi. Dalam hal ini, jari-jari bola yang digunakan adalah 10.
Parameter "n=80" menunjukkan jumlah titik yang digunakan untuk membuat plot. Semakin besar nilai n, semakin banyak titik yang digunakan untuk membuat plot, sehingga plot akan menjadi lebih halus dan akurat.
Parameter "fscale=4" menunjukkan faktor skala pada sumbu z. Dalam hal ini, faktor skala pada sumbu z adalah 4.
Parameter "scale=1.2" menunjukkan faktor skala pada plot. Semakin besar nilai scale, semakin besar ukuran

plot yang dihasilkan.
Parameter "frame=3" menunjukkan jenis frame yang digunakan pada plot. Dalam hal ini, jenis frame yang digunakan adalah frame kotak dengan sumbu x, y, dan z yang ditampilkan.

```
>plot3d("x^2+y",distance=3,zoom=1,angle=pi/2,height=0):
```



Tampilan dapat diubah dengan berbagai cara.
- distance: jarak pandang ke plot.
- zoom: nilai zoom.
- angle: sudut terhadap sumbu y negatif dalam radian.
- height: ketinggian tampilan dalam radian.

```
>plot3d("x^4+y^2",a=0,b=1,c=-1,d=1, angle=-20?, height=20?, ...
>  center=[0.4,0,0], zoom=5):
```

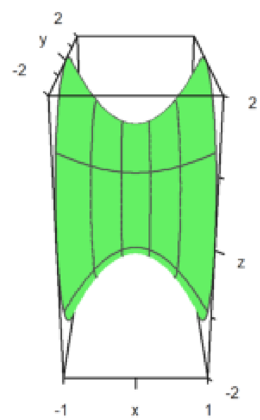
```
Closing bracket missing in function call!
Error in:
plot3d("x^4+y^2",a=0,b=1,c=-1,d=1, angle=-20?, height=20?,   c ...
                                            ^
```

Plot selalu terlihat berada di tengah kubus plot. Anda dapat memindahkan bagian tengah dengan parameter center.
Parameter center digunakan untuk memindahkan pusat plot ke lokasi tertentu dalam ruang. Dalam hal ini, pusat plot diatur ke titik (0.4, 0, 0) dalam ruang tiga dimensi. Parameter center berguna ketika kita ingin mengubah sudut pandang plot atau ketika kita ingin menyelaraskan plot dengan objek lain dalam scene. Dengan menentukan pusat plot, kita dapat mengontrol posisi kamera dan arah tampilan plot.
Ada beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.
fscale: menskalakan ke nilai fungsi (defaultnya adalah <fscale).
scale: angka atau vektor 1x2 untuk diskalakan ke arah x dan y.
frame: jenis bingkai (default 1).

```
>function testplot () := plot3d("5*exp(-x^2-y^2)",r=2,<fscale,<scale,distance=13,height=50
>center=[0,0,-2],frame=3); ...
>rotate("testplot"); testplot():
```

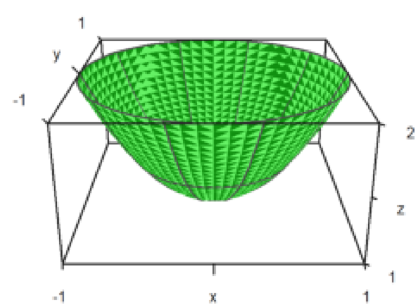
```
Closing bracket missing in function call!
testplot:
    useglobal; return plot3d("5*exp(-x^2-y^2)",r=2,<fscale,<scale ...
Try "trace errors" to inspect local variables after errors.
rotate:
    f$(args());
```

```
>plot3d("x^2+1",a=-1,b=1,rotate=true,grid=5):
```

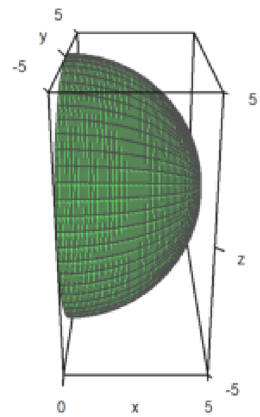


Penjelasan:
Secara umum, parameter "a" dan "b" digunakan untuk menentukan rentang nilai variabel independen dalam suatu fungsi. Dalam kasus ini, "a=-1" dan "b=1" menunjukkan bahwa fungsi tersebut akan diplot pada interval [-1, 1]. Parameter "rotate=true" menunjukkan bahwa grafik akan diputar untuk memberikan tampilan bentuk tiga dimensi yang lebih baik. Parameter "grid=5" menunjukkan bahwa grid dengan jarak 5 unit akan ditampilkan pada grafik.
Parameter memutar 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=2,grid=5):
```



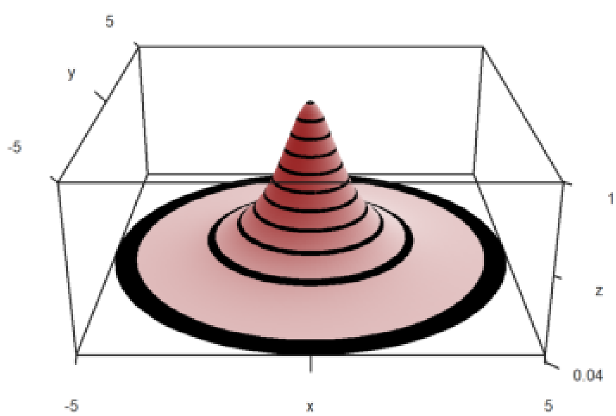
```
>function testplot () := plot3d("sqrt(25-x^2)",a=0,b=5,rotate=1); ...
>rotate("testplot"); testplot():
```




```
>function testplot () := plot3d("x^4+y^2",a=0,b=1,c=-1,d=1,height=20?, ...
>center=[0.4,0,0],zoom=5); ...
>rotate("testplot"); testplot():
```

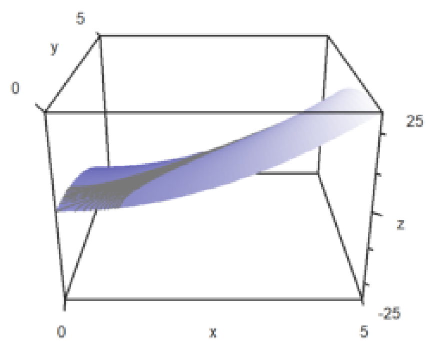
```
Closing bracket missing in function call!
testplot:
    useglobal; return plot3d("x^4+y^2",a=0,b=1,c=-1,d=1,height=20 ...
Try "trace errors" to inspect local variables after errors.
rotate:
    f$(args());
```

```
>function testplot () := plot3d("1/(x^2+y^2+1)",r=5,>polar, ...
>fscale=2,>hue,n=100,zoom=4,>contour,color=red); ...
>rotate("testplot"); testplot():
```



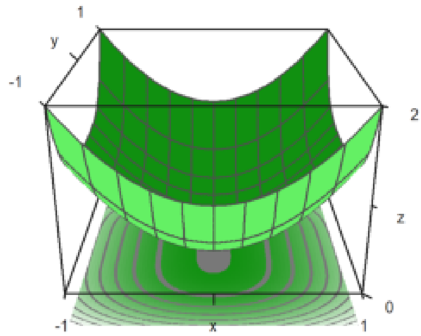
Parameter "r=5" menunjukkan jari-jari bola yang digunakan untuk membuat plot tiga dimensi. Dalam hal ini, jari-jari bola yang digunakan adalah 5.
Parameter ">polar" menunjukkan bahwa plot yang dibuat adalah plot polar tiga dimensi. Plot polar adalah plot yang dibuat dengan menggunakan koordinat polar, yaitu koordinat yang terdiri dari jarak dan sudut.
Parameter "fscale=2" menunjukkan faktor skala pada sumbu z. Dalam hal ini, faktor skala pada sumbu z adalah 2.
Parameter ">hue" menunjukkan bahwa warna pada plot akan diatur berdasarkan nilai fungsinya. Semakin tinggi nilai fungsinya, semakin terang warnanya.
Parameter "n=100" menunjukkan jumlah titik yang digunakan untuk membuat plot. Semakin besar nilai n, semakin banyak titik yang digunakan untuk membuat plot, sehingga plot akan menjadi lebih halus dan akurat.
Parameter "zoom=4" menunjukkan level zoom pada plot.
Parameter ">contour" menunjukkan bahwa garis kontur akan ditampilkan pada plot.
Parameter "color=blue" menunjukkan warna garis kontur pada plot. Dalam hal ini, warna yang digunakan adalah biru.
Untuk plotnya, Euler menambahkan garis grid. Sebaliknya dimungkinkan untuk menggunakan garis level dan satu warna atau warna spektral. Euler dapat menggambar ketinggian fungsi pada sebuah plot dengan bayangan. Di semua plot 3D, Euler dapat menghasilkan anaglyph merah/cyan.
-hue: Mengaktifkan bayangan cahaya, bukan kabel.
-contour: Membuat plot garis kontur otomatis pada plot.
-level=... (atau level): Vektor nilai garis kontur.

```
>function testplot () := plot3d("x^2-y^2",0,5,0,5,level=-1:0.1:1,color=blue); ...
>rotate("testplot"); testplot():
```



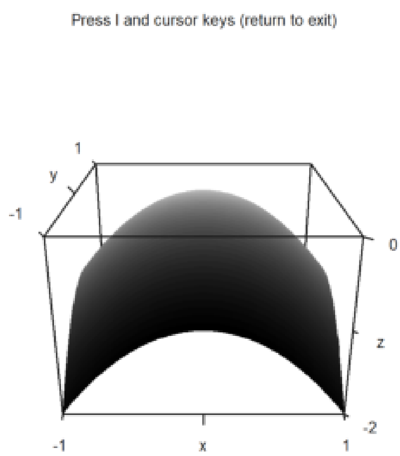
Parameter "level=-1:0.1:1" menunjukkan rentang nilai fungsinya yang akan ditampilkan pada plot. Dalam hal ini, rentang nilai fungsinya adalah dari -1 hingga 1 dengan interval 0.1.

```
>function testplot () := plot3d("x^2+y^4",>cp,cpcolor=green,cpdelta=0.2); ...
>rotate("testplot"); testplot():
```



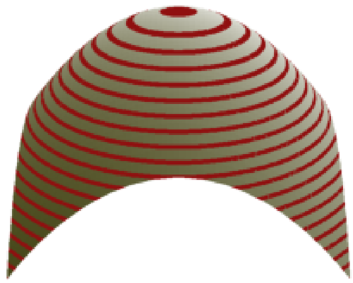
Parameter ">cp" menunjukkan bahwa titik kontrol akan ditambahkan pada plot. Titik kontrol digunakan untuk menentukan bentuk dan posisi plot tiga dimensi.
Parameter "cpcolor=green" menunjukkan warna titik kontrol yang akan digunakan. Dalam hal ini, warna yang digunakan adalah hijau.
Parameter "cpdelta=0.2" menunjukkan jarak antara titik kontrol. Semakin kecil nilai cpdelta, semakin banyak titik kontrol yang akan ditambahkan pada plot.

```
>plot3d("-x^2-y^2", ...
>hue=true,light=[0,1,1],amb=0,user=true, ...
> title="Press l and cursor keys (return to exit)"):
```



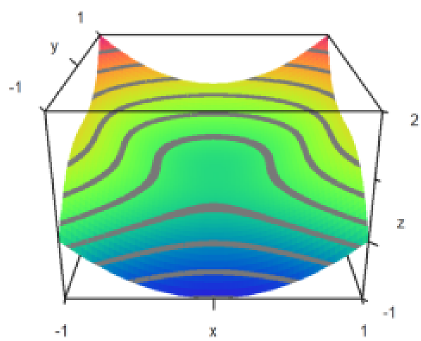
parameter "hue=true" menunjukkan bahwa warna pada plot akan diatur berdasarkan nilai fungsinya. Semakin tinggi nilai fungsinya, semakin terang warnanya.
Parameter "light=light=[0,1,1] menunjukkan intensitas cahaya pada plot. Nilai light=[0,1,1] menunjukkan bahwa cahaya datang dari arah positif y dan z.
Parameter "amb=0" menunjukkan intensitas cahaya ambient pada plot. Nilai 0 menunjukkan bahwa tidak ada cahaya ambient yang digunakan.

```
>function testplot () := 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); ...
>rotate("testplot"); testplot():
```

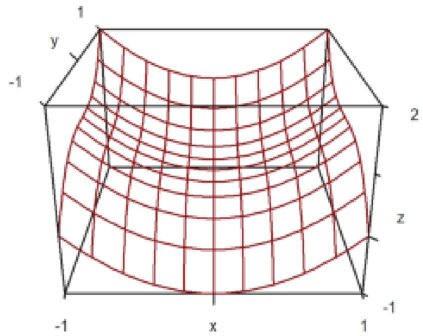


Parameter "frame=false" digunakan untuk menghilangkan frame pada plot tiga dimensi. Parameter "color=rgb(0.2,0.2,0)" menunjukkan warna dasar plot. Dalam hal ini, warna yang digunakan adalah hitam dengan nilai RGB (0.2, 0.2, 0). Parameter "dl=0.01" menunjukkan jarak antara titik-titik pada plot. Semakin kecil nilai dl, semakin banyak titik yang digunakan untuk membuat plot, sehingga plot akan menjadi lebih halus dan akurat. Namun, semakin kecil nilai dl, semakin lama waktu yang dibutuhkan untuk membuat plot.

```
>function testplot () := plot3d("x^2+y^3",>contour,>spectral); ...
>rotate("testplot"); testplot():
```



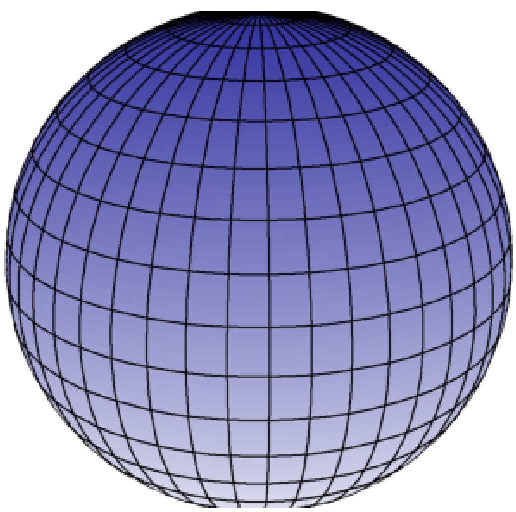

```
>function testplot () := plot3d("x^2+y^3", >transparent, grid=10, wirecolor=red); ...
>rotate("testplot"); testplot():
```



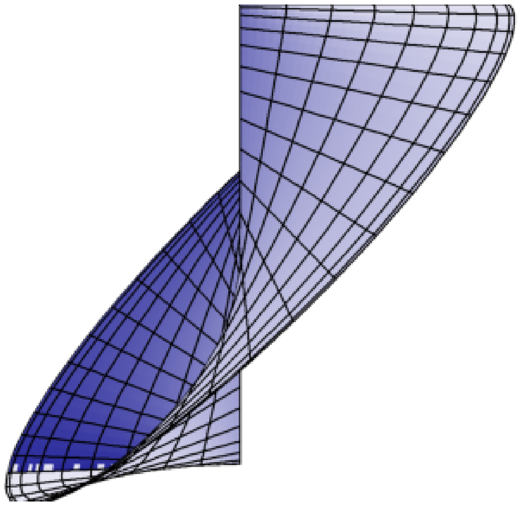
Fungsi Parametrik 3D

Fungsi parametrik merupakan jenis fungsi matematika yang menggambarkan hubungan antara dua atau lebih variabel, dimana masing-masing koordinat (x, y, z...) dinyatakan sebagai fungsi lain dari beberapa parameter. Fungsi parametrik dapat digunakan untuk menggambar kurva, lintasan, atau hubungan antara berbagai variabel yang bergantung pada parameter-parameter tertentu.
Sebagai contoh :

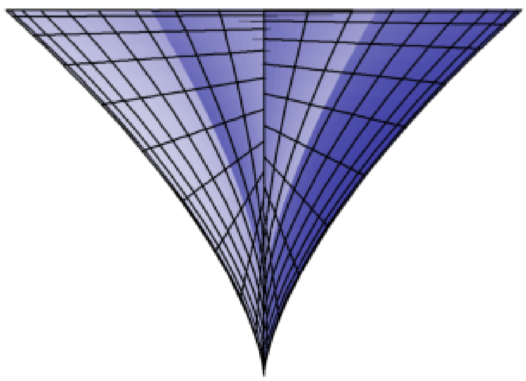
```
>plot3d("cos(x)*cos(y)","sin(x)*cos(y)","sin(y)", a=0,b=2*pi,c=pi/2,d=-pi/2,...
>>hue,color=blue,light=[0,1,3],<frame,...
>n=90,grid=[20,50],wirecolor=black,zoom=5):
```



```
>plot3d("cos(x)*cos(y)","sin(x)*cos(y)","cos(x)", a=0,b=2*pi,c=pi/2,d=-pi/2,...
>>hue,color=blue,light=[0,1,3],<frame,...
>n=90,grid=[20,50],wirecolor=black,zoom=5):
```



```
>plot3d("cos(x)^3*sin(y)","sin(x)^2*sin(y)","cos(x)^2", a=0,b=2*pi,c=pi/2,d=-pi/2,...
>>hue,color=blue,light=[0,1,5],<frame,...
>n=90,grid=[20,50],wirecolor=black,zoom=5):
```



8 Menggambar Fungsi Implisit Implisit

Fungsi implisit (implicit function) adalah fungsi yang memuat lebih dari satu variabel, berjenis variabel bebas dan variabel terikat yang berada dalam satu ruas sehingga tidak bisa dipisahkan pada ruas yang berbeda.

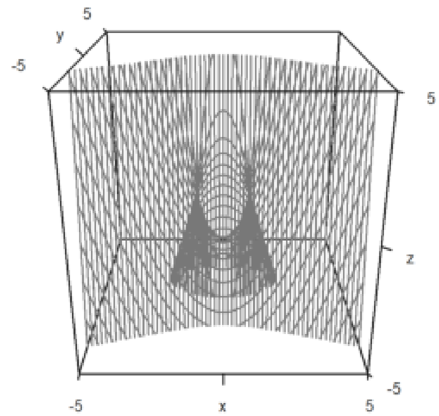
$$F(x, y, z) = 0$$

(1 persamaan dan 3 variabel), terdiri dari 2 variabel bebas dan 1 terikat
Misalnya,

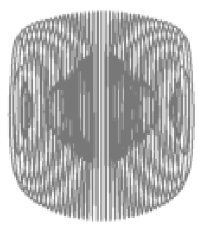
$$F(x, y, z) = x^2 + y^2 + z^2 = 1$$

adalah persamaan implisit yang menggambarkan bola dengan jari-jari 1 dan pusat di (0,0,0).

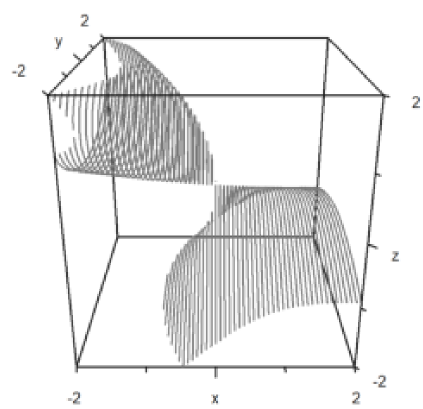
```
>plot3d("x^2+y^3+z*y-1", r=5, implicit=3):
```




```
>c=1; d=1;
>plot3d("((x^2+y^2-c^2)^2+(z^2-1)^2)*((y^2+z^2-c^2)^2+(x^2-1)^2)*((z^2+x^2-c^2)^2+(y^2-1)^
```



```
>plot3d("x^2+y^2+4*x*z+z^3",>implicit, r=2, zoom=2.5):
```



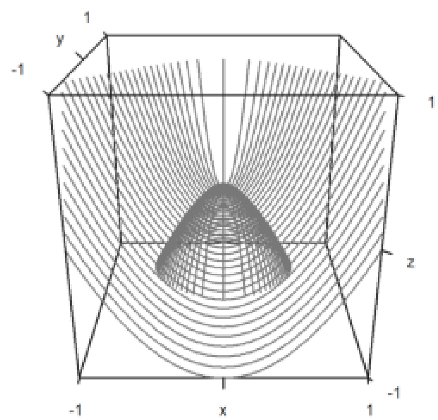
Selain plot kontur yang sudah di jelaskan sebelumnya, pada EMT juga ada plot umplisit dalam tiga dimensi. Euler menghasilkan potongan melalui objek. Fitur plot3d termasuk plot implisit. Plot-plot ini menunjukkan himpunan nol dari sebuah fungsi dalam tiga variabel.
Solusi dari

$$f(x, y, z) = 0$$

dapat divisualisasikan dalam potongan yang sejajar dengan bidang x-y, bidang x-z, dan bidang y-z.
- implisit = 1: potong sejajar dengan bidang-y-z
- implicit = 2: memotong sejajar dengan bidang x-z
- implicit=4: memotong sejajar dengan bidang x-y
Ambil contoh dari persamaan latex pada fungsi implisit tadi dan tambahkan nilai-nilai ini, sehingga kita dapat memplot persamaan ini

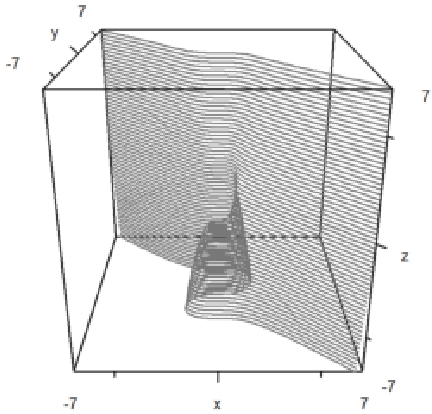
$$M = (x, y, z) : x^2 + y^3 + zy = 1$$

```
>plot3d("x^2+y^3+z*y", r=1, implicit=2):
```

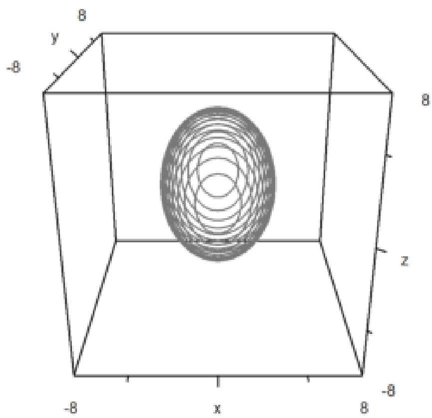


Contoh fungsi implisit yang lainnya

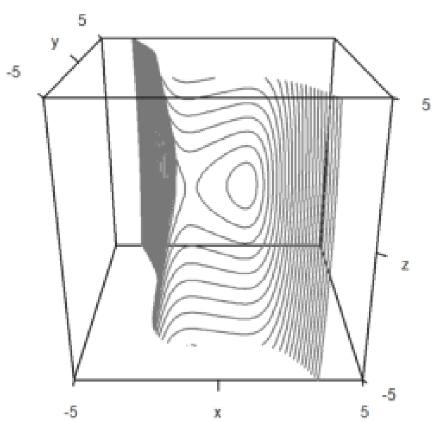
```
>plot3d("x^3+y^3+z*y-1",r=7,implicit=4):
```



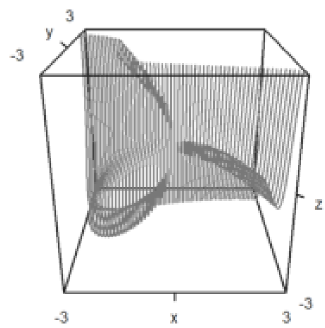

```
>plot3d("2*x^2 + 3*y^2 + z^2 - 25",r=8,implicit=2):
```



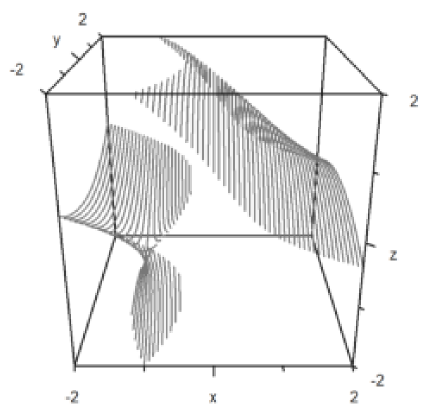

```
>plot3d("x^5 + 5*y^3 + 3*z^2 - 5*x - 7*y - 5*z + 10",r=5,implicit=2):
```



```
>plot3d("x^3+y^5+5*x*z+z^3",>implicit,r=3,zoom=2):
```




```
>plot3d("x^2+y^2+4*x*z+z^3-5",>implicit,r=2,zoom=2.5):
```



Fungsi Implisit Menggunakan Povray

Povray dapat memplot himpunan di mana f(x,y,z)=0, seperti parameter implisit di plot3d. Namun, hasilnya terlihat jauh lebih baik.
Sintaks untuk fungsi-fungsi tersebut sedikit berbeda. Anda tidak dapat menggunakan output dari ekspresi Maxima atau Euler.

$$((x^2+y^2-c^2)^2+(z^2-1)^2)*((y^2+z^2-c^2)^2+(x^2-1)^2)*((z^2+x^2-c^2)^2+(y^2-1)^2)=d$$

```
>load povray;
>defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

```
C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe
```

```
>povstart(angle=70°,height=50°, zoom=4);
>writeln(povsurface("pow(x,2)*y-pow(y,3)-pow(z,2)",povlook(blue)));
>writeAxes();
>povend();
```

```
Command was not allowed!
exec:
    return _exec(program,param,dir,print,hidden,wait);
povray:
    exec(program,params,defaulthome);
Try "trace errors" to inspect local variables after errors.
povend:
    povray(file,w,h,aspect,exit);
```

BAB 4

KB PEKAN 6-7: MENGGUNAKAN EMT UNTUK KALKULUS

[a4paper,10pt]article eumat

Nama : Fransisca Renita Pejoresa
NIM : 22305144012
Kelas : Matematika E 2022

Bab Kalkulus

BAB 4. INTERGAL TAK TENTU (Antiderivatif)

CAKUPAN MATERI MELIPUTI DIANTARANYA:
-Defini Integral tak tentu
-Sifat- sifat integral tak tentu
-Integral tak tentu fungsi aljabar, trigonometri, eksponensial, logaritma, dan komposisi fungsi
-Visualisasi dan kurva fungsi
1. Definisi Intergal Tak Tentu

```
Integral tak tentu (indefinite integral) adalah integral yang
```

tidak memiliki batas-batas nilai tertentu, sehingga hanya diperoleh fungsi umumnya saja disertai suatu konstanta C.

```
Misalkan diketahui suatu fungsi F(x) yang merupakan fungsi umum
```

yang bersifat F'(x)=f(x), maka integral tak tentu merupakan himpunan anti turunan F(x) dari f(x) pada interval negatif tak hingga sampai tak hingga yang dinotasikan :

$$F(x) = \int f(x)\, dx + C$$

```
>$F(x)=('integrate(f(x),x)+c)
```

$$F\left(x\right)=\int {f\left(x\right)\;dx}+c$$

Definisi kurva fungsi antiderifatif

```
Kurva fungsi antiderivatif adalah kurva yang menggambarkan
```

hubungan antara suatu fungsi dan antiderivatifnya. Antiderivatif, juga dikenal sebagai integral tak tentu. Dalam integral, fungsi antiderivatif dapat dianggap sebagai "anti turunan" dari fungsi aslinya.
Contoh :

$$\int 3x^2 dx$$

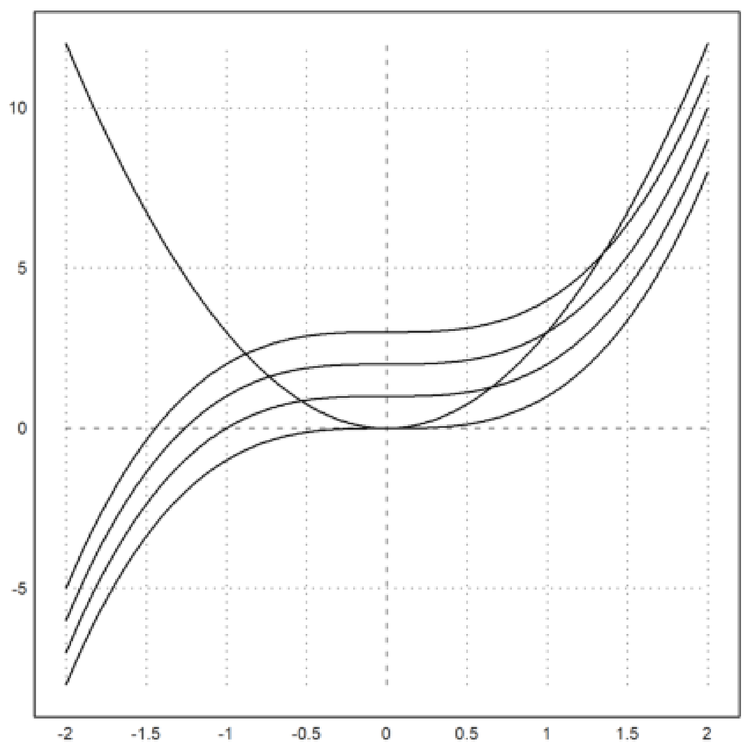
```
>$F(x)=('integrate(3*x^2,x)+c)
```

$$F\left(x\right)=3\,\int {x^2\;dx}+c$$

```
>$showev('integrate(3*x^2,x)+c)
```

$$3\,\int {x^2\;dx}+c=x^3+c$$

```
>plot2d(["3*x^2","x^3","x^3+1","x^3+2","x^3+3"]): //grafik fungsinya, hasil integral, pena
```



Penyelesaiannya dengan memasukan sebarang nilai C

$$\int x^5 dx$$

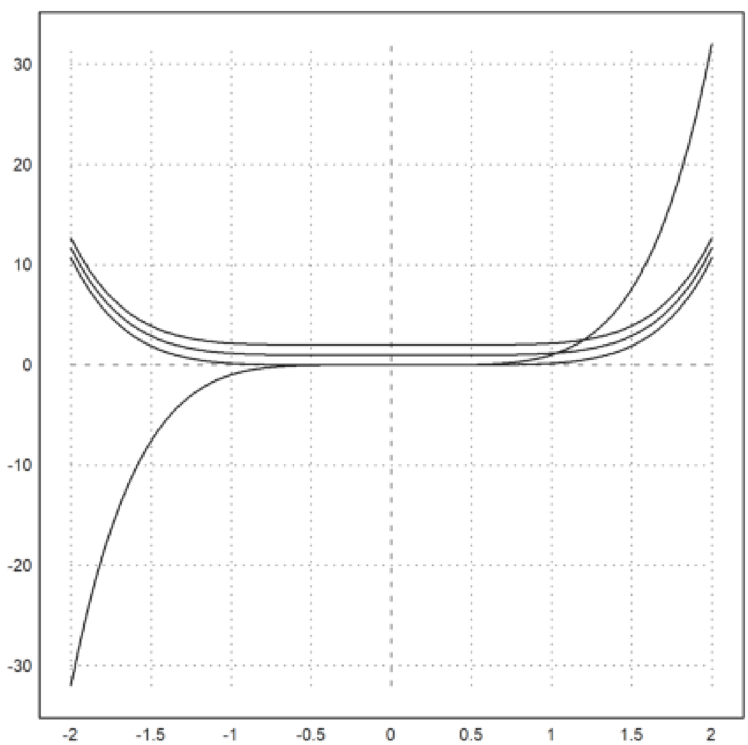
```
>$F(x)=('integrate(x^5,x) +c)
```

$$F\left(x\right)=\int {x^5\;dx}+c$$

```
>$showev('integrate(x^5,x)+c)
```

$$\int {x^5\;dx}+c=\frac{x^6}{6}+c$$

```
>plot2d(["x^5","x^6/6","(x^6/6)+1","(x^6/6)+2"]):
```



---

2. Sifat-sifat Integral Tak Tentu

Dalam perhitungan, integral tak tentu memiliki sifat-sifat yang

dapat digunakan. Ada tiga sifat integral tak tentu yaitu sebagai berikut:

a. Sifat Pangkat

$$\int x^n dx + c = \frac{x^n+1}{n+1} + c$$

```
>$showev('integrate(x^n,x)+c)
```

Answering "Is n equal to -1?" with "no"

$$\int x^n\, dx + c = \frac{x^{n+1}}{n+1} + c$$

b. Penjumlahan dan Pengurangan

$$\int [f(x) \pm g(x)]dx = \int f(x)dx \pm \int g(x)dx$$

```
>function f(x) &&= f(x)
```

f(x)

```
>function g(x) &&= g(x)
```

g(x)

Penjumlahan

```
>$('integrate([f(x)+g(x)],x))=('integrate(f(x),x))+('integrate(g(x),x))
```

$$\int [g(x) + f(x)]\, dx = \int g(x)\, dx + \int f(x)\, dx$$

Pengurangan

```
>$('integrate([f(x)+g(x)],x))=('integrate(f(x),x))-('integrate(g(x),x))
```

$$\int [g(x) + f(x)]\, dx = \int f(x)\, dx - \int g(x)\, dx$$

c. Konstanta

$$\int k.f(x)dx = k \int f(x)dx$$

```
>$('integrate(kf(x),x))=(k*'integrate(f(x),x))
```

$$\int kf\left(x\right)\,dx=k\,\int f\left(x\right)\,dx$$

---

3. INTERGAL TAK TENTU FUNGSI ALJABAR

```
A. Defnisi
  Integral tak tentu fungsi aljabar merupakan sebuah operasi
```

matematika yang menghasilkan fungsi lain yang turunan parsialnya akan sama dengan fungsi asal. Dalam konteks fungsi aljabar, integral tak tentu biasanya melibatkan fungsi-fungsi seperti polinomial, eksponensial, dan trigonometri, dan menghasilkan fungsi yang mewakili daerah di bawah kurva fungsi asal terhadap variabel independen.

```
B. Rumus-rumus integral fungsi aljabar
```

Bentuk pertama

$$\int dx = x + C$$

Dalam bentuk pertama bukan berarti tidak ada konstanta yang terlibat dalam integral tak tentu, tapi ada konstanta yaitu angka 1, di dalam matematika biasanya angka 1 sebagai konstanta tidak dituliskan.
Bentuk kedua

$$\int kdx = kx + C$$

k merupakan konstanta yang berupa sebarang bilangan.
Bentuk ketiga

$$\int kx^n dx = \frac{k}{n+1}x^{n+1} + C$$

k dan n merupakan sebarang bilangan bulat, dengan k adalah konstanta dan n adalah pangkat dari x dengan syarat n tidak sama dengan -1.
Bentuk keempat

$$\int k.f(x)dx = k\int f(x)dx$$

```
Dengan k merupakan sebarang bilangan bulat.
```

Bentuk kelima

$$\int (f(x) \pm g(x))dx = \int f(x)dx \pm \int g(x)dx$$

Bentuk keenam

$$\int k(ax+b)^n dx = \frac{k}{a(n+1)}(ax+b)^{n+1} + C$$

Dalam bentuk integral keenam hanya berlaku jika angka pada pangkat x

adalah 1.

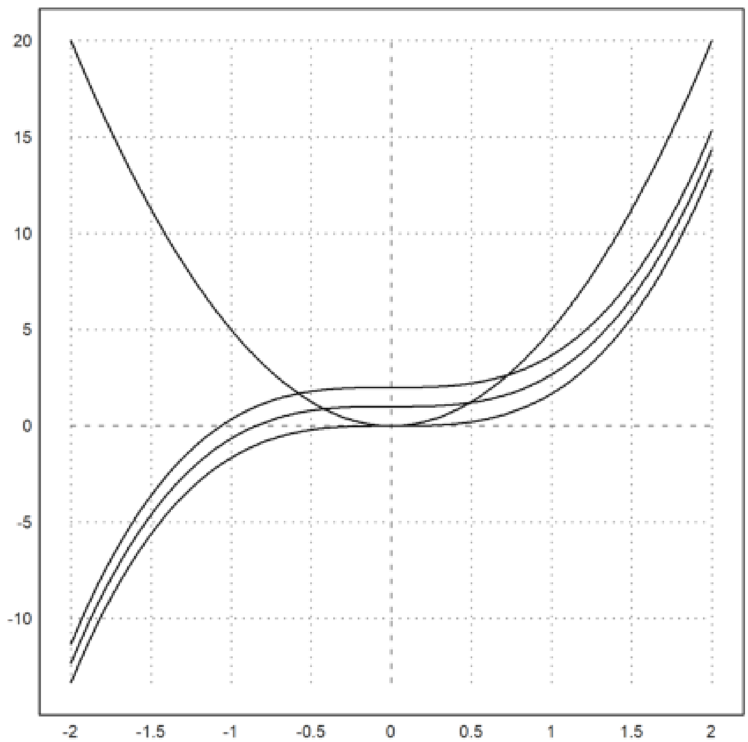
C. Contoh Soal & Kurva

$$\int 5x^2 dx$$

```
>$showev('integrate(5*x^2,x)+c)
```

$$5\int x^2\, dx + c = \frac{5\,x^3}{3} + c$$

```
>plot2d(["5*x^2","(5*x^3/3)","(5*x^3/3)+1","(5*x^3/3)+2"]):
```



$$f(x) = 4x + 2, g(x) = 2x + 1$$

```
>function f(x) &&= 4*x+2
```

```
4 x + 2
```

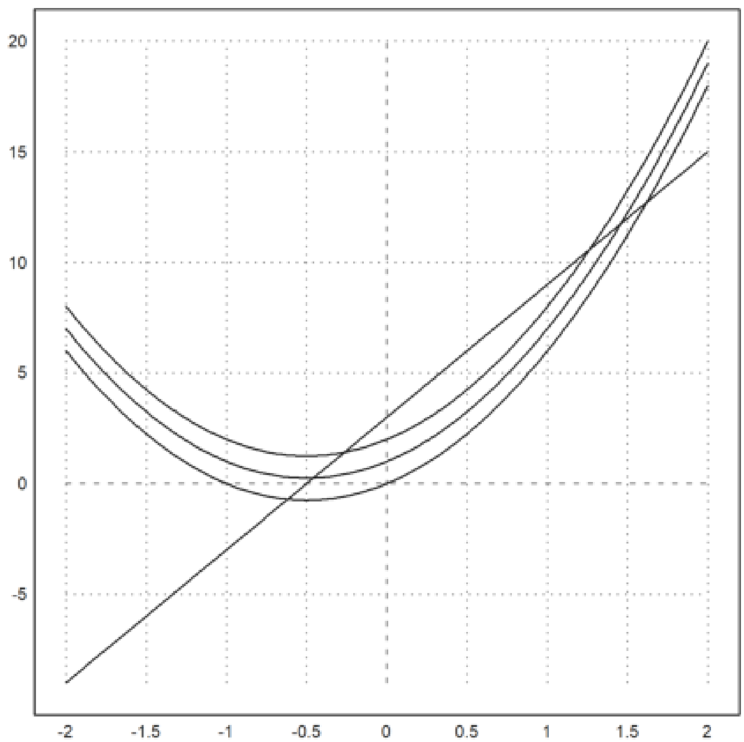
```
>function g(x) &&= 2*x+1
```

```
2 x + 1
```

```
>$showev('integrate(f(x)+(g(x)),x)+c)
```

$$\int 6\,x+3\;dx+c=3\,x^2+3\,x+c$$

```
>plot2d(["6*x+3","3*x^2+3*x","(3*x^2+3*x)+1","(3*x^2+3*x)+2"]):
```




```
>$showev('integrate(x*sqrt(x+2),x))
```

$$\int x\,\sqrt{x+2}\;dx=\frac{2\,\left(x+2\right)^{\frac{5}{2}}}{5}-\frac{4\,\left(x+2\right)^{\frac{3}{2}}}{3}$$

---

4. INTERGAL TAK TENTU FUNGSI NON ALJABAR (transenden)

```
4.1 Intergal Tak Tentu Fungsi Trigonometri
```

A. Defnisi
Integral tak tentu fungsi trigonometri merupakan operasi

matematika yang digunakan untuk mencari fungsi asal sebelumnya (biasanya ditambahkan dengan konstanta) yang ketika diambil turunan akan menghasilkan fungsi trigonometri tersebut.

Secara umum, integral tak tentu fungsi trigonometri seperti

sin(x), cos(x), atau tan(x) melibatkan berbagai rumus dan teknik integral yang berbeda tergantung pada jenis fungsi trigonometri yang terlibat.

B. Rumus-rumus integral fungsi trigonometri

image: trigonometri.png

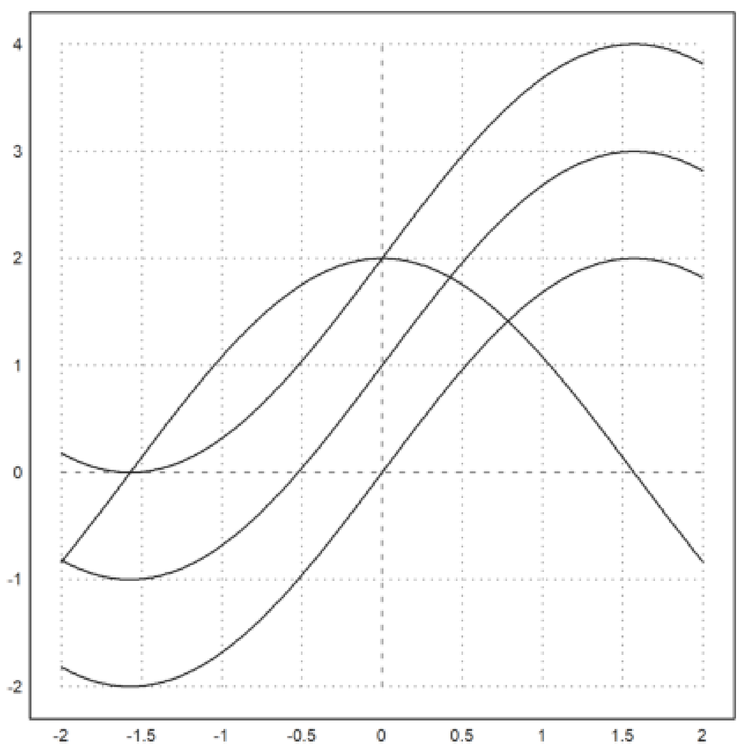
C. Contoh Soal & Kurva

$$\int 2cosxdx$$

```
>$showev('integrate(2*cos(x),x)+c)
```

$$2\int \cos x\,dx + c = 2\,\sin x + c$$

```
>plot2d(["2*cos(x)","2*sin(x)","2*sin(x)+1","2*sin(x)+2"]):
```

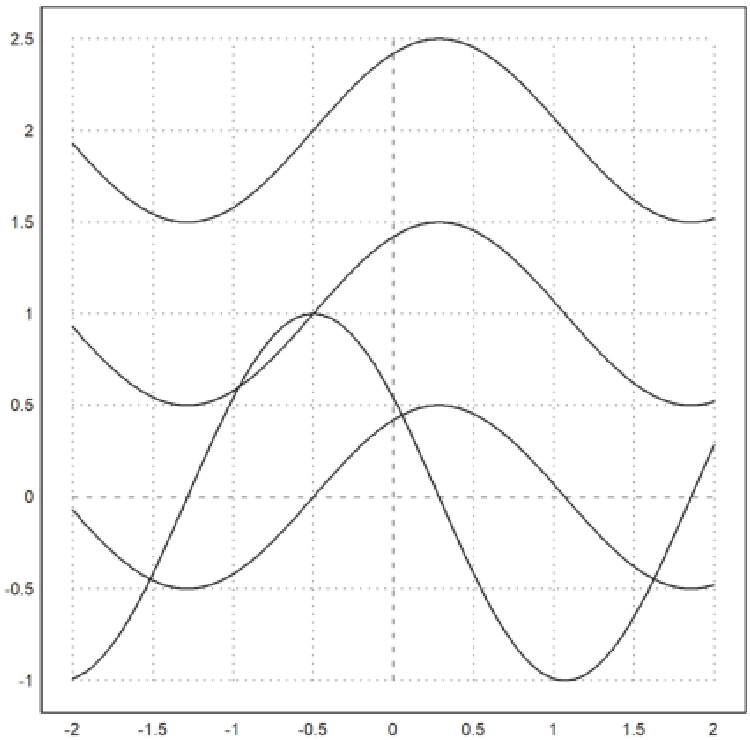


$$\int cos(2x+1)dx$$

```
>$showev('integrate(cos(2*x+1),x)+c)
```

$$\int \cos\left(2\,x+1\right)\,dx+c=\frac{\sin\left(2\,x+1\right)}{2}+c$$

```
>plot2d(["cos(2*x+1)","sin(2*x+1)/2","(sin(2*x+1)/2)+1","(sin(2*x+1)/2)+2"]):
```




```
4.2 Intergal Tak Tentu Fungsi Eksponensial
```

```
A. Defnisi
Integral dari fungsi eksponensial adalah operasi matematika yang
```

digunakan untuk menemukan area di bawah kurva fungsi eksponensial tertentu. Integral fungsi eksponensial merupakan proses untuk menemukan fungsi yang, ketika di turunkan, akan menghasilkan fungsi eksponensial tersebut.

```
B. Rumus-rumus integral fungsi eksponensial
```

Secara umum, integral dari fungsi eksponensial e^x adalah:

$$\int e^x dx = e^x + c$$

```
>$showev('integrate((E^x),x)+ c)
```

$$\int e^{x}\,dx+c=e^{x}+c$$

di mana "C" adalah konstanta integrasi. Ini berarti hasil dari integral ini adalah fungsi eksponensial e^x itu sendiri ditambah dengan konstanta integrasi.

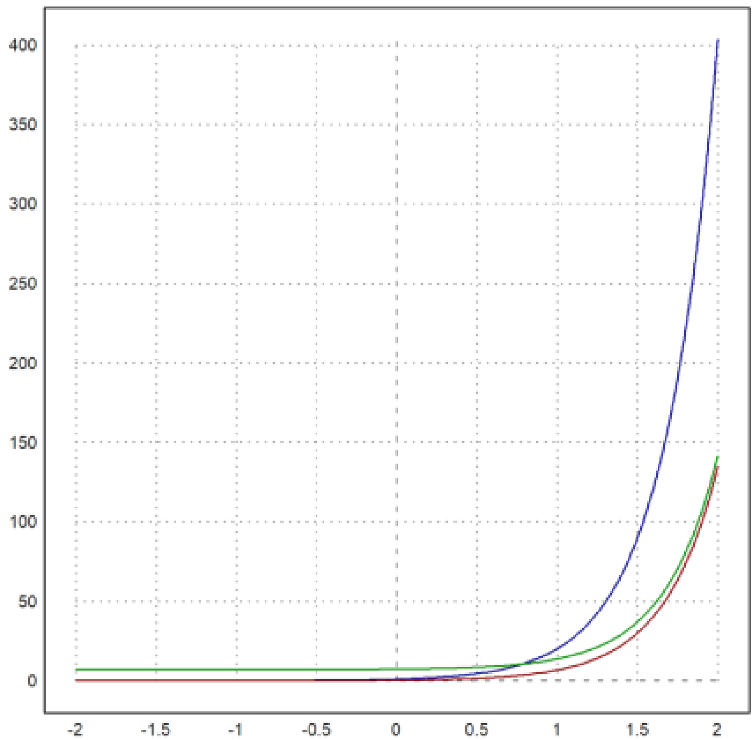
C. Contoh Soal & Kurva

$$\int e^{3}xdx$$

```
>$showev('integrate((E^x)^3,x)+c)
```

$$\int e^{3\,x}\,dx+c=\frac{e^{3\,x}}{3}+c$$

```
>plot2d(["(E^x)^3","((E^x)^3)/3","(((E^x)^3)/3)+7"],color=[blue,red,green]):
```

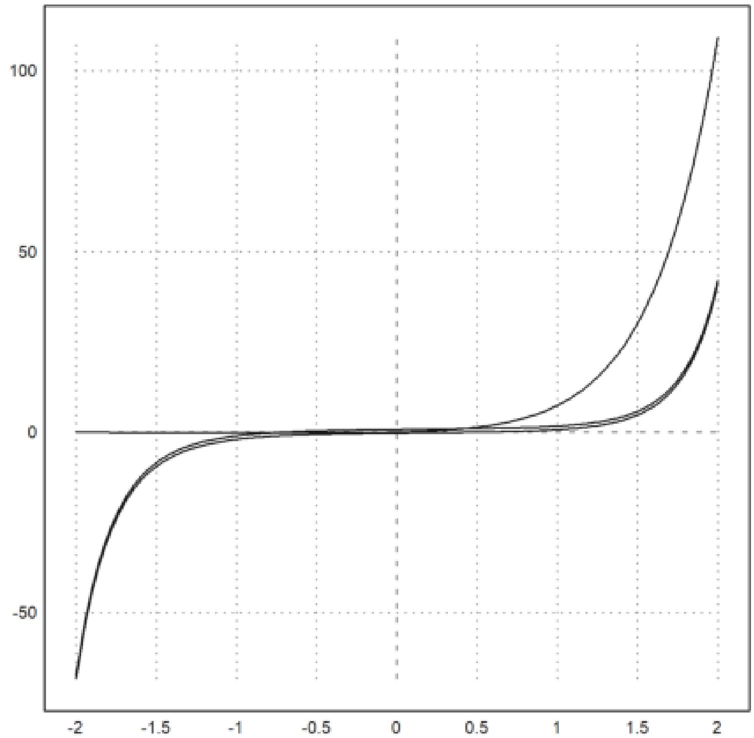


$$\int xe^{2x}dx$$

```
>$showev('integrate(x*(E^x)^2,x)+c)
```

$$\int x\, e^{2\, x}\, dx + c = \frac{(2\, x - 1)\, e^{2\, x}}{4} + c$$

```
>plot2d(["x*(E^x)^2","((2*x-1)E^x^2)/4","(2*x-1)E^x^2/4 +1"]):
```




```
4.3 Intergal Tak Tentu Fungsi Logaritma
      A. Defnisi
   Integral dari fungsi logaritma adalah operasi matematika yang
```

digunakan untuk menemukan area di bawah kurva fungsi logaritma tertentu.

```
      B. Rumus integral fungsi logaritma
```

$$\int log(x)dx = xlog(x) - x + C$$

```
>$showev('integrate(ln(x),x)+c)
```

$$\int \log x\, dx + c = x\, \log x - x + c$$

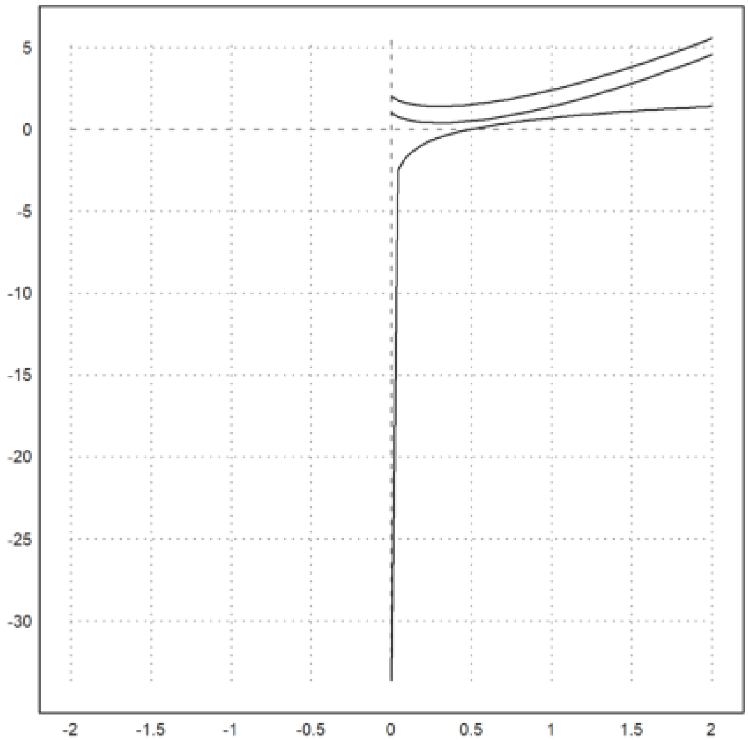
```
      C. Contoh Soal & Kurva
```

$$\int log(2x)dx$$

```
>$showev('integrate(log(2*x),x)+c)
```

$$\int \log\left(2\,x\right)\;dx+c=\frac{2\,x\,\log\left(2\,x\right)-2\,x}{2}+c$$

```
>plot2d(["log(2*x)","(2*x)log(2*x)-2*x/2 +1","(2*x)log(2*x)-2*x/2 +2"]):
```



---

5. INTERGAL TAK TENTU FUNGSI KOMPOSISI

```
A. Defnisi
 Integral tak tentu dari fungsi komposisi, juga dikenal sebagai
```

"integral tak tentu dari substitusi," adalah teknik integral yang digunakan untuk mengintegrasikan fungsi yang merupakan hasil dari komposisi dua fungsi.

```
B. Contoh Soal dan kurva
```

f(x)=x+1, g(x)=x+2

```
>function f(x) &= x+1
```

```
x + 1
```

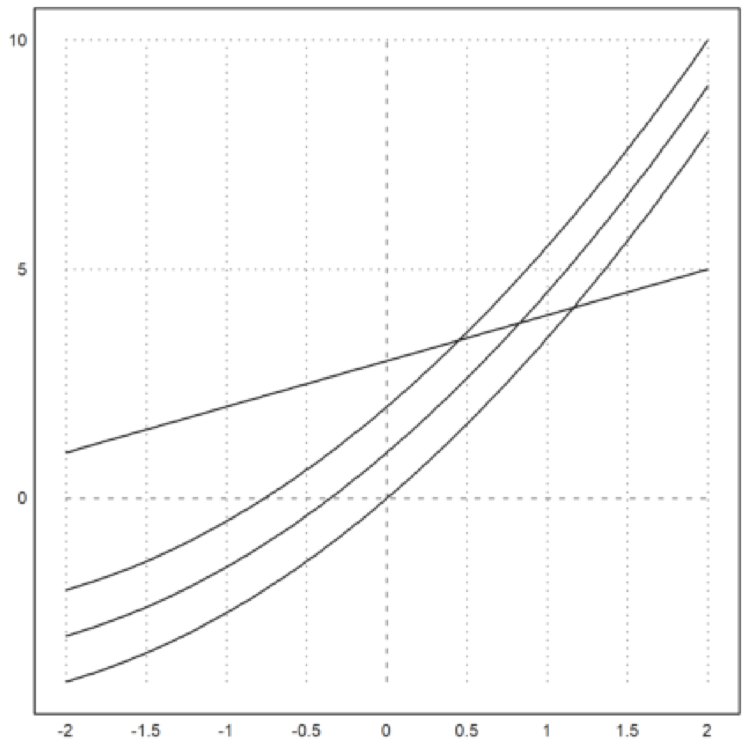
```
>function g(x) &= x+2
```

x + 2

```
>$showev('integrate(f(g(x)),x)+c)
```

$$\int 2\,x+2\,dx+c=x^2+2\,x+c$$

```
>plot2d(["x+3","((x^2)/2)+3*x","(((x^2)/2)+3*x)+1","(((x^2)/2)+3*x)+2"]):
```




```
>
>
```

2. LIMIT FUNGSI

Materi mencakup di antaranya:
1. Mendefinisikan Limit Fungsi pada EMT

```
1.1 Definisi LIMIT KIRI
1.2 Definisi LIMIT KANAN
```

2. LIMIT FUNGSI ALJABAR
3. LIMIT FUNGSI NON ALJABAR (transenden)

```
3.1 Limit Fungsi Trigonometri
3.2 Limit Fungsi Eksponensial
4.3 Limit Fungsi Logaritma
```

Definisi Limit

Dalam matematika, konsep limit digunakan untuk menjelaskan perilaku suatu fungsi saat peubah bebasnya mendekati suatu titik tertentu, atau menuju tak hingga; atau perilaku dari suatu barisan saat indeks mendekati tak hingga. Limit dipakai dalam kalkulus (dan cabang lainnya dari analisis matematika) untuk membangun pengertian kekontinuan, turunan dan integral.
Dalam pelajaran matematika, limit biasanya mulai dipelajari saat pengenalan terhadap kalkulus.

Limit Fungsi

Jika f(x) adalah fungsi real dan c adalah bilangan real, maka:

$$\lim_{x \to c} f(x) = L$$

Notasi tersebut menyatakan bahwa f(x) untuk niai x mendekati c sama dengan L. F(x) disini dapat berupa bermacam-macam jenis fungsi. Dan L dapat berupa konstanta, ataupun "und" (tak terdefinisi), "ind" (tak tentu namun terbatas), "infinity" (kompleks tak hingga). Begitupun dengan batas c, dapat berupa sebarang nilai atau pada tak hingga (-inf, minf, dan inf).
Sebuah fungsi dapat dikatakan memiliki limit apabila limit kanan dan limit kiri nya memiliki nilai yang sama. Dimana, limit dari fungsi tersebut adalah nilai dari limit kanan dan limit kiri fungsi yang bernilai sama tadi.

```
>
```

```
>
```

Limit Kiri dan Kanan

Pengertian limit kiri dan limit kanan berkaitan dengan pendekatan nilai sebuah fungsi saat variabel inputnya mendekati suatu nilai tertentu dari sisi kiri atau kanan titik tersebut.
Limit kiri didefinisikan sebagai nilai yang didekati oleh fungsi saat variabel inputnya mendekati suatu nilai tertentu dari nilai yang lebih kecil atau dari sisi kiri.
Limit kiri ditulis sebagai:

$$\lim_{x \to c^-} f(x) = L$$

Ini berarti bahwa saat x mendekati c dari sisi kiri, nilai dari fungsi f(x) mendekati nilai L.
Sebaliknya, limit kanan didefinisikan sebagai nilai yang didekati oleh fungsi saat variabel inputnya mendekati suatu nilai tertentu dari nilai yang lebih besar atau dari sisi kanan.
Limit kanan ditulis sebagai:

$$\lim_{x \to c^+} f(x) = L$$

Ini berarti bahwa saat x mendekati c dari sisi kanan, nilai dari fungsi f(x) mendekati nilai L.
Dalam banyak kasus, untuk limit fungsi yang ada, nilai limit kiri dan limit kanan mungkin berbeda.

```
>
```

```
>
```

```
>
```

Limit pada EMT

Pada EMT cara mendefinisikan limit yaitu dengan format :
$showev('limit(f(x),x,c))
Format tersebut akan menampilkan limit yang dimaksud dan hasilnya. Jika kita ingin menampilkan hasilnya saja dari sebuah limit tanpa menampilkan limitnya, kita bisa menggunakan format :
'limit(f(x),x,c)
Sedangkan, untuk limit kanan dan limit kiri seperti pada definisi dapat ditampilkan di EMT dengan cara menambah opsi "plus" atau "minus" :
$showev('limit(f(x),x,c, plus)) atau 'limit(f(x),x,c, minus)
Limit dapat divisualisasikan menggunakan plot 2 dimensi. Pada EMT sendiri, format yang bisa digunakan untuk memvisualisasikan limit adalah :
plot2d("f(x)",-c,c):
aspect(1.5); plot2d("f(x)",c); plot2d(x,c>points,style="ow",>add):
Dengan f(x) adalah fungsi pada limit yang dicari, dan c berupa bilangan real menyesuaikan batas dari limit itu sendiri.

```
>
```

Limit Fungsi Aljabar

Limit fungsi aljabar adalah nilai yang didekati oleh sebuah fungsi saat variabel inputnya mendekati suatu nilai tertentu.
Secara matematis, kita dapat menyatakan limit fungsi aljabar sebagai berikut:
Diberikan fungsi f(x), dengan x mendekati suatu nilai c, maka limit fungsi f(x) saat x mendekati c dapat ditulis sebagai:

$$\lim_{x\to c} f(x) = L$$

Di mana L adalah nilai yang didekati oleh fungsi f(x) saat x mendekati c. Limit fungsi ini menggambarkan perilaku fungsi pada titik c dan dapat membantu kita mengidentifikasi apakah suatu fungsi memiliki nilai tertentu pada suatu titik atau apakah ada asimtot vertikal atau horizontal pada grafik fungsi tersebut.

```
>$showev('limit((x^3-13*x^2+51*x-63)/(x^3-4*x^2-3*x+18),x,3))
```

$$\lim_{x\to 3} \frac{x^3 - 13\,x^2 + 51\,x - 63}{x^3 - 4\,x^2 - 3\,x + 18} = -\frac{4}{5}$$

penyelesaian :
dari persamaan polinomial diatas dapat kita masukkan limitnya yakni x=3 kedalam persamaan tersebut, sehingga didapatkan :

$$\lim_{x\to 3} \frac{(x^3 - 13x^2 + 51x - 63)}{(x^3 - 4x^2 - 3x + 18)}$$

$$\lim_{x\to 3} \frac{(3^3 - 13(3)^2 + 51(3) - 63)}{(3^3 - 4(3)^2 - 3(3) + 18)}$$

$$\lim_{x\to 3} \frac{(27 - 117 + 153 - 63)}{(27 - 36 - 9 + 18)}$$

$$\lim_{x\to 3} \frac{(0)}{0} = \infty$$

Maka, untuk mencari limitnya dapat dicari faktor dari persamaan polinomial tersebut terlebih dahul, sehingga:

```
>$& factor((x^3-13*x^2+51*x-63)/(x^3-4*x^2-3*x+18))
```

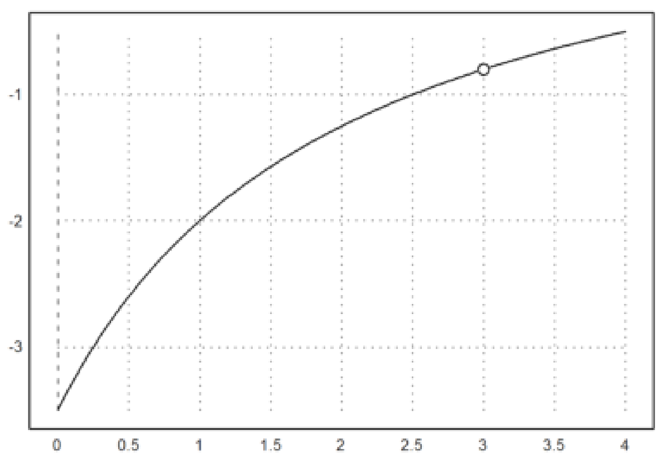
$$\frac{x-7}{x+2}$$

sehingga dari faktor diatas, dapat dicari limitnya yakni:

$$\lim_{x\to 3} \frac{x-7}{x+2} = \lim_{x\to 3} \frac{3-7}{3+2} = \frac{-4}{5}$$

MAKA DAPAT DIBUKTIKAN BAHWA NILAI LIMIT TERSEBUT BERNILAI BENAR

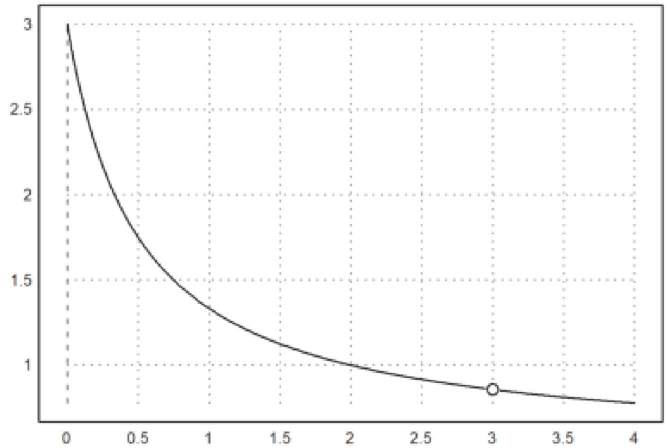
```
>aspect(1.5); plot2d("(x^3-13*x^2+51*x-63)/(x^3-4*x^2-3*x+18)",0,4); plot2d(3,-4/5,>points
```




```
>$showev('limit((x^2-9)/(2*x^2-5*x-3),x,3))
```

$$\lim_{x\to 3} \frac{x^2-9}{2\,x^2-5\,x-3} = \frac{6}{7}$$

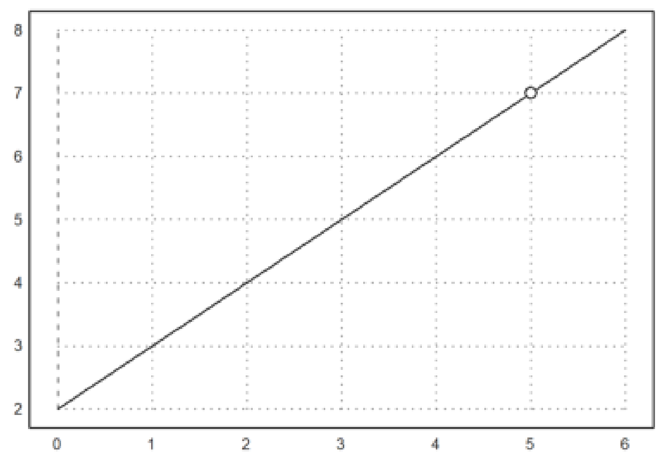
```
>aspect(1.5); plot2d("(x^2-9)/(2*x^2-5*x-3)",0,4); plot2d(3,6/7,>points,style="ow",>add):
```




```
>$showev('limit((x^2-3*x-10)/(x-5),x,5))
```

$$\lim_{x\to 5}{\frac{x^2-3\,x-10}{x-5}}=7$$

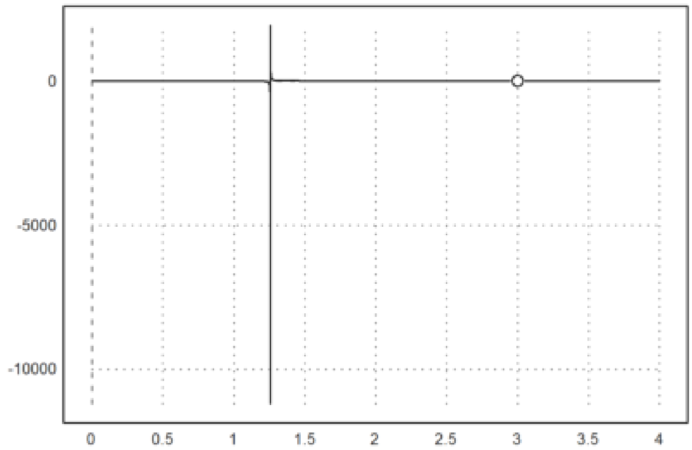
```
>aspect(1.5); plot2d("(x^2-3*x-10)/(x-5)",0,6); plot2d(5,7,>points,style="ow",>add):
```




```
>$showev('limit(((2*x^2-2*x+5)/(3*x^2+x-6)),x,3))
```

$$\lim_{x\to 3}{\frac{2\,x^2-2\,x+5}{3\,x^2+x-6}}=\frac{17}{24}$$

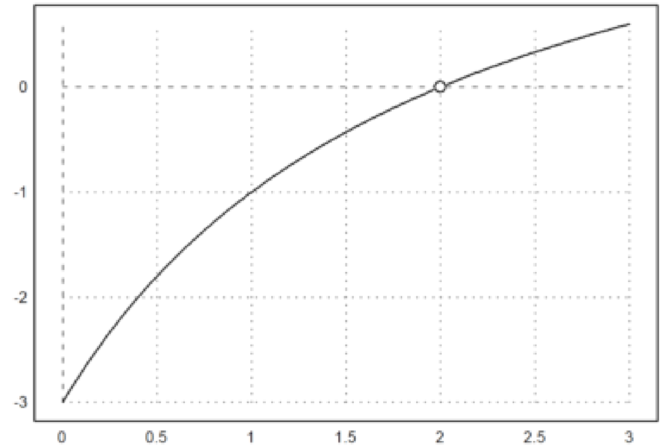
```
>aspect(1.5); plot2d("(2*x^2-2*x+5)/(3*x^2+x-6)",0,4); plot2d(3,17/24,>points,style="ow",>
```




```
>$showev('limit((3*x-6)/(x+2),x,2))
```

$$\lim_{x\to 2}\frac{3\,x-6}{x+2}=0$$

```
>aspect(1.5); plot2d("(3*x-6)/(x+2)",0,3); plot2d(2,0,>points,style="ow",>add):
```




```
>
```

Limit Fungsi Non Aljabar

1. Limit Fungsi Trigonometri

Limit fungsi trigonometri adalah nilai yang didekati oleh sebuah fungsi trigonometri saat variabel inputnya mendekati suatu nilai tertentu. Fungsi trigonometri melibatkan fungsi sinus, kosinus, tangen, kotangen, dan sebagainya. Pada umumnya, limit fungsi trigonometri dihitung dengan menggunakan pendekatan geometri yang melibatkan lingkaran unit.
Misalnya, untuk fungsi sinus, kita dapat menyatakan limit fungsi sinus saat x mendekati suatu nilai tertentu c sebagai:

$$\lim_{x \to c} sin(x) = sin(c)$$

Ini berarti bahwa saat x mendekati c, nilai sinus dari x akan mendekati sinus dari c.

```
>$showev('limit(2*x*sin(x)/(1-cos(x)),x,0))
```

$$2 \left(\lim_{x\to 0}{\frac{x\,\sin x}{1-\cos x}}\right) = 4$$

PENYELESAIAN :

$$2(\lim_{x \to 0} \frac{xsinx}{1-cosx})$$

$$2(\lim_{x \to 0} \frac{0sin(0)}{1-cos(0)})$$

$$2(\lim_{x \to 0} \frac{0}{1-1}) = \infty$$

MAKA kita perlu mengubah persamaan trigonometri diatas dengan menggunakan aturan L HOSTIPAL menjadi :

$$2(\lim_{x \to 0} \frac{xsin(x)}{1-cosx})$$

dideferensialkan (diturunkan)

$$2(\lim_{x \to 0} \frac{sinx + xcosx}{sinx})$$

berdasarkan aturan L HOSPITAL persamaan trigonometri tersebut masih menghasilkan latex: \infty
sehingga dapat kita turunkan lagi (dideferensialkan) menjadi:

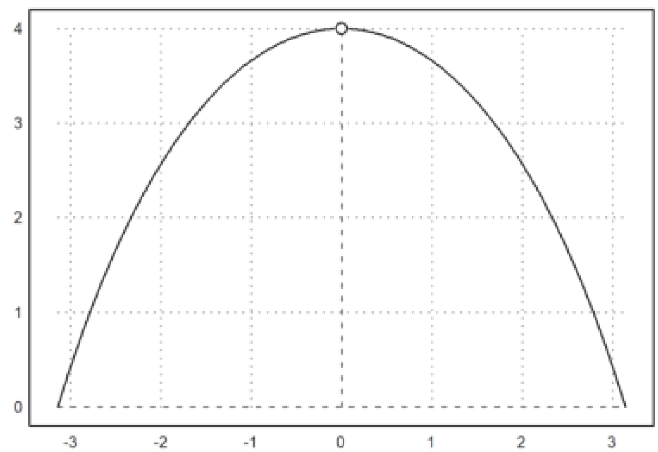
$$2(\lim_{x \to 0} \frac{2cosx - xsinx}{cosx})$$

$$2(\lim_{x \to 0} \frac{2cos(0) - (0)sin(0)}{cos(0)})$$

$$2(\lim_{x \to 0} \frac{2-0}{1} = 2 \times 2 = 4$$

JADI TERBUKTI BAHWA PERSAMAAN TRIGONOMETRI dari EMT TERSEBUT TERBUKTI TERBUKTI BENAR

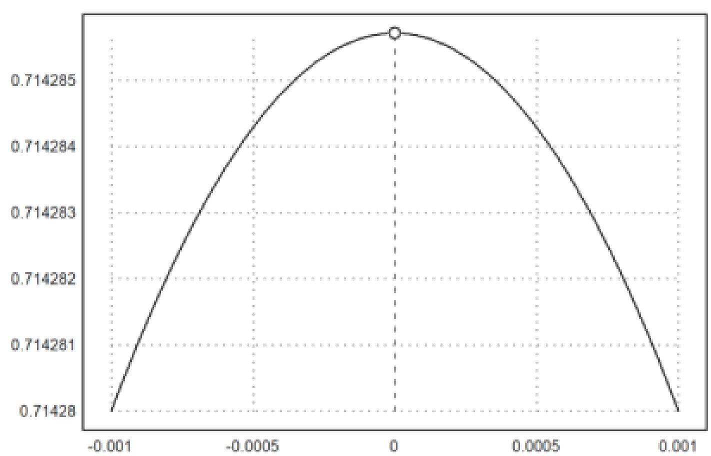
```
>plot2d("2*x*sin(x)/(1-cos(x))",-pi,pi); plot2d(0,4,>points,style="ow",>add):
```




```
>$showev('limit(cot(7*h)/cot(5*h),h,0))
```

$$\lim_{h\to 0}\frac{\cot(7\,h)}{\cot(5\,h)}=\frac{5}{7}$$

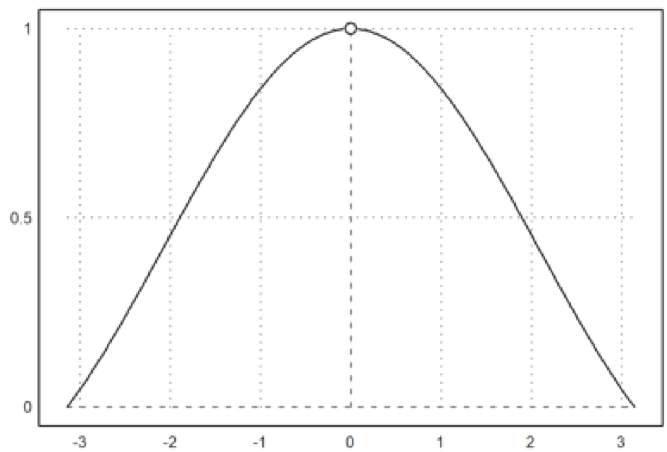
```
>plot2d("cot(7*x)/cot(5*x)",-0.001,0.001); plot2d(0,5/7,>points,style="ow",>add):
```




```
> $showev('limit(sin(x)/x,x,0))
```

$$\lim_{x\to 0}\frac{\sin x}{x}=1$$

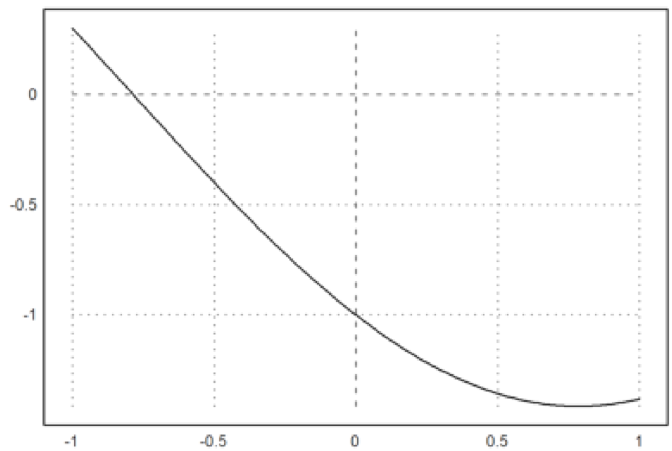
```
>plot2d("sin(x)/x",-pi,pi); plot2d(0,1,>points,style="ow",>add):
```




```
>$showev('limit(cos(2*x)/(sin(x) - cos (x)),x,0))
```

$$\lim_{x\to 0}\frac{\cos\left(2\,x\right)}{\sin x-\cos x}=-1$$

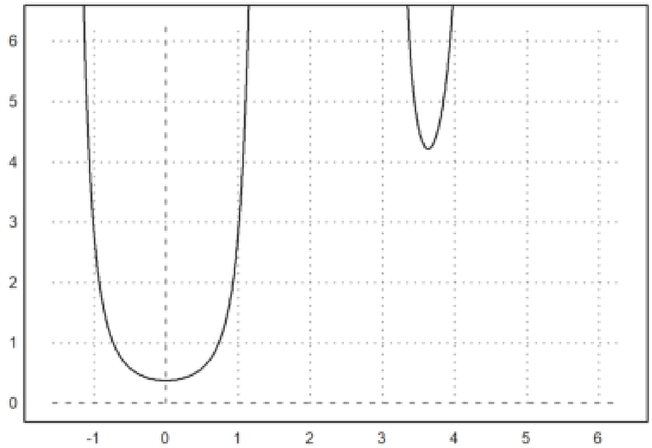
```
>plot2d("cos(2*x)/(sin(x) - cos (x))",-1,1):
```




```
>$showev('limit((3*x*tan(x))/(1-cos(4*x)),x,0))
```

$$3\,\left(\lim_{x\to 0}\frac{x\,\tan x}{1-\cos\left(4\,x\right)}\right)=\frac{3}{8}$$

```
>plot2d("(3*x*tan(x))/(1-cos(4*x))",-pi/2,2pi,0,2pi):
```



2. Limit Fungsi Eksponensial

limit fungsi eksponensial adalah nilai yang didekati oleh sebuah fungsi eksponensial saat variabel inputnya mendekati suatu nilai tertentu. Fungsi eksponensial melibatkan bentuk fungsional seperti a^x, dengan a sebagai basis dan x sebagai eksponen.
Misalnya, limit fungsi eksponensial saat x mendekati suatu nilai tertentu c dapat dinyatakan sebagai:

$$\lim_{x\to c} a^x = a^c$$

Ini berarti bahwa saat x mendekati c, nilai dari fungsi eksponensial a^x akan mendekati nilai a^c.
Limit fungsi trigonometri dan limit fungsi eksponensial memiliki beragam sifat dan properti yang dapat digunakan dalam analisis matematika. Mereka juga sering digunakan dalam pemodelan dan aplikasi ilmu pengetahuan yang melibatkan perubahan atau pertumbuhan yang berkaitan dengan sudut atau eksponensial.

```
>$showev('limit((1+2/(3*x))^(5*x),x,inf))
```

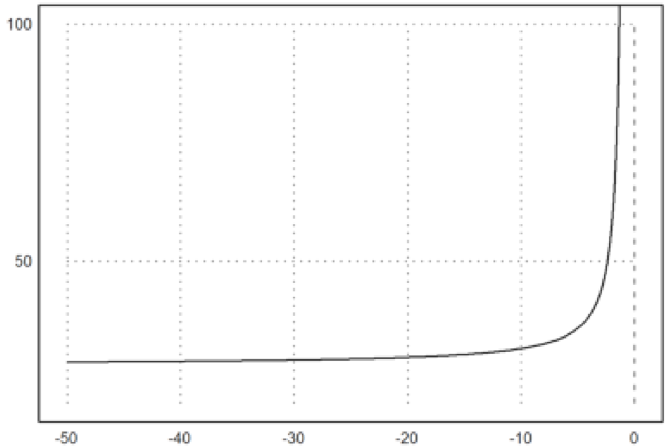
$$\lim_{x\to\infty} \left(\frac{2}{3\,x}+1\right)^{5\,x} = e^{\frac{10}{3}}$$

Penyelesaian limit fungsi eksponensial tersebut:

$$\lim_{x\to\infty}\left(1+\frac{2}{3x}\right)^{5x} = \lim_{x\to\infty}\left(1+\frac{2}{3x}\right)^{5x\cdot\frac{3}{3}}$$
$$= \lim_{x\to\infty}\left(1+\frac{2}{3x}\right)^{3x\cdot\frac{5}{3}} = \left[\lim_{x\to\infty}\left(1+\frac{2}{3x}\right)^{3x}\right]^{\frac{5}{3}}$$
$$= \left[\lim_{3x\to\infty}\left(1+\frac{2}{3x}\right)^{3x}\right]^{\frac{5}{3}} = \left[\lim_{y\to\infty}\left(1+\frac{2}{y}\right)^{y}\right]^{\frac{5}{3}}$$
$$(e^2)^{\frac{5}{3}} = e^{\frac{10}{3}}$$

JADI TERBUKTI PENYELESAIAN LIMIT FUNGSI EKSPONENSIAL TERSEBUT

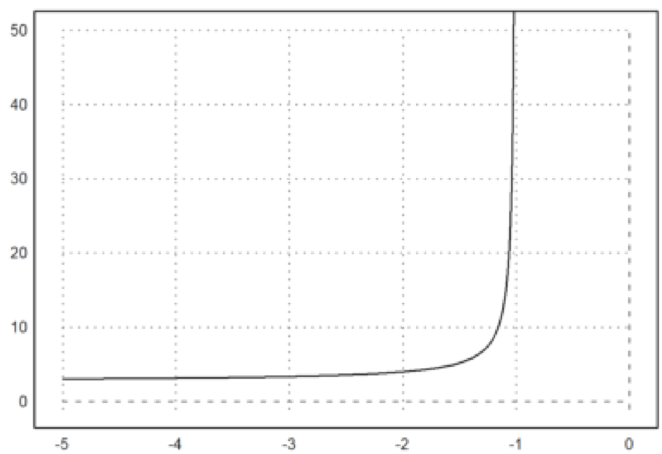
```
>plot2d("(1+2/(3*x))^(5*x)",-50,0,20,100):
```




```
>$showev('limit((1+1/x)^x,x,inf))
```

$$\lim_{x \to \infty} \left(\frac{1}{x} + 1\right)^x = e$$

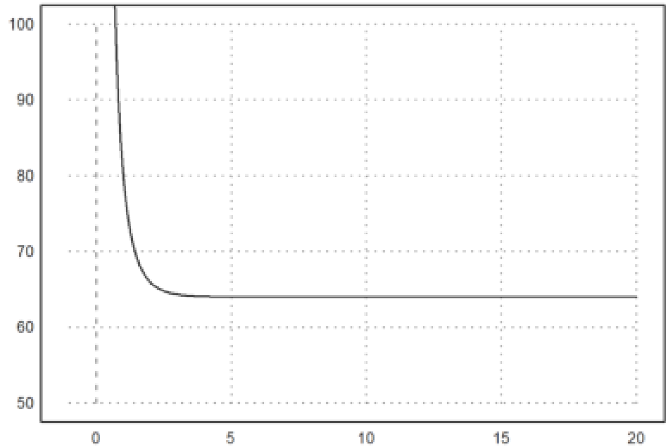
```
>plot2d("(1+1/x)^x",-5,0,-1,50):
```




```
>$showev('limit((2^(4*x)+2^(6*x))^(1/x),x,inf))
```

$$\lim_{x \to \infty} \left(2^{6\,x} + 2^{4\,x}\right)^{\frac{1}{x}} = 64$$

```
>plot2d("(2^(4*x)+2^(6*x))^(1/x)",-1,20,50,100):
```



```
>
```

3. Limit Fungsi Logaritma

limit fungsi logaritma adalah nilai yang didekati oleh sebuah fungsi logaritma saat variabel inputnya mendekati suatu nilai tertentu. Fungsi logaritma melibatkan logaritma basis a dari x, yang ditulis sebagai log_a(x). Misalnya, untuk fungsi logaritma alami (basis e), kita dapat menyatakan limit fungsi logaritma saat x mendekati suatu nilai tertentu c sebagai:

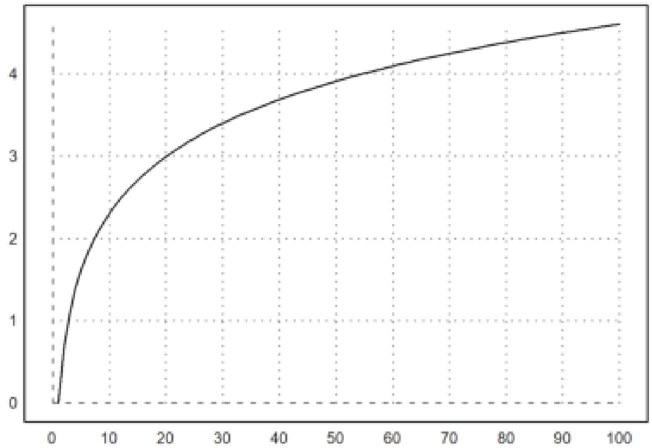
$$\lim_{x \to c} ln(x) = ln(c)$$

Ini berarti bahwa saat x mendekati c, nilai logaritma natural dari x akan mendekati logaritma natural dari c. i dan limit fungsi eksponensial memiliki beragam sifat dan properti yang dapat digunakan dalam analisis matematika. Mereka juga sering digunakan dalam pemodelan dan aplikasi ilmu pengetahuan yang melibatkan perubahan atau pertumbuhan yang berkaitan dengan sudut atau eksponensial.

```
>$showev('limit(log(x), x, minf))
```

$$\lim_{x \to -\infty} \log x = \mathit{infinity}$$

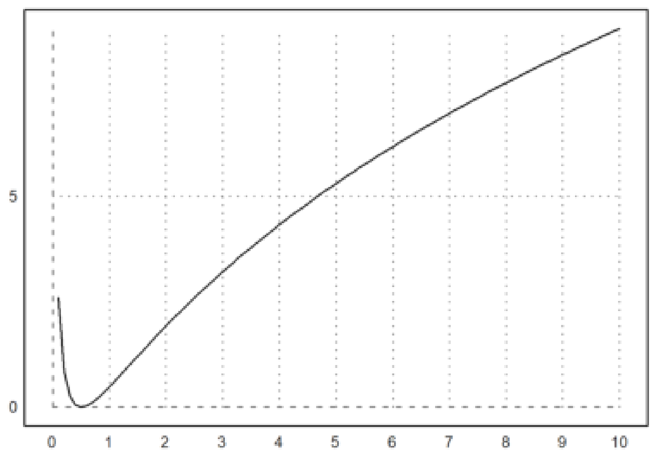
```
>plot2d("log(x)",0,100):
```




```
>$showev('limit((log(2*x))^2,x,2))
```

$$\lim_{x\to 2}\log^2{(2\,x)} = \log^2 4$$

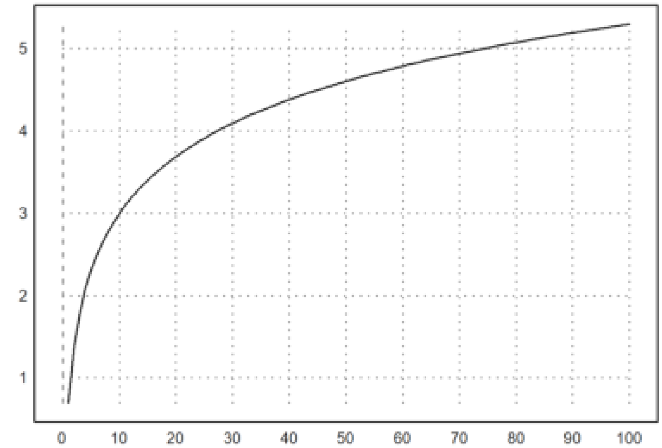
```
>plot2d("log(2*x)^2",0,10):
```




```
>$showev('limit(log(2*x), x, 0))
```

$$\lim_{x\to 0}\log{(2\,x)} = \mathit{infinity}$$

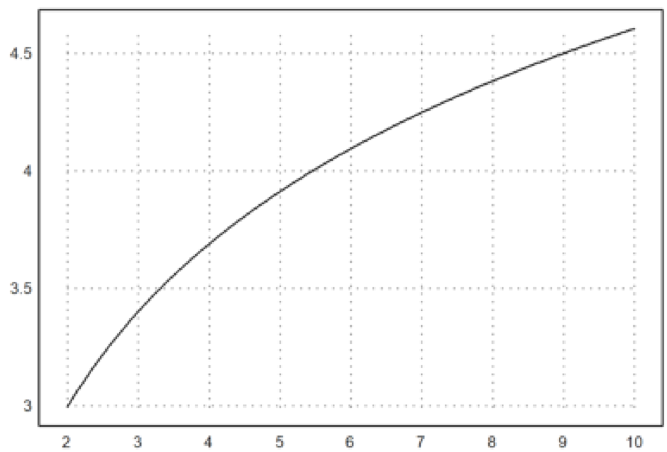
```
>plot2d("log(2*x)",0,100):
```




```
>$showev('limit(log(10*x), x, 10))
```

$$\lim_{x\to 10} \log (10\,x) = \log 100$$

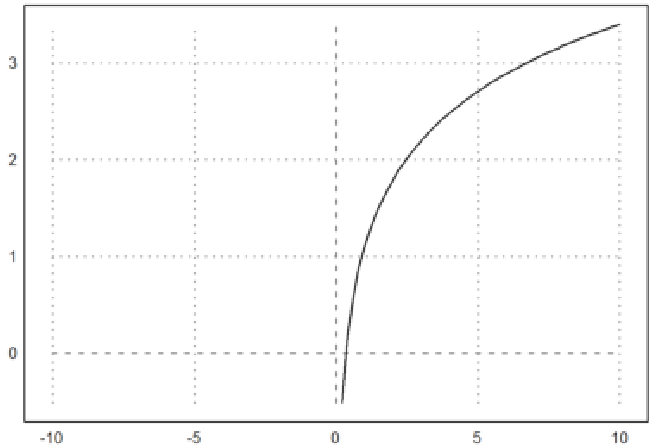
```
>plot2d("log(10*x)",2,10):
```




```
>$showev('limit(log(3*x),x,0))
```

$$\lim_{x\to 0} \log (3\,x) = \mathit{infinity}$$

```
>plot2d("log(3*x)",-10,10):
```



BAB 3. TURUNAN FUNGSI
Cakupan materi :
-Definisi turunan
-Sifat-sifat turunan
-Turunan fungsi aljabar, trigonometri, eksponensial, logaritma,dan komposisi fungsi.
-Visualisasi dan kurva fungsi
-Aplikasi turunan

1. Definisi Turunan

```
Turunan fungsi atau bisa disebut juga differensial merupakan
```

konsep yang mengukur perubahan instan dari suatu fungsi terhadap perubahan variabel independen. Turunan atau differensial suatu fungsi menggambarkan seberapa cepat nilai fungsi tersebut berubah pada titik tertentu dalam domain fungsi.
Secara umum, jika "f(x)" adalah suatu fungsi yang tergantung pada variabel "x," maka turunan atau differensialnya, disimbolkan sebagai "f'(x)" atau "df/dx," didefinisikan sebagai berikut

$$f'(x) = \lim_{h\to 0} \frac{f(x+h) - f(x)}{h}$$

Di sini, "h" adalah perubahan kecil dalam variabel "x." Ketika "h" mendekati nol, kita mendapatkan perubahan instan dari fungsi "f(x)" pada titik "x".

2. Sifat-sifat Turunan

1. Sifat Linear
Turunan dari jumlah atau selisih dua fungsi adalah jumlah atau selisih dari turunan-turunan fungsi-fungsi tersebut.

$$(f(x) + g(x))' = f'(x) + g'(x)$$

$$(f(x) - g(x))' = f'(x) - g'(x)$$

2. Aturan Perkalian
Turunan dari perkalian dua fungsi adalah hasil dari turunan pertama dikalikan dengan fungsi kedua ditambah fungsi pertama dikalikan dengan turunan kedua.

$$(f(x).g(x))' = f'(x).g(x) + f(x).g'(x)$$

$$(uv)' = u'v + uv'$$

3. Aturan Rantai(Chain Rule)
Aturan rantai digunakan ketika kita memiliki komposisi fungsi, yaitu suatu fungsi yang terdiri dari fungsi-fungsi lain. Aturan rantai mengatakan bahwa turunan dari fungsi komposisi adalah produk dari turunan-turunan fungsi-fungsi tersebut.

$$(f(g(x)))' = f'(g(x)).g'(x)$$

4. Turunan Konstanta
Turunan dari suatu konstanta adalah nol.

$$f(x) = k$$

$$f'(x) = 0$$

5. Turunan Identitas
Turunan dari x terhadap x adalah 1.

$$f(x) = x$$

$$f'(x) = 1$$

6. Turunan dari x^n
Jika

$$f(x) = x^n,$$

dimana n adalah bilangan bulat positif, maka

$$f'(x) = n.x^n - 1$$

7. Turunan Eksponensial
Turunan dari fungsi eksponensial, seperti

$$f(x) = e^x$$

adalah dirinya sendiri, yaitu

$$f'(x) = e^x$$

$$(e^x)' = e^x$$

8. Turunan Logaritma
Turunan dari logaritma alami, seperti

$$f(x) = ln(x)$$

adalah 1/x, yaitu

$$f'(x) = 1/x$$

$$(ln(x))' = 1/x$$

3. TURUNAN FUNGSI ALJABAR

```
3.1 Definisi
   Turunan fungsi aljabar adalah proses untuk menemukan turunan
```

(differensial) dari fungsi matematika yang termasuk dalam kategori aljabar. Fungsi-fungsi aljabar adalah fungsi yang terdiri dari operasi-operasi aljabar dasar, seperti penambahan, pengurangan, perkalian, dan pembagian. Contoh umum dari fungsi aljabar yaitu fungsi linier, fungsi pangkat, fungsi akar, fungsi irasional, dan masih banyak lainnya.

```
3.2 Contoh Soal & Visualisasi Kurvanya
```

$$f(x) = x^n$$

Menggunakan definisi limit

```
>$showev('limit(((x+h)^n-x^n)/h,h,0)) // turunan x^n
```

$$\lim_{h\to 0}\frac{(x+h)^n - x^n}{h} = n\,x^{n-1}$$

Pembuktian

$$f'(x) = \lim_{h\to 0}\frac{f(x+h) - f(x)}{h}$$

Untuk

$$f(x) = x^n, f(x+h) = (x+h)^n$$

$$\frac{d}{dx}x^n = \lim_{h\to 0}\frac{(x+h)^n - x^n}{h}$$

Dengan

$$(a+b)^n = \sum_{k=0}^{n} a^k b^{n-k}$$

maka

$$= \lim_{h\to 0} \frac{(x^n + \frac{n}{1!}x^{n-1}h + \frac{n(n-1)}{2!}x^{n-2}h^2 + \frac{n(n-1)(n-2)}{3!}x^{n-3}h^3 + ...) - x^n}{h}$$

$$= \lim_{h\to 0} \frac{n.x^{n-1}h + \frac{n(n-1)}{2!}x^{n-2}h^2 + \frac{n(n-1)(n-2)}{3!}x^{n-3}h^3 + ...}{h}$$

$$= \lim_{h\to 0} n.x^{n-1} + \frac{n(n-1)}{2!}.x^{n-2}h + \frac{n(n-1)(n-2)}{3!}.x^{n-3}h^2 + ...$$

$$= n.x^{n-1} + 0 + 0 + ... + 0$$

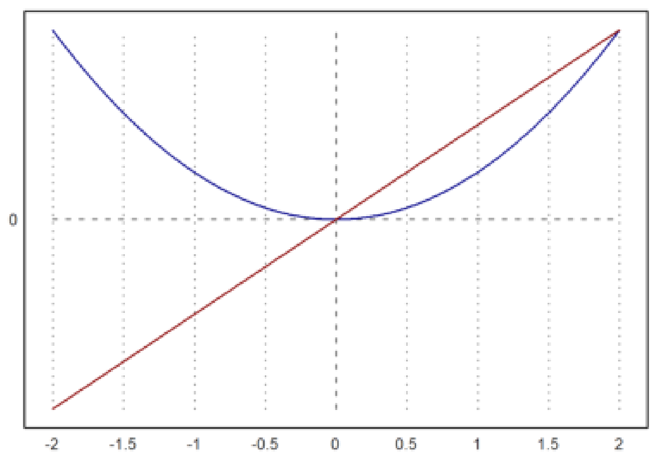
$$= n.x^{n-1}$$

Jadi, terbukti benar bahwa

$$f'(x^n) = n.x^{n-1}$$

Visualisasi grafiknya

```
>plot2d(["x^2","2*x^(2-1)"],color=[blue,red]): //grafik fungsi dan turunannya
```



$$f(x) = 2x^2 + 5x + 9$$

Cara pertama (definisi limit)

```
>$showev('limit(((2*(x+h)^2 + 5*(x+h) + 9) - (2*x^2 + 5*x + 9))/h,h,0))// turunan
```

$$\lim_{h\to 0} \frac{2\,(x+h)^2 - 2\,x^2 + 5\,(x+h) - 5\,x}{h} = 4\,x + 5$$

Pembuktian

```
>p &= expand((2*(x+h)^2 + 5*(x+h) + 9) - (2*x^2 + 5*x + 9))|simplify; $p //pembilang dijab
```

$$4\,h\,x+2\,h^2+5\,h$$

```
>q &=ratsimp(p/h); $q // ekspresi yang akan dihitung limitnya disederhanakan (dibagi h)
```

$$4\,x+2\,h+5$$

```
>$limit(q,h,0) // nilai limit sebagai turunan
```

$$4\,x+5$$

Pada cara pertama, menggunakan definisi limit yang sudah dituliskan di atas, kemudian pembilangnya dijabarkan dan disederhanakan lalu dibagi dengan h. Setelah itu, sesuai dengan definisi limit, ambil batas (limit) saat "h" mendekati nol sehingga didapatkan turunan atau derivatif dari fungsi "f(x)" pada titik "x" yaitu 4x+5. Cara kedua(menggunakan formula diff)

```
>function f(x) &= 2*x^2+5*x+9 //Mendefinisikan fungsi f(x)
```

```
                                 2
                              2 x  + 5 x + 9
```

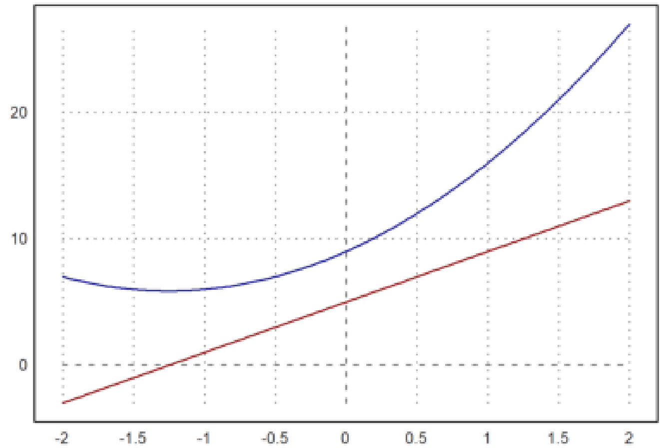
```
>$showev('diff(f(x),x))
```

$$\frac{d}{d\,x}\,\left(2\,x^2+5\,x+9\right)=4\,x+5$$

```
>function df(x) &=diff(f(x),x)// df(x)=f'(x)
```

```
                                  4 x + 5
```

```
>plot2d(["f(x)","df(x)"],color=[blue,red]): //grafik fungsi dan turunannya
```



$$f(x) = \frac{2x-5}{x+2}$$

Cara pertama (definisi limit)

```
>$showev('limit(((2*(x+h)-5)/((x+h) + 2) - (2*x-5)/(x+2))/h,h,0))// turunan 2x^2+5
```

$$\lim_{h \to 0} \frac{\frac{2\,(x+h)-5}{x+h+2} - \frac{2\,x-5}{x+2}}{h} = \frac{9}{x^2+4\,x+4}$$

```
>p &= expand((2*(x+h)-5)/((x+h) + 2) - (2*x-5)/(x+2))|simplify; $p //pembilang dij
```

$$\frac{2\,x}{x+h+2} + \frac{2\,h}{x+h+2} - \frac{5}{x+h+2} - \frac{2\,x}{x+2} + \frac{5}{x+2}$$

```
>q &=ratsimp(p/h); $q // ekspresi yang akan dihitung limitnya disederhanakan
```

$$\frac{9}{x^2+(h+4)\,x+2\,h+4}$$

```
>$limit(q,h,0) // nilai limit sebagai turunan
```

$$\frac{9}{x^2+4\,x+4}$$

Cara kedua (menggunakan formula diff)

```
>function f(x) &= (2*x-5)/(x+2)
```

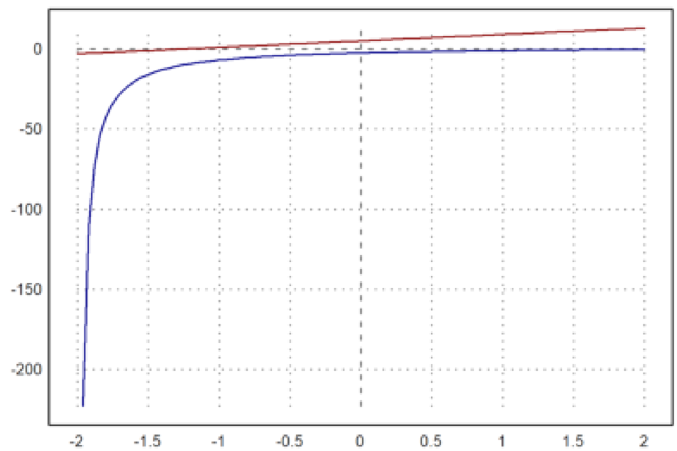
```
2 x - 5
-------
 x + 2
```

```
>$showev('diff(f(x),x))
```

$$\frac{d}{d\,x}\,\left(\frac{2\,x-5}{x+2}\right)=\frac{2}{x+2}-\frac{2\,x-5}{\left(x+2\right)^2}$$

Visualisasi grafiknya

```
>plot2d(["f(x)","df(x)"],color=[blue,red]): //grafik fungsi dan turunannya
```



4. Turunan Fungsi Trigonometri

```
    4.1 Definisi
        Turunan fungsi trigonometri adalah proses untuk menghitung
```

turunan (differensial) dari fungsi trigonometri, seperti sin(x), cos(x), tan(x), dan fungsi trigonometri lainnya.

```
    4.2 Contoh Soal dan Visualisai Grafiknya
```

$$f(x) = sin x$$

```
>function f(x) &= sin(x) // mendifinisikan fungsi f
```

```
                    sin(x)
```

```
>function df(x) &=diff(f(x),x) // df(x) = f'(x)
```

```
                    cos(x)
```

Pembuktian

$$f'(x) = \lim_{h\to 0} \frac{sin(x+h) - sin(x)}{h}$$

$$sin(a+b) = sin(a)cos(a) + cos(a)sin(b)$$

$$= \lim_{h\to 0} \frac{sin(x)cos(h) + cos(x)sin(h) - sin(x)}{h}$$

$$= \lim_{h\to 0} sinx. \frac{cos(h) - 1}{h} + \lim_{h\to 0} cos(x). \frac{sin(h)}{h}$$

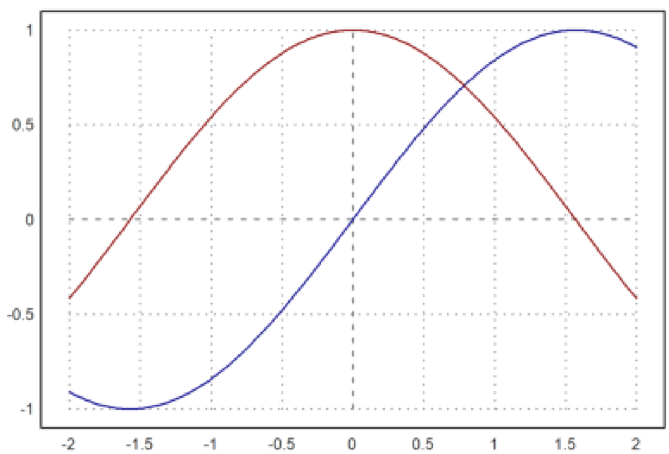
$$= sin(x).0 + cos(x).1$$

$$= cos(x)$$

Jadi, terbukti benar bahwa

$$f'(sin(x)) = cos(x)$$

```
>plot2d(["f(x)","df(x)"],color=[blue,red]): //grafik fungsi dan turunannya
```



$$f(x) = arcsin(x)$$

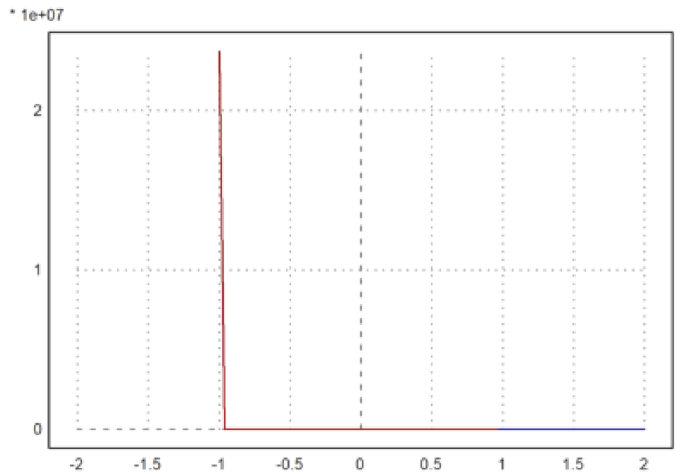
```
>function f(x) &= arcsin(x) // mendifinisikan fungsi f
```

```
arcsin(x)
```

```
>$showev('limit((asin(x+h)-asin(x))/h,h,0)) // turunan arcsin(x)
```

$$\lim_{h\to 0} \frac{\arcsin(x+h) - \arcsin x}{h} = \frac{1}{\sqrt{1-x^2}}$$

```
>plot2d(["log(x)","1/(sqrt(1-x^2))"],color=[blue,red]): //grafik fungsi dan turunannya
```



$$f(x) = sin(3x^5 + 7)^2$$

```
>function f(x) &= sin(3*x^5+7)^2
```

```
                        2     5
                     sin  (3 x  + 7)
```

```
>$showev('diff(f(x),x))
```

$$\frac{d}{d\,x}\,\sin^2\left(3\,x^5+7\right)=30\,x^4\,\cos\left(3\,x^5+7\right)\,\sin\left(3\,x^5+7\right)$$

Mencari turunan menggunakan formula diff. Diff sendiri merupakan formula pada Euler Math Toolbox yang berfungsi untuk mencari turunan.

```
>$% with x=3
```

$$\%at\left(\frac{d}{d\,x}\,\sin^2\left(3\,x^5+7\right),x=3\right)=2430\,\cos 736\,\sin 736$$

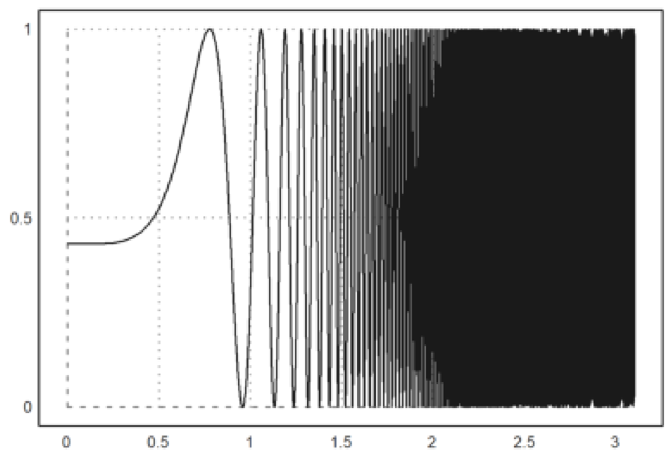
Saat x=3 diperoleh turunan pertama seperti dituliskan di atas

```
>$float(%)
```

$$\%at\left(\frac{d^{1.0}}{d\,x^{1.0}}\,\sin^2\left(3.0\,x^5+7.0\right),x=3.0\right)=1198.728637211748$$

Jika turunan pertama saat x=3 dioperasikan menghasilkan seperti yang dituliskan di atas.

```
>plot2d(f,0,3.1):
```



'0': Ini adalah batas bawah dari rentang sumbu-x yang akan digambarkan dalam grafik. Dalam hal ini, garis awal grafik dimulai dari x=0.
'3.1': Ini adalah batas atas dari rentang sumbu-x yang akan digambarkan dalam grafik. Dalam hal ini, garis akhir grafik adalah x=3.1.
Hasil dari perintah plot2d(f,0,3.1) adalah grafik dari fungsi f(x) yang digambarkan dari rentang x dari 0 hingga 3.1. Jadi, grafik ini akan menunjukkan bagaimana fungsi f(x) berubah saat x bergerak dari 0 hingga 3.1.

5. Turunan Fungsi Eksponensial

5.1 Definisi

```
Turunan dari fungsi eksponensial ditemukan berdasarkan aturan
```

dasar turunan. Fungsi eksponensial dasar adalah fungsi dalam bentuk "a^x," di mana "a" adalah konstanta positif dan "x" adalah variabel independen.
5.2 Contoh Soal dan Visualisasi Grafik

$$f(x) = 3x^x$$

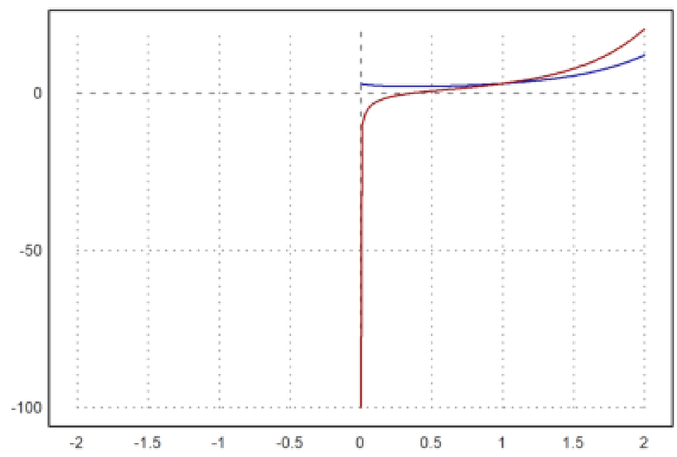
```
>function f(x) &= 3*x^x
```

```
   x
3 x
```

```
>&assume(x>0); $showev('limit((f(x+h)-f(x))/h,h,0)) // turunan f(x)=3x^x
```

$$\lim_{h\rightarrow 0}{\frac{3\,\left(x+h\right)^{x+h}-3\,x^{x}}{h}}=x^{x}\,\left(3\,\log x+3\right)$$

```
>plot2d(["f(x)","x^x*(3*log(x) +3)"],color=[blue,red]): //grafik fungsi dan turunannya
```



$$f(x) = e^x$$

```
>$factor(E^(x+h)-E^x)
```

$$\left(e^h - 1\right)\, e^x$$

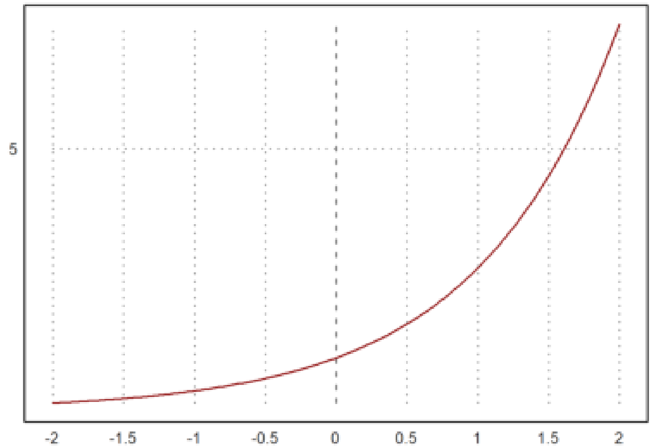
Jika langsung menggunakan definisi limit maka akan error sehingga e^(x+h) - e^x perlu difaktorkan dahulu.

```
>$showev('limit(factor((E^(x+h)-E^x)/h),h,0)) // turunan f(x)=e^x
```

$$\left(\lim_{h\to 0}\frac{e^h - 1}{h}\right) e^x = e^x$$

Setelah difaktorkan maka bisa dicari limitnya dan diperoleh turunan pertama dari e^x adalah fungsi itu sendiri (ingat kembali sifat turunan eksponensial).

```
>plot2d(["E^x","E^x"],color=[blue,red]): //grafik fungsi dan turunannya
```



6. Turunan Fungsi Logaritma

```
6.1 Definisi
    Turunan dari fungsi logaritma adalah aturan turunan yang
```

digunakan untuk menghitung turunan fungsi logaritma alami (logaritma berbasis e, biasanya disimbolkan sebagai "ln(x)") dan logaritma berbasis lain (seperti logaritma berbasis 10 atau berbasis lainnya).

```
6.2 Contoh Soal dan Visualisasi Grafiknya
```

$$f(x) = log(x)$$

```
>$showev('limit((log(x+h)-log(x))/h,h,0)) // turunan log(x)
```

$$\lim_{h \to 0} \frac{\log\left(x+h\right)-\log x}{h}=\frac{1}{x}$$

Pembuktian

$$f'(x) = \lim_{h \to 0} \frac{log(x+h) - logx}{h}$$

$$= \lim_{h \to 0} \frac{\frac{d}{dh}(log(x+h) - logx)}{\frac{d}{dh}(h)}$$

$$= \lim_{h \to 0} \frac{\frac{1}{x+h}}{1}$$

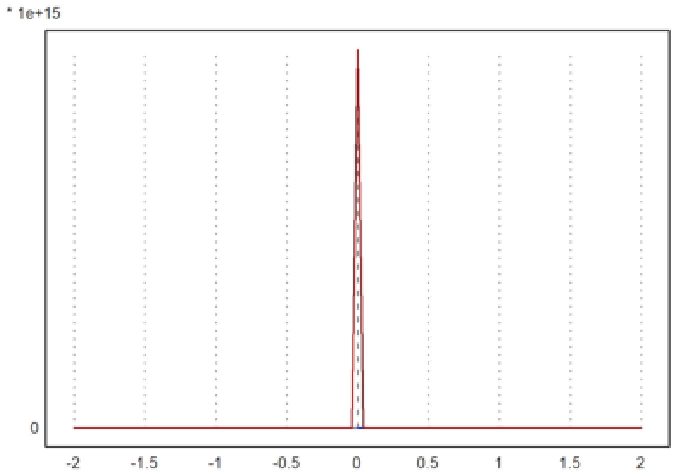
$$= \lim_{h \to 0} \frac{1}{x+h}$$

$$= \frac{1}{x}$$

Jadi, terbukti benar bahwa

$$f'(x) = \lim_{h \to 0} \frac{log(x+h) - logx}{h} = \frac{1}{x}$$

```
>plot2d(["log(x)","1/x"],color=[blue,red]): //grafik fungsi dan turunannya
```



7. Turunan Fungsi Komposisi

```
7.1 Definisi           Turunan fungsi komposisi adalah aturan
```

turunan yang digunakan untuk menghitung turunan dari fungsi yang merupakan hasil dari komposisi dua atau lebih fungsi.

```
7.2 Contoh Soal dan Visualisasi Grafiknya
```

$$f(x) = x^2 + 1$$

$$g(x) = x + 5$$

```
>function f(x) &= x^2+1
```

```
                        2
                       x  + 1
```

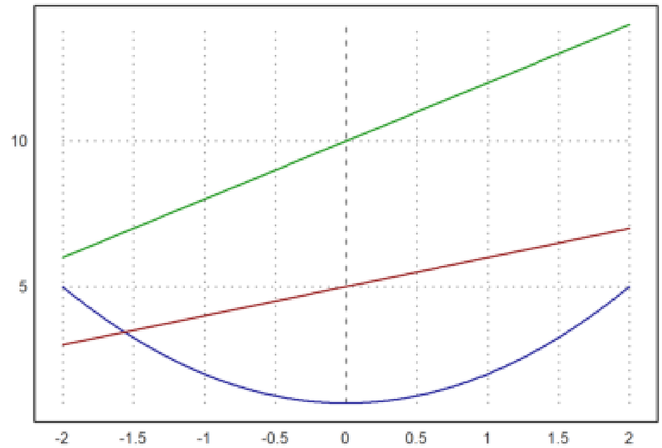
```
>function g(x) &= x+5
```

```
                       x + 5
```

```
>$showev('diff(f(g(x)),x))
```

$$\frac{d}{d\,x}\,\left(\left(x+5\right)^2+1\right)=2\,\left(x+5\right)$$

```
>plot2d(["f(x)","g(x)","2*(x+5)"],color=[blue,red,green]): //grafik fungsi dan turunannya
```



9. Aplikasi Turunan

```
Aplikasi dari turunan yaitu untuk mengoptimasi fungsi. Turunan
```

digunakan dalam optimisasi matematika untuk menemukan titik maksimum atau minimum dari suatu fungsi (nilai ekstrim).

$$f(x) = 5cos(2x) - 2xsin(2x)$$

```
>function f(x) &=5*cos(2*x)-2*x*sin(2*x) // mendifinisikan fungsi f
```

```
5 cos(2 x) - 2 x sin(2 x)
```

Langkah pertama yaitu mendefinisikan fungsinya dahulu supaya memudahkan dalam mencari turunan dan visualisasi grafiknya.

```
>function df(x) &=diff(f(x),x) // df(x) = f'(x)
```

```
- 12 sin(2 x) - 4 x cos(2 x)
```

Langkah kedua, mencari turunannya.

```
>xp=solve("df(x)",1,2,0) // solusi f'(x)=0 pada interval [1, 2]
```

```
1.35822987384
```

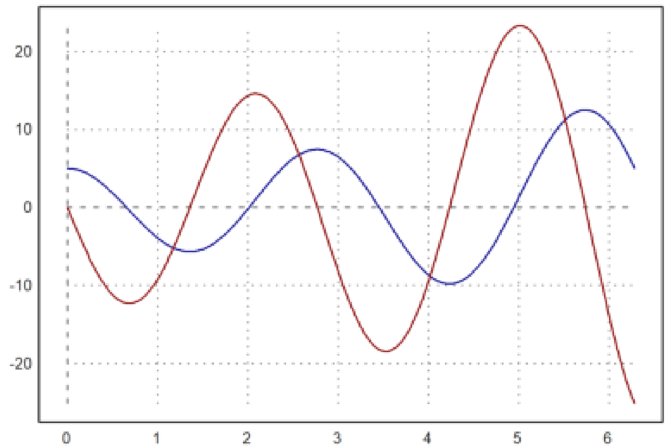
Mencari nilai ekstrimnya, nilai ekstrim diperoleh saat turunan pertamanya = 0.

```
>df(xp), f(xp) // cek bahwa f'(xp)=0 dan nilai ekstrim di titik tersebut
```

```
0
-5.67530133759
```

Diperoleh titik ekstrim

```
>plot2d(["f(x)","df(x)"],0,2*pi,color=[blue,red]): //grafik fungsi dan turunannya
```



Latihan Soal

1. Tentukan nilai turunan berikut dan sketsakan grafiknya.

$$f(x) = 5x^3 - 4$$

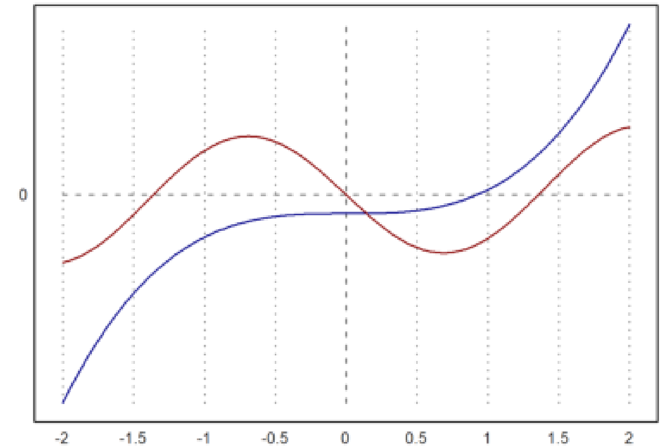
```
>function f(x) &= 5*x^3-4; $f(x)
```

$$5\,x^3 - 4$$

```
>function df(x) &= limit((f(x+h)-f(x))/h,h,0); &df(x)//df(x)=f'(x)
```

```
- 12 sin(2 x) - 4 x cos(2 x)
```

```
>plot2d(["f(x)","df(x)"],color=[blue,red]):
```



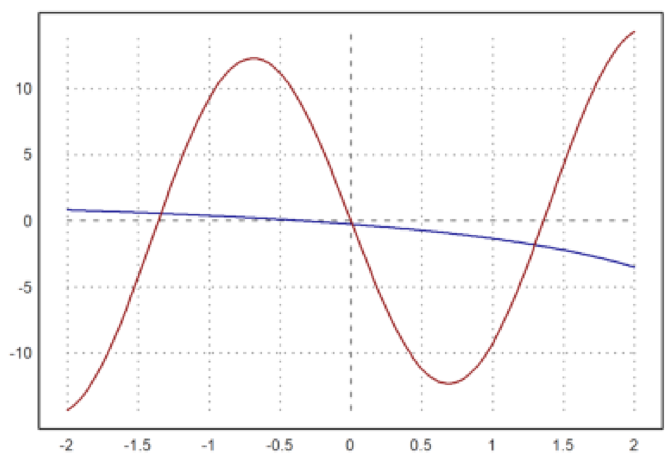
2. Carilah turunan dari fungsi berikut

$$f(x) = \frac{3x+1}{x-4}$$

```
>function f(x) &= (3*x+1)/(x-4); $f(x)
```

$$\frac{3\,x+1}{x-4}$$

```
>function df(x) &= limit((f(x+h)-f(x))/h,h,0); $df(x) // df(x) = f'(x)
>plot2d(["f(x)","df(x)"],color=[blue,red]):
```



3. Carilah turunan fungsi berikut.

$$f(x) = 4sin(x) - cos(x)$$

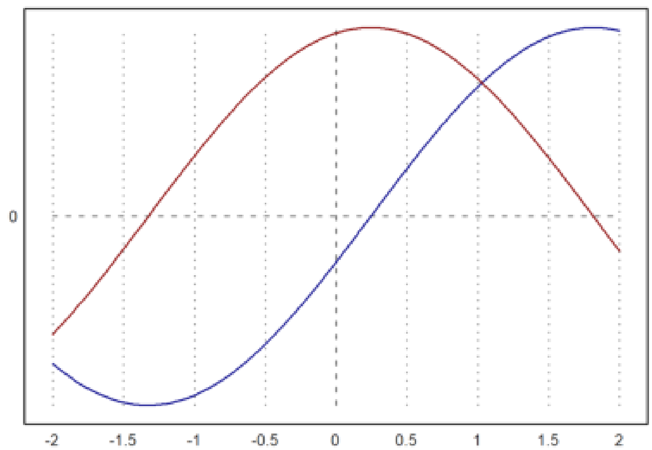
```
>function f(x) &= (4*sin(x)-cos(x)); $f(x)
```

$$4\sin x-\cos x$$

```
>function df(x) &= limit((f(x+h)-f(x))/h,h,0); &df(x)
```

```
sin(x) + 4 cos(x)
```

```
>plot2d(["f(x)","df(x)"],color=[blue,red]):
```



4. Tentukan turunan dan grafik fungsi berikut.

$$f(x)=\frac{sin(x)+cos(x)}{sin(x)}$$

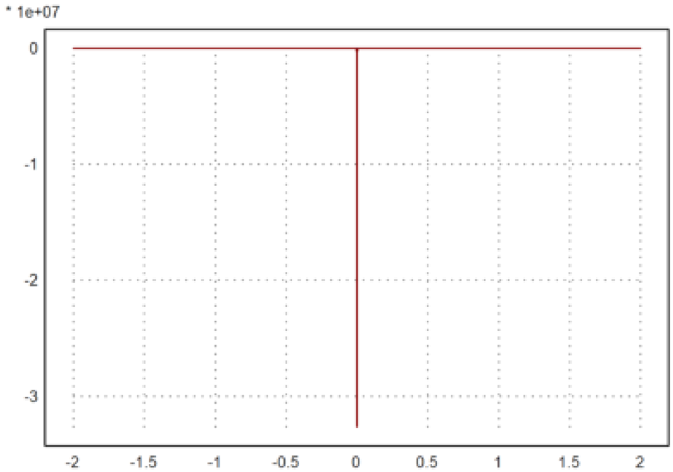
```
>function f(x) &= (sin(x)+cos(x))/(sin(x)); $f(x)
```

$$\frac{\sin x+\cos x}{\sin x}$$

```
>function df(x) &= limit((f(x+h)-f(x))/h,h,0); $df(x) // df(x) = f'(x)
```

$$\frac{-\sin^2 x-\cos^2 x}{\sin^2 x}$$

```
>plot2d(["f(x)","df(x)"],color=[blue,red]):
```



Sekian yang bisa disampaikan, mohon maaf bila terdapat kesalahan. Terima kasih.

BAB 5

KB PEKAN 8: MENGGUNAKAN EMT UNTUK GEOMETRI

[a4paper,10pt]article eumat

Fransisca Renita Pejoresa
22305144012
Matematika E

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 geometeri 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 atur tiga poin dan plot.

```
>A=[1,0]; plotPoint(A,"A"); // definisi dan gambar tiga titik
>B=[0,1]; plotPoint(B,"B");
>C=[2,2]; plotPoint(C,"C");
```

Lalu 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 untuk garis adalah [a, b, c], yang merepresentasikan garis
dengan persamaan ax + by = c

```
>lineThrough(B,C) // garis yang melalui B dan C
```

```
[-1,  2,  2]
```

Hitung garis tegak lurus melalui A pada BC.

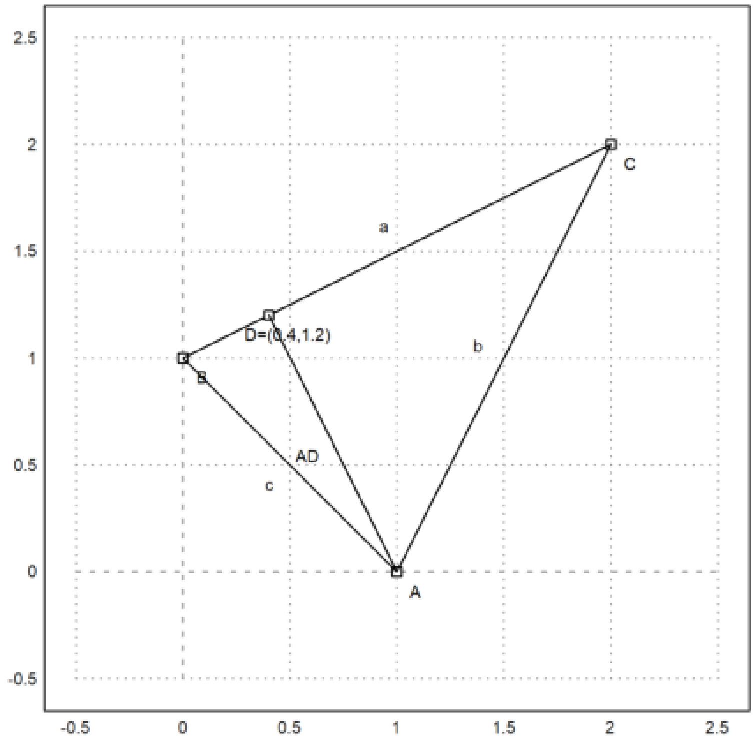
```
>h=perpendicular(A,lineThrough(B,C)); // garis h tegak lurus BC melalui A
```

Dan perpotongannya 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:

$$L_{\triangle ABC} = \frac{1}{2}AD.BC.$$

```
>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 langusng dengan fungsi
```

    1.5

Cara lain menghitung luas segitigas ABC:

```
>distance(A,D)*distance(B,C)/2
```

    1.5

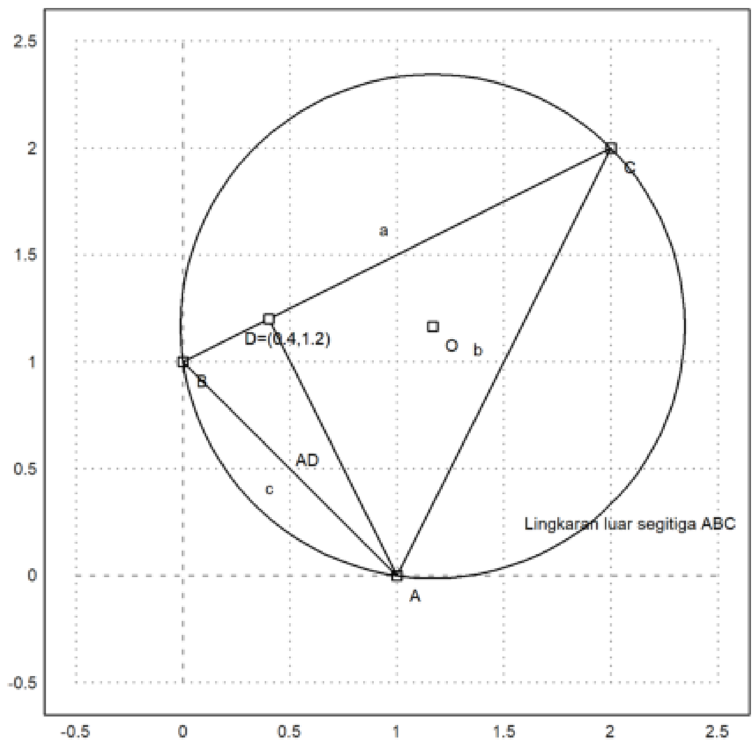
sudut di C.

```
>degprint(computeAngle(B,C,A))
```

```
36°52'11.63''
```

Sekarang lingkaran sirkit 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
```

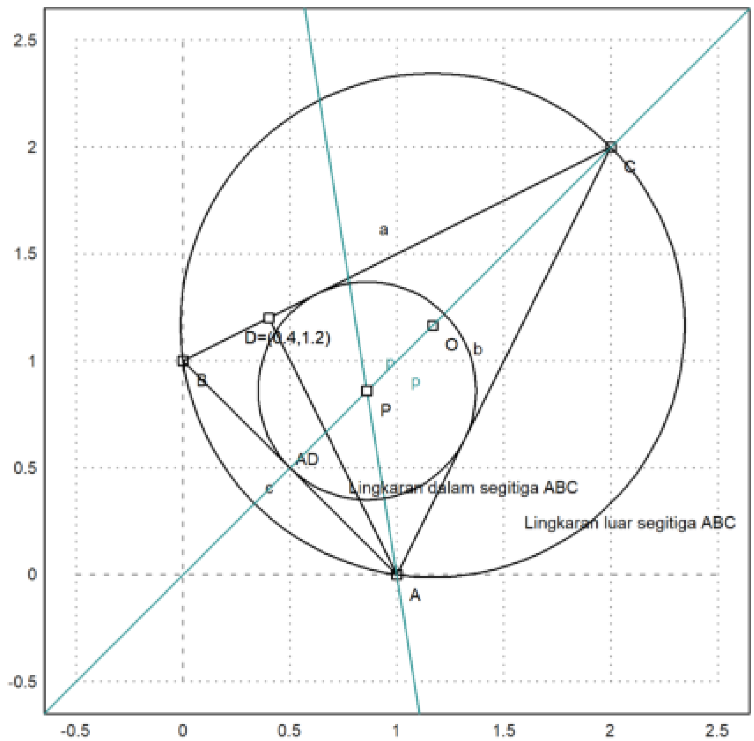
```
[0.86038,  0.86038]
```

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 dal
```



Latihan

1. Tentukan ketiga titik singgung lingkaran dalam dengan sisi-sisi segitiga ABC.
2. Gambar segitiga dengan titik-titik sudut ketiga titik singgung tersebut.
3. Tunjukkan bahwa garis bagi sudut yang ke tiga juga melalui titik pusat lingkaran dalam.
4. Gambar jari-jari lingkaran dalam.

Jawab :

1.) Titik singgung BC dengan lingkaran dalam

```
>s=lineThrough(B,C)
```

```
[-1,  2,  2]
```

```
>m=circleWithCenter(P,r)
```

```
[0.86038,  0.86038,  0.509654]
```

```
>S=lineCircleIntersections(s,m)
```

```
[0.632456,  1.31623]
```

Titik singgung garis AC dengan lingkaran dalam

```
>p=lineThrough(A,C)
```

```
[-2,  1,  -2]
```

```
>Q=lineCircleIntersections(p,m)
```

```
[1.31623,  0.632456]
```

```
>q=lineThrough(A,B)
```

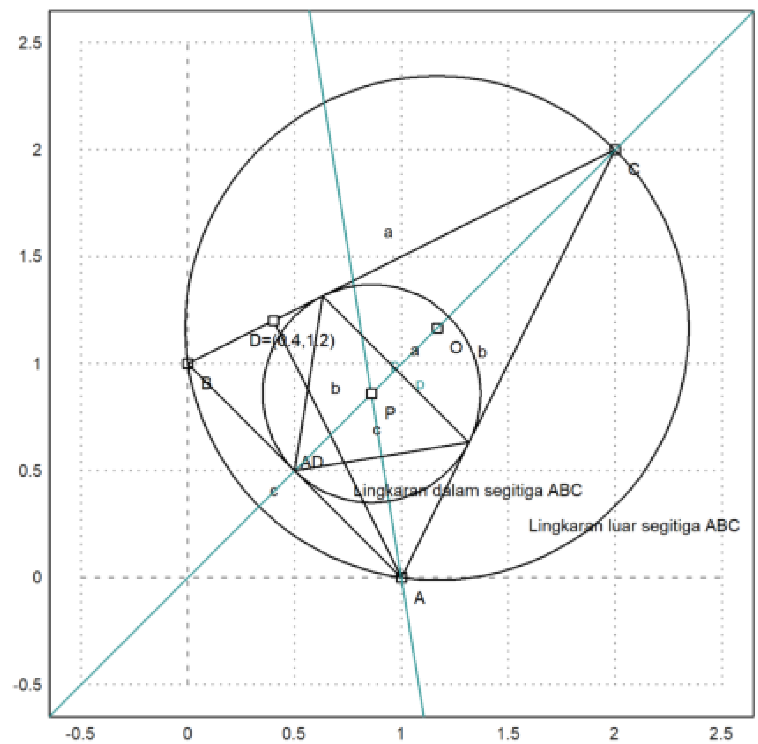
```
[-1,  -1,  -1]
```

```
>L=lineCircleIntersections(q,m)
```

```
[0.5,  0.5]
```

jadi titik singgung lingkaran dalam dengan sisi-sisi segitiga adalah
(0.632456, 1.31623) , (1.31623,0.632456), dan (0.5,0.5)
2.)

```
>plotSegment(S,Q,"a");
>plotSegment(S,L,"b");
>plotSegment(L,Q,"c"):
```



3.)

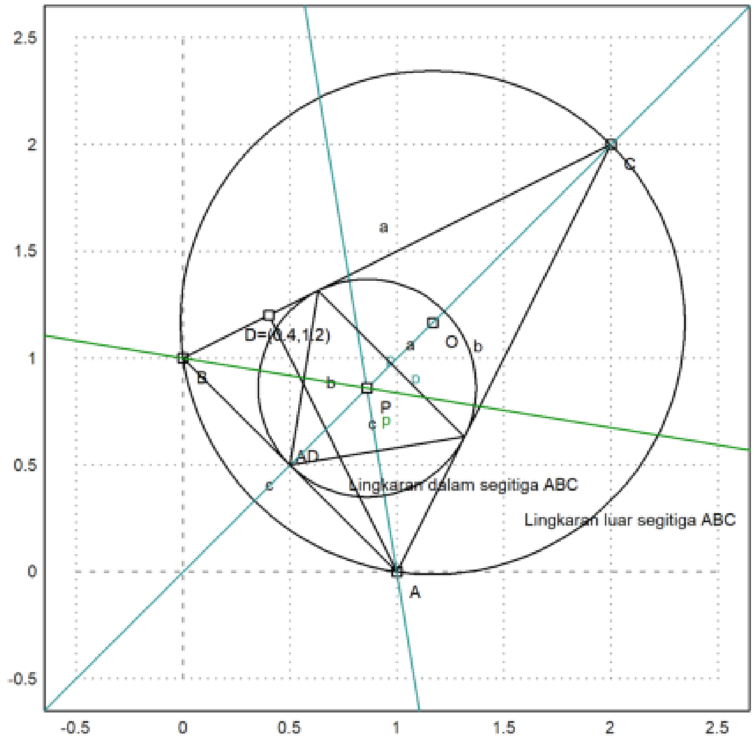
```
>P, r
```

```
[0.86038,  0.86038]
0.509653732104
```

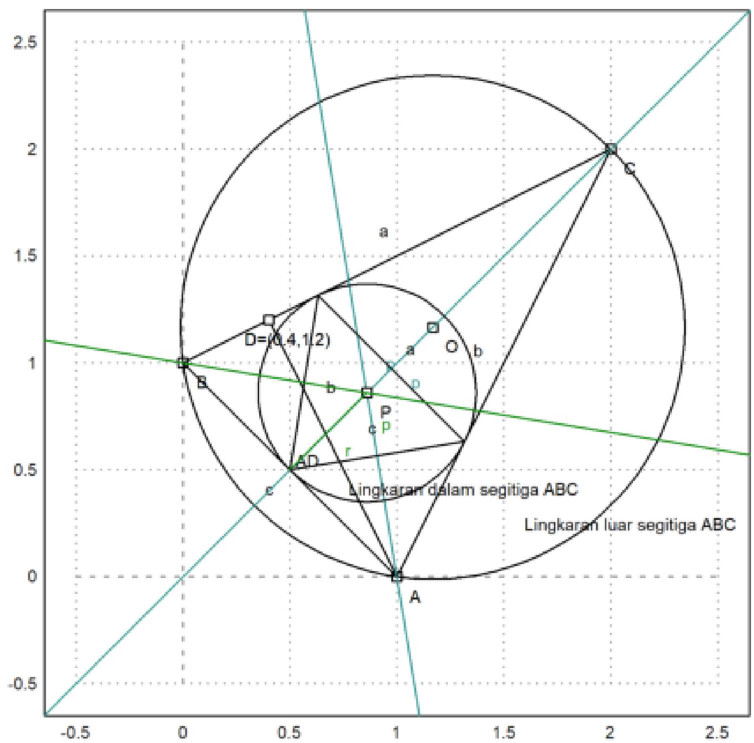
```
>k=angleBisector(A,B,C)
```

```
[-0.264911,  -1.63246,  -1.63246]
```

```
>color(3); plotLine(k):
```




```
>plotSegment(P,L,"r"):
```



Contoh 2: Geometri Smbolik

Kita dapat menghitung geometri tepat dan simbolis menggunakan Maxima.
Geometri file.e menyediakan fungsi yang sama (dan lebih banyak lagi) di Maxima. Namun, sekarang kita dapat menggunakan perhitungan simbolik.

```
>A &= [1,0]; B &= [0,1]; C &= [2,2]; // menentukan tiga titik A, B, C
```

Fungsi garis dan lingkaran bekerja seperti fungsi Euler, tetapi menyediakan penghitungan simbolik.

```
>c &= lineThrough(B,C) // c=BC
```

```
[- 1, 2, 2]
```

Kita bisa mendapatkan persamaan untuk sebuah garis dengan mudah.

```
>$getLineEquation(c,x,y), $solve(%,y) | expand // persamaan garis c
```

$$\left[y=\frac{x}{2}+1\right]$$

$$\left[y=\frac{x}{2}+1\right]$$

```
>$getLineEquation(lineThrough(A,[x1,y1]),x,y) // persamaan garis melalui A dan (x1, y1)
```

$$(x_1-1)\ y-x\ y_1=-y_1$$

```
>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
```

```
 2   6
[-,  -]
 5   5
```

```
>$projectToLine(A,lineThrough(B,C)) // proyeksi A pada BC
```

$$\left[\frac{2}{5},\frac{6}{5}\right]$$

```
>$distance(A,Q) // jarak AQ
```

$$\frac{3}{\sqrt{5}}$$

```
>cc &= circleThrough(A,B,C); $cc // (titik pusat dan jari-jari) lingkaran melalui A, B, C
```

$$\left[\frac{7}{6}, \frac{7}{6}, \frac{5}{3\sqrt{2}}\right]$$

```
>r&=getCircleRadius(cc); $r , $float(r) // tampilkan nilai jari-jari
```

$$1.178511301977579$$

```
>$computeAngle(A,C,B) // nilai <ACB
```

$$\arccos\left(\frac{4}{5}\right)$$

```
>$solve(getLineEquation(angleBisector(A,C,B),x,y),y)[1] // persamaan garis bagi <ACB
```

$$y = x$$

```
>P &= lineIntersection(angleBisector(A,C,B),angleBisector(C,B,A)); $P // titik potong 2 ga
```

$$\left[\frac{\sqrt{2}\sqrt{5}+2}{6}, \frac{\sqrt{2}\sqrt{5}+2}{6}\right]$$

```
>P() // hasilnya sama dengan perhitungan sebelumnya
```

```
[0.86038,  0.86038]
```

Garis dan Lingkaran yang Berpotongan

Tentu saja, kita juga bisa memotong garis dengan lingkaran, dan lingkaran dengan lingkaran.

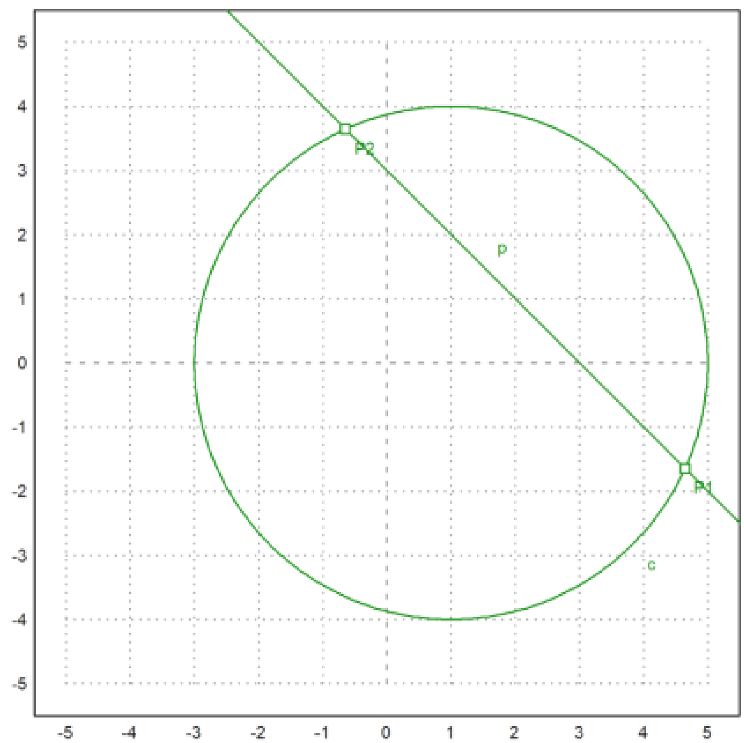
```
>A &:= [1,0]; c=circleWithCenter(A,4);
>B &:= [1,2]; C &:= [2,1]; l=lineThrough(B,C);
>setPlotRange(5); plotCircle(c); plotLine(l);
```

Perpotongan garis dengan lingkaran mengembalikan dua titik dan jumlah titik perpotongan.

```
>{P1,P2,f}=lineCircleIntersections(l,c);
>P1, P2,
```

```
  [4.64575,  -1.64575]
  [-0.645751,  3.64575]
```

```
>plotPoint(P1); plotPoint(P2):
```



Hal yang sama 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
```

$$\left[\left[\sqrt{7}+2,1-\sqrt{7}\right],\left[2-\sqrt{7},\sqrt{7}+1\right]\right]$$

Akan ditunjukkan bahwa sudut-sudut yang menghadap bsuusr yang sama adalah sama besar.

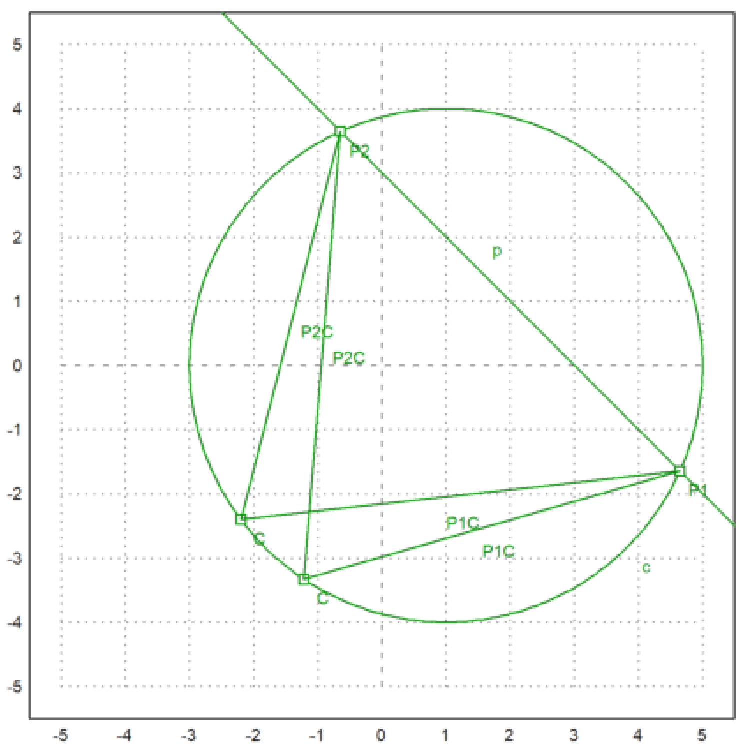
```
>C=A+normalize([-2,-3])*4; plotPoint(C); plotSegment(P1,C); plotSegment(P2,C);
>degprint(computeAngle(P1,C,P2))
```

```
69°17'42.68''
```

```
>C=A+normalize([-4,-3])*4; plotPoint(C); plotSegment(P1,C); plotSegment(P2,C);
>degprint(computeAngle(P1,C,P2))
```

```
69°17'42.68''
```

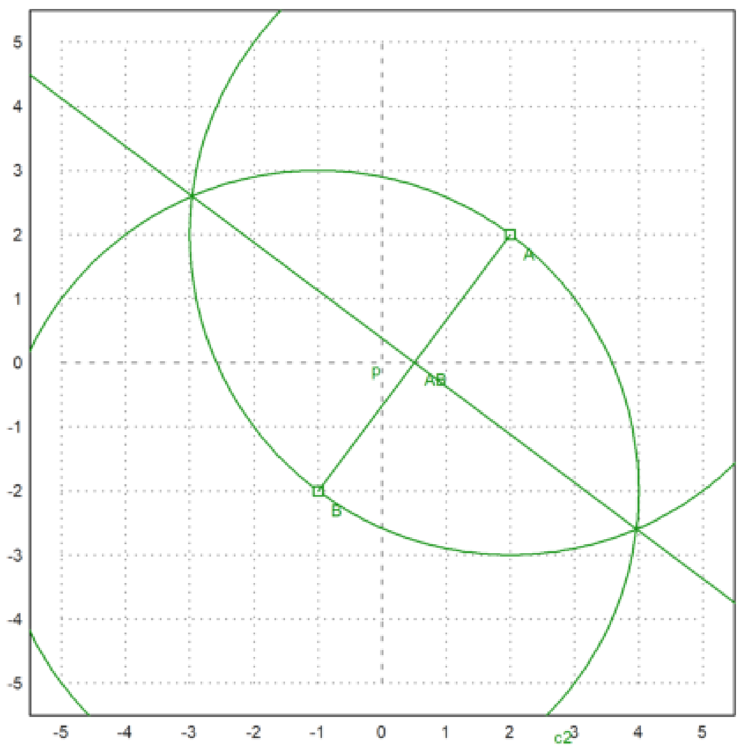
```
>insimg;
```



Garis Sumbu

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 melallui 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. Tapi kita bisa menyederhanakan, jika kita menyelesaikan y.

```
>g &= getLineEquation(lineThrough(P1,P2),x,y);
>$solve(g,y)
```

$$\left[y=\frac{-\left(2\,b_1-2\,a_1\right)\,x+b_2^2+b_1^2-a_2^2-a_1^2}{2\,b_2-2\,a_2}\right]$$

Ini memang sama dengan tengah tegak lurus, yang dihitung dengan cara yang sama sekali berbeda.

```
>$solve(getLineEquation(middlePerpendicular(A,B),x,y),y)
```

$$\left[y=\frac{-\left(2\,b_1-2\,a_1\right)\,x+{b_2}^2+{b_1}^2-{a_2}^2-{a_1}^2}{2\,b_2-2\,a_2}\right]$$

```
>h &=getLineEquation(lineThrough(A,B),x,y);
>$solve(h,y)
```

$$\left[y=\frac{\left(b_2-a_2\right)\,x-a_1\,b_2+a_2\,b_1}{b_1-a_1}\right]$$

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:

$$x^2+y^2=b^2,\quad (x-a)^2+y^2=c^2.$$

```
>sol &= solve([x^2+y^2=b^2,(x-a)^2+y^2=c^2],[x,y])
```

```
[]
```

Ekstrak larutan y.

```
>ysol &= y with sol[2][2]; $ysol
```

```
Maxima said:
part: invalid index of list or matrix.
 -- an error. To debug this try: debugmode(true);

Error in:
ysol &= y with sol[2][2]; $ysol ...
                         ^
```

Kami mendapatkan formula Heron.

```
>function H(a,b,c) &= sqrt(factor((ysol*a/2)^2)); $'H(a,b,c)=H(a,b,c)
```

$$H\left(a,b,[1,0,4]\right)=\frac{\left|a\right|\,\left|ysol\right|}{2}$$

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 abs(a)*abs(ysol)/2
Error in:
H(3,4,5) //luas segitiga siku-siku dengan panjang sisi 3, 4, 5 ...
        ^
```

Dan jelas juga, bahwa ini adalah segitiga dengan luas maksimal dan kedua sisinya 3 dan 4.

```
>aspect (1.5); plot2d(&H(3,4,x),1,7): // Kurva luas segitiga sengan panjang sisi 3, 4, x (
```

```
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 elips.

```
>s1 &= subst(d-c,b,sol[2]); $s1
```

```
Maxima said:
part: invalid index of list or matrix.
 -- an error. To debug this try: debugmode(true);
```

```
Error in:
s1 &= subst(d-c,b,sol[2]); $s1 ...
                          ^
```

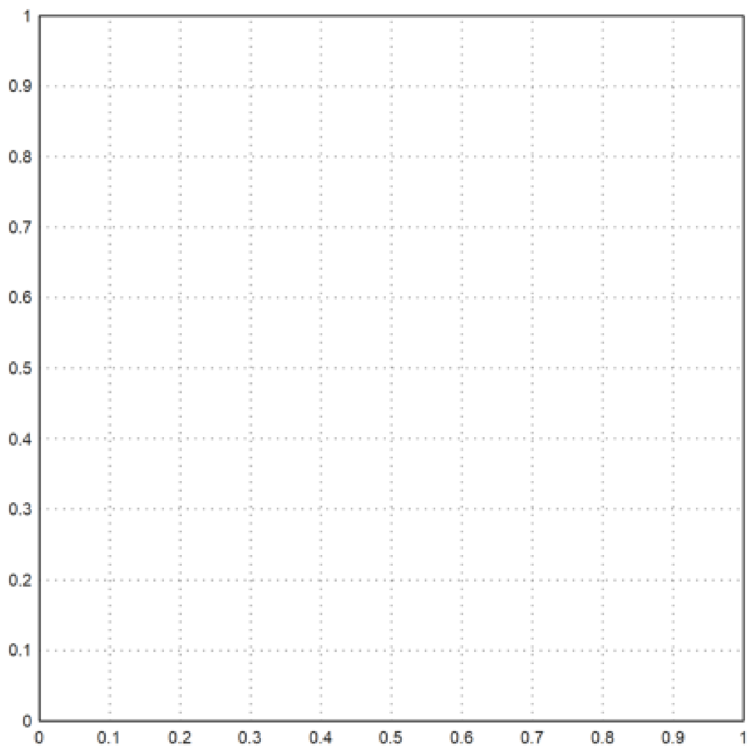
Dan manfaatkan ini.

```
>function fx(a,c,d) &= rhs(s1[1]); $fx(a,c,d), function fy(a,c,d) &= rhs(s1[2]); $fy(a,c,d
```

$$0$$

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 elips ini, yaitu.

$$\frac{(x-x_m)^2}{u^2}+\frac{(y-y_m)}{v^2}=1,$$

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])
```

$$\frac{a^2}{d^2}$$

Kita melihat bahwa tinggi dan luas segitiga adalah maksimal untuk x = 0. Jadi luas segitiga dengan a + b + c = d adalah maksimal, jika sama sisi. Kami ingin mendapatkan 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
```

$$\left[\frac{a\,ysol^2}{2}=0,0=0\right]$$

Kami mendapatkan beberapa minima, yang termasuk dalam segitiga dengan satu sisi 0, dan solusi a = b = c = d / 3.

```
>$solve(eqns,[a,b])
```

$$[[a=0,b=\%r_1]]$$

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])
```

```
Maxima said:
diff: second argument must be a variable; found [1,0,4]
 -- an error. To debug this try: debugmode(true);

Error in:
... la,    diff(H(a,b,c)^2,c)=la,a+b+c=d],[a,b,c,la]) ...
                                                     ^
```

Kita bisa membuat plot situasinya

Pertama, atur poin di Maxima.

```
>A &= at([x,y],sol[2]); $A
```

```
Maxima said:
part: invalid index of list or matrix.
 -- 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
```

$$[a,0]$$

Kemudian atur rentang plot, dan plot poinnya.

```
>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 segmennya.

```
>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 tengah tegak lurus di Maxima.

```
>h &= middlePerpendicular(A,B); g &= middlePerpendicular(B,C);
```

Dan bagian tengah dari keliling.

```
>U &= lineIntersection(h,g);
```

Kami mendapatkan rumus untuk jari-jari lingkaran sunat.

```
>&assume(a>0,b>0,c>0); $distance(U,B) | radcan
```

$$\frac{\sqrt{a_2^2 + a_1^2}\,\sqrt{a_2^2 + a_1^2 - 2\,a\,a_1 + a^2}}{2\,|a_2|}$$

Mari kita tambahkan ini ke plot.

```
>plotPoint(U()); ...
>plotCircle(circleWithCenter(mxmeval("U"),mxmeval("distance(U,C)"))):
```

```
Variable a2 not found!
Use global variables or parameters for string evaluation.
Error in ^
Error in expression: [a/2,(a2^2+a1^2-a*a1)/(2*a2)]
Error in:
plotPoint(U()); plotCircle(circleWithCenter(mxmeval("U"),mxmev ...
                 ^
```

Menggunakan geometri, kami mendapatkan rumus sederhana

$$\frac{a}{\sin(\alpha)} = 2r$$

untuk radius. Kami dapat memeriksa, apakah ini benar dengan Maxima. Maxima akan memfaktorkannya hanya jika kita mengkuadratkannya.

```
>$c^2/sin(computeAngle(A,B,C))^2  | factor
```

$$\left[\frac{a_2^2 + a_1^2}{a_2^2}, 0, \frac{16\left(a_2^2 + a_1^2\right)}{a_2^2}\right]$$

Contoh 4: Garis Euler dan Parabola

Garis euler adalah garis yang ditentukan dari segitiga yang tidak sama sisi. Ini adalah garis tengah segitiga, dan melewati beberapa titik penting yang ditentukan dari segitiga, termasuk pusat ortosentrum, sirkumenter, pusat massa, titik Exeter, dan pusat lingkaran sembilan titik segitiga.
Untuk demonstrasi, kami menghitung dan memplot garis Euler dalam segitiga.
Pertama, kami menentukan sudut segitiga di Euler. Kami menggunakan definisi, yang terlihat dalam ekspresi simbolik.

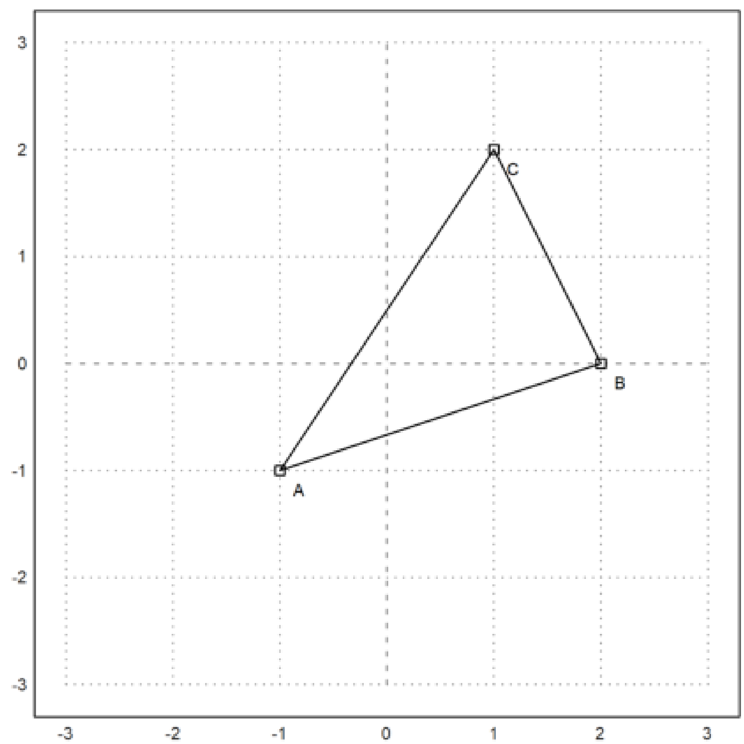
```
>A::=[-1,-1]; B::=[2,0]; C::=[1,2];
```

Untuk memplot objek geometris, kami menyiapkan area plot, dan menambahkan poin ke dalamnya. 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)
```

$$-\frac{7}{2}$$

Kita dapat menghitung koefisien dari sisi c.

```
>c &= lineThrough(A,B)
```

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

Dan juga dapatkan rumus untuk baris ini.

```
>$getLineEquation(c,x,y)
```

$$3\,y-x=-2$$

Untuk bentuk Hesse, kita perlu menentukan 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]])
```

$$\frac{7}{\sqrt{10}}$$

$\frac{7}{\sqrt{10}}$

Sekarang kami menghitung sirkit ABC.

```
>LL &= circleThrough(A,B,C); $getCircleEquation(LL,x,y)
```

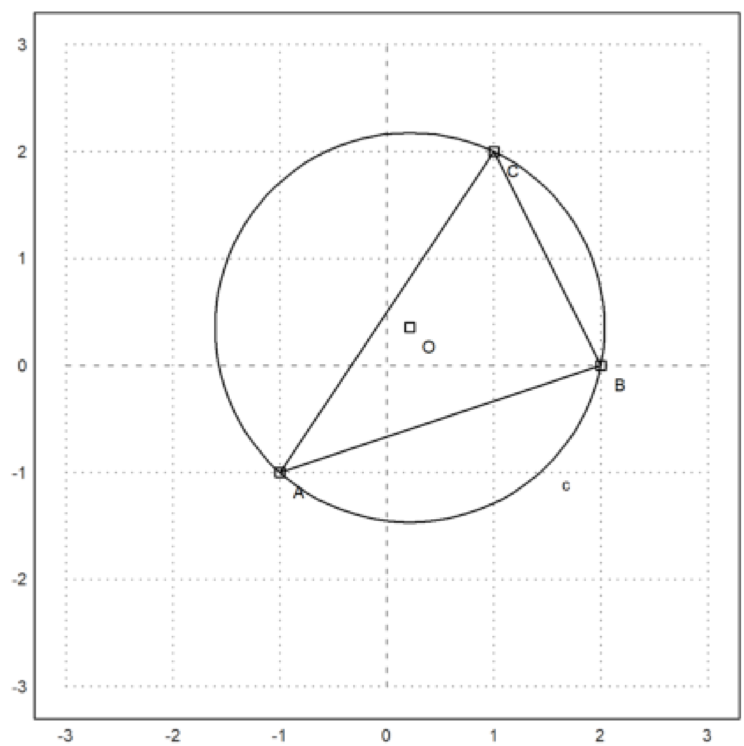
$$\left(y-\frac{5}{14}\right)^2+\left(x-\frac{3}{14}\right)^2=\frac{325}{98}$$

```
>O &= getCircleCenter(LL); $O
```

$$\left[\frac{3}{14},\frac{5}{14}\right]$$

Plot lingkaran dan pusatnya. Cu dan U adalah simbolik. Kami mengevaluasi ekspresi ini untuk Euler.

```
>plotCircle(LL()); plotPoint(O(),"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
```

$$\left[\frac{11}{7}, \frac{2}{7}\right]$$

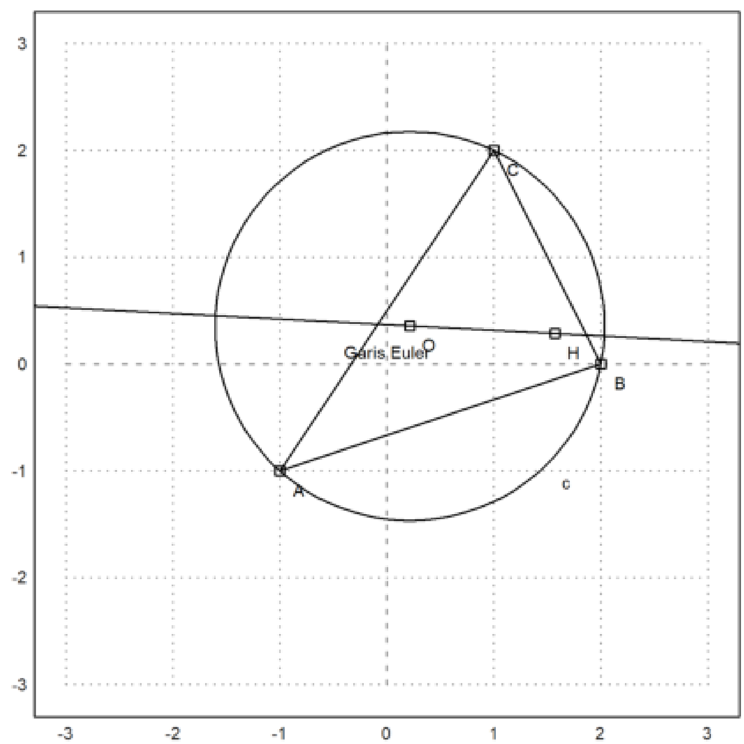
Sekarang kita dapat menghitung garis Euler dari segitiga tersebut.

```
>el &= lineThrough(H,O); $getLineEquation(el,x,y)
```

$$-\frac{19\,y}{14} - \frac{x}{14} = -\frac{1}{2}$$

Tambahkan ke plot kita.

```
>plotPoint(H(),"H"); plotLine(el(),"Garis Euler"):
```

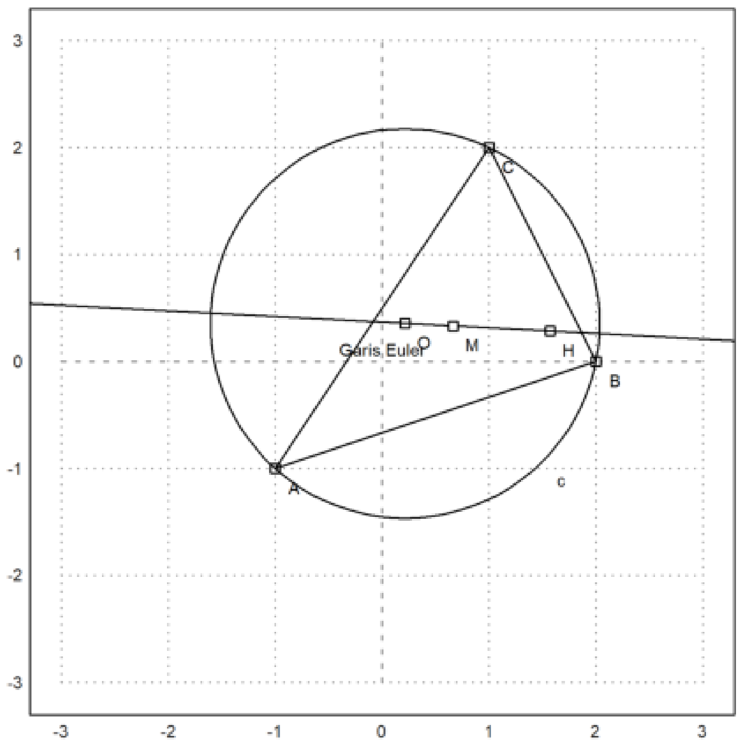


Pusat gravitasi harus berada di garis ini.

```
>M &= (A+B+C)/3; $getLineEquation(el,x,y) with [x=M[1],y=M[2]]
```

$$-\frac{1}{2} = -\frac{1}{2}$$

```
>plotPoint(M(),"M"): // titik berat
```



Teorinya mengatakan bahwa MH = 2 * MO. Kita perlu menyederhanakan dengan radcan untuk mencapai ini.

```
>$distance(M,H)/distance(M,O)|radcan
```

$$2$$

Fungsinya termasuk fungsi untuk sudut juga.

```
>$computeAngle(A,C,B), degprint(%())
```

$$\arccos\left(\frac{4}{\sqrt{5}\,\sqrt{13}}\right)$$

```
60°15'18.43''
```

Persamaan untuk pusat lingkaran tidak terlalu bagus.

```
>Q &= lineIntersection(angleBisector(A,C,B),angleBisector(C,B,A))|radcan; $Q
```

$$\left[\frac{\left(2^{\frac{3}{2}}+1\right)\,\sqrt{5}\,\sqrt{13}-15\,\sqrt{2}+3}{14},\frac{\left(\sqrt{2}-3\right)\,\sqrt{5}\,\sqrt{13}+5\,2^{\frac{3}{2}}+5}{14}\right]$$

Mari kita hitung juga ekspresi jari-jari lingkaran yang tertulis.

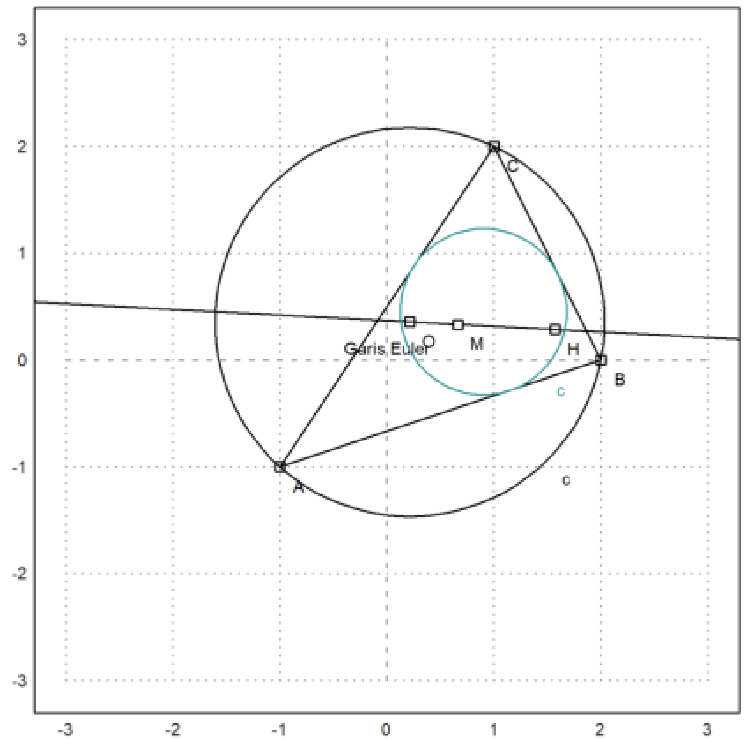
```
>r &= distance(Q,projectToLine(Q,lineThrough(A,B)))|ratsimp; $r
```

$$\frac{\sqrt{\left(-41\,\sqrt{2}-31\right)\,\sqrt{5}\,\sqrt{13}+115\,\sqrt{2}+614}}{7\,\sqrt{2}}$$

```
>LD &=  circleWithCenter(Q,r); // Lingkaran dalam
```

Mari kita tambahkan ini ke plot.

```
>color(5); plotCircle(LD()):
```



Parabola

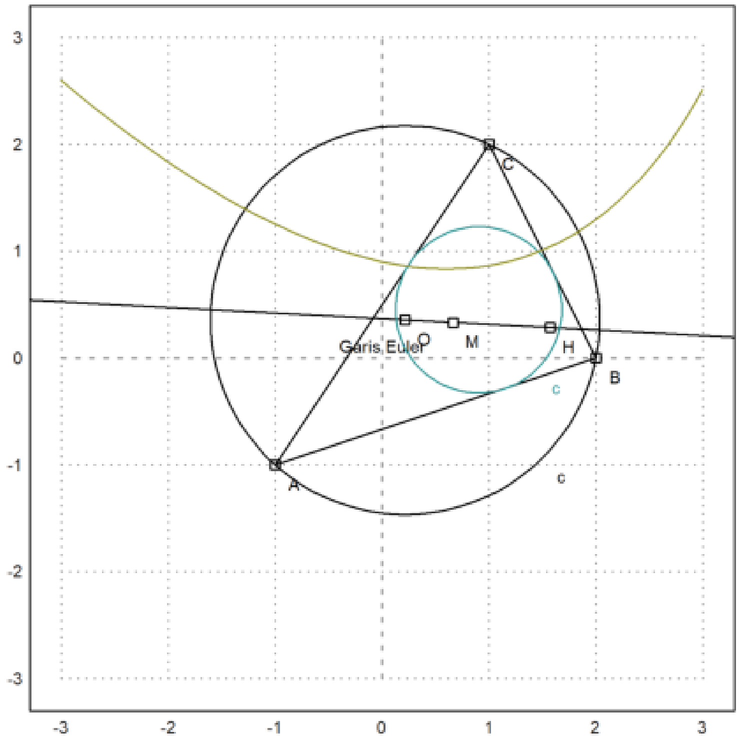
Selanjutnya akan dicari persamaan tempat kedudukan titik-titik yang berjarak sama ke titik C dan ke garis AB.

```
>p &= getHesseForm(lineThrough(A,B),x,y,C)-distance([x,y],C); $p='0
```

$$\frac{3\,y-x+2}{\sqrt{10}}-\sqrt{\left(2-y\right)^2+\left(1-x\right)^2}=0$$

Persamaan tersebut dapat digambar menjadi satu dengan gambar sebelumnya.

```
>plot2d(p,level=0,add=1,contourcolor=6):
```



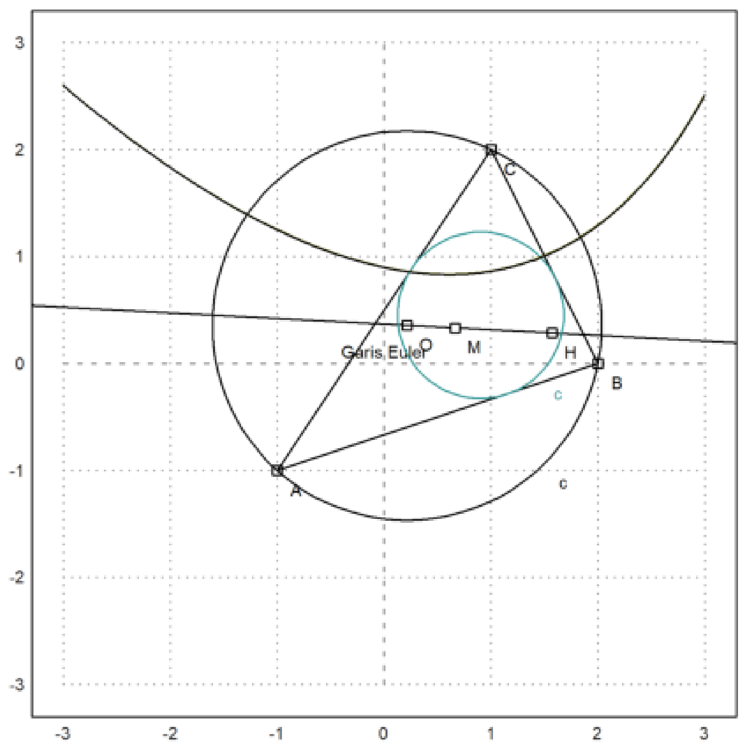
This should be some function, but the default solver of Maxima can find the solution only, if we square the equation. Consequently, we get a fake solution.

```
>akar &= solve(getHesseForm(lineThrough(A,B),x,y,C)^2-distance([x,y],C)^2,y)
```

```
[y = - 3 x - sqrt(70) sqrt(9 - 2 x) + 26,
                      y = - 3 x + sqrt(70) sqrt(9 - 2 x) + 26]
```

The first solution is
maxima: akar[1]
Adding the first solution to the plot show, that it is indeed the path we are looking for. The theory tells us that it is a rotated parabola.

```
>plot2d(&rhs(akar[1]),add=1):
```




```
>function g(x) &= rhs(akar[1]); $'g(x)= g(x)// fungsi yang mendefinisikan kurva di atas
```

$$g(x) = -3\,x - \sqrt{70}\,\sqrt{9 - 2\,x} + 26$$

```
>T &=[-1, g(-1)]; // ambil sebarang titik pada kurva tersebut
>dTC &= distance(T,C); $fullratsimp(dTC), $float(%) // jarak T ke C
```

$$2.135605779339061$$

```
>U &= projectToLine(T,lineThrough(A,B)); $U // proyeksi T pada garis AB
```

$$\left[\frac{80 - 3\,\sqrt{11}\,\sqrt{70}}{10}, \frac{20 - \sqrt{11}\,\sqrt{70}}{10}\right]$$

```
>dU2AB &= distance(T,U); $fullratsimp(dU2AB), $float(%) // jatak T ke AB
```

$$2.135605779339061$$

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.

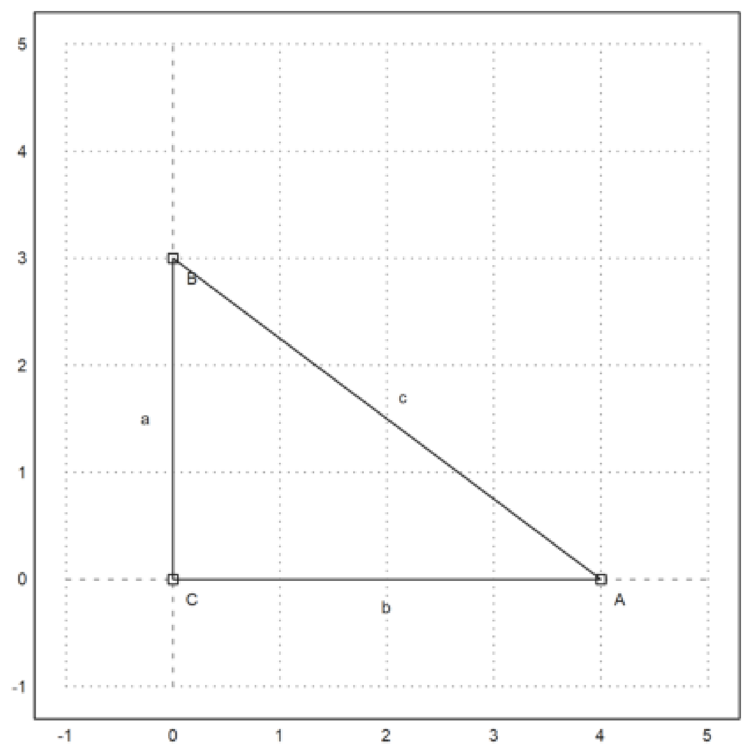
Contoh 5: Trigonometri Rasional

Ini terinspirasi oleh ceramah N.J.Wildberger. Dalam bukunya "Proporsi Agung", Wildberger mengusulkan untuk menggantikan pengertian klasik tentang jarak dan sudut dengan kuadransi 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 dapat dilakukan dengan kertas dan pensil saja. 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 mengevaluasi ke pendekatan numerik saja.

```
>load geometry;
```

Untuk pendahuluan pertama, kami menggunakan segitiga persegi panjang dengan proporsi Mesir terkenal 3, 4 dan 5. Perintah berikut adalah perintah Euler untuk memplot 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 wa adalah sudut di A. Cara biasa untuk menghitung sudut ini, adalah dengan melakukan invers dari fungsi sinus. Hasilnya adalah sudut yang tidak dapat dicerna, yang hanya dapat dicetak secara perkiraan.

```
>wa := arcsin(3/5); degprint(wa)
```

```
36°52'11.63''
```

Trigonometri rasional mencoba menghindari hal ini.
Pengertian pertama dari trigonometri rasional adalah kuadran, yang menggantikan jarak. Faktanya, itu hanyalah kuadrat jarak. Berikut ini, a, b, dan c menunjukkan kuadran sisi-sisinya.
Teorema Pythogoras menjadi a + b = c lalu.

```
>a &= 3^2; b &= 4^2; c &= 5^2; &a+b=c
```

```
25 = 25
```

Gagasan kedua dari trigonometri rasional adalah penyebarannya. Spread mengukur bukaan antar baris. Ini adalah 0, jika garis sejajar, dan 1, jika garis persegi panjang. Ini adalah kuadrat dari 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 segitiga persegi panjang mana pun dengan satu sudut di A.

```
>sa &= a/c; $sa
```

$$\frac{9}{25}$$

Ini lebih mudah dihitung daripada sudut, tentu saja. Tetapi Anda kehilangan properti yang sudut dapat ditambahkan dengan mudah.
Tentu saja, kita dapat mengubah nilai perkiraan sudut wa menjadi sprad, dan mencetaknya sebagai pecahan.

```
>fracprint(sin(wa)^2)
```

```
9/25
```

Hukum kosinus dari trgonometri 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 sebaran di sudut A. Sisi a, seperti biasa, berlawanan dengan sudut A.
Hukum ini diimplementasikan dalam file geometry.e yang kami muat ke Euler.

```
>$crosslaw(aa,bb,cc,saa)
```

$$\left[\left(bb-aa+\frac{7}{6}\right)^2,\left(bb-aa+\frac{7}{6}\right)^2,\left(bb-aa+\frac{5}{3\sqrt{2}}\right)^2\right]=\left[\frac{14\,bb\,(1-saa)}{3},\frac{14\,bb\,(1-saa)}{3},\frac{5\,2^{\frac{3}{2}}\,bb\,(1-saa)}{3}\right]$$

Dalam kasus kami, kami mendapatkan

```
>$crosslaw(a,b,c,sa)
```

$$1024=1024$$

Mari kita gunakan crosslaw ini untuk mencari sebaran di A. Untuk melakukan ini, kita menghasilkan crosslaw untuk kuadran a, b, dan c, dan menyelesaikannya untuk sebaran yang tidak diketahui sa.
Anda dapat melakukan ini dengan tangan dengan mudah, tetapi saya menggunakan Maxima. Tentu saja, kami mendapatkan hasilnya, kami sudah mendapatkannya.

```
>$crosslaw(a,b,c,x), $solve(%,x)
```

$$\left[x=\frac{9}{25}\right]$$

$$\left[x=\frac{9}{25}\right]$$

Kami sudah tahu ini. Definisi penyebaran adalah kasus khusus dari hukum lintas hukum.
Kita juga bisa menyelesaikan ini untuk umum a, b, c. Hasilnya adalah rumus yang menghitung sebaran sudut segitiga berdasarkan kuadran ketiga sisinya.

```
>$solve(crosslaw(aa,bb,cc,x),x)
```

$$\left[\left[\frac{168\,bb\,x+36\,bb^2+(-72\,aa-84)\,bb+36\,aa^2-84\,aa+49}{36},\frac{168\,bb\,x+36\,bb^2+(-72\,aa-84)\,bb+36\,aa^2-84\,aa+49}{36},\right.\right.$$

Kita bisa membuat fungsi dari hasilnya. Fungsi seperti itu sudah ditentukan dalam file geometry.e Euler.

```
>$spread(a,b,c)
```

$$\frac{9}{25}$$

Sebagai contoh, kita bisa menggunakannya untuk menghitung sudut segitiga bersisi

$$a,\quad a,\quad \frac{4a}{7}$$

Hasilnya rasional, yang tidak mudah didapat jika kita menggunakan trigonometri klasik.

```
>$spread(a,a,4*a/7)
```

$$\frac{6}{7}$$

Ini adalah sudut dalam derajat.

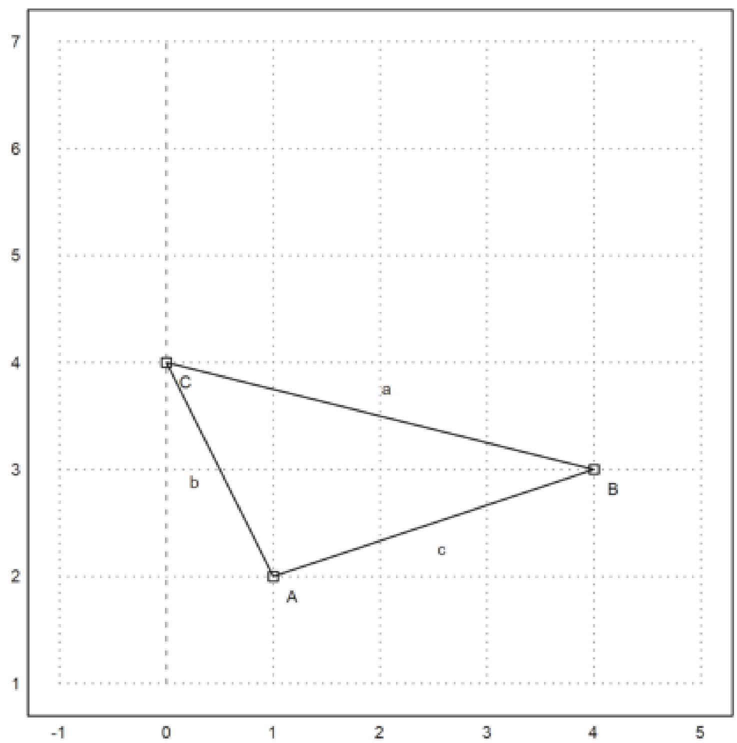
```
>degprint(arcsin(sqrt(6/7)))
```

```
67°47'32.44''
```

Contoh lain

Sekarang, mari kita coba contoh yang lebih canggih.
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 Pythogoras, 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)
```

$$\sqrt{10}$$

Euler juga memiliki fungsi kuadrans antara dua titik.
Dalam contoh berikut, karena c + b bukan a, segitiga tidak persegi panjang.

```
>c &= quad(A,B); $c, b &= quad(A,C); $b, a &= quad(B,C); $a,
```

$$17$$

Pertama, mari kita hitung sudut tradisional. Fungsi computeAngle menggunakan metode biasa berdasarkan perkalian titik dari dua vektor. Hasilnya adalah beberapa pendekatan floating point.

```
>wb &= computeAngle(A,B,C); $wb, $(wb/pi*180)()
```

$$\arccos\left(\frac{11}{\sqrt{10}\,\sqrt{17}}\right)$$

```
32.4711922908
```

Menggunakan pensil dan kertas, kita bisa melakukan hal yang sama dengan hukum silang. Kami memasukkan kuadran a, b, dan c ke dalam hukum silang dan menyelesaikan untuk x.

```
>$crosslaw(a,b,c,x), $solve(%,x),
```

$$\left[x=\frac{49}{50}\right]$$

$$\left[x=\frac{49}{50}\right]$$

Artinya, fungsi penyebaran yang didefinisikan dalam "geometry.e" tidak.

```
>sb &= spread(b,a,c); $sb
```

$$\frac{49}{170}$$

Maxima mendapatkan hasil yang sama dengan menggunakan trigonometri biasa, jika kita memaksakannya. Itu menyelesaikan istilah sin (arccos (...)) menjadi hasil pecahan. Kebanyakan siswa tidak dapat melakukan ini.

```
>$sin(computeAngle(A,B,C))^2
```

$$\frac{49}{170}$$

Setelah kita mendapatkan sebaran di B, kita bisa menghitung tinggi ha di sisi a. Ingat bahwa

$$s_b=\frac{h_a}{c}$$

Menurut definisi.

```
>ha &= c*sb; $ha
```

$$\frac{49}{17}$$

Gambar berikut telah diproduksi dengan program geometri C.a.R., yang dapat menggambar kuadran dan menyebar.
gambar : (20) Rational_Geometry_CaR.png
Menurut definisi, panjang ha adalah akar kuadrat dari kuadrannya.

```
>$sqrt(ha)
```

$$\frac{7}{\sqrt{17}}$$

Sekarang kita bisa menghitung luas segitiga. Jangan lupa, bahwa kita berurusan dengan kuadran!

```
>$sqrt(ha)*sqrt(a)/2
```

$$\frac{7}{2}$$

Rumus determinan yang biasa menghasilkan hasil yang sama.

```
>$areaTriangle(B,A,C)
```

$$\frac{7}{2}$$

The Heron Formula

Sekarang, mari kita selesaikan masalah ini secara umum!

```
>&remvalue(a,b,c,sb,ha);
```

Pertama-tama kita menghitung spread di B untuk segitiga dengan sisi a, b, dan c. Kemudian kami menghitung luas area yang dikuadratkan ("kuadrea"?), Memfaktorkannya dengan Maxima, dan kami 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)
```

$$\frac{(-c+b+a)\,(c-b+a)\,(c+b-a)\,(c+b+a)}{16}$$

$$\frac{(-c+b+a)\,(c-b+a)\,(c+b-a)\,(c+b+a)}{16}$$

The Triple Spread Rule

Kerugian dari spread adalah bahwa mereka tidak lagi hanya menambahkan sudut serupa.
Namun, tiga sebaran segitiga memenuhi aturan "penyebaran rangkap tiga" berikut.

```
>&remvalue(sa,sb,sc); $triplespread(sa,sb,sc)
```

$$(sc+sb+sa)^2 = 2\left(sc^2+sb^2+sa^2\right)+4\,sa\,sb\,sc$$

Aturan ini berlaku untuk tiga sudut yang bertambah menjadi 180 ?.

$$\alpha+\beta+\gamma=\pi$$

Sejak penyebaran

$$\alpha, \pi-\alpha$$

sama, aturan penyebaran tiga kali lipat juga benar, jika

$$\alpha+\beta=\gamma$$

Karena penyebaran sudut negatif adalah sama, aturan penyebaran tiga kali lipat juga berlaku, jika

$$\alpha+\beta+\gamma=0$$

Misalnya, kita dapat menghitung sebaran sudut 60 ?. Ini 3/4. Persamaan memiliki solusi kedua, di mana semua spread adalah 0.

```
>$solve(triplespread(x,x,x),x)
```

$$\left[x=\frac{3}{4}, x=0\right]$$

Sebaran 90 ? jelaslah 1. Jika dua sudut dijumlahkan menjadi 90 ?, penyebarannya menyelesaikan persamaan penyebaran rangkap tiga dengan a, b, 1. Dengan perhitungan berikut kita mendapatkan a + b = 1.

```
>$triplespread(x,y,1), $solve(%,x)
```

$$[x=1-y]$$

Karena penyebaran 180 ? -t sama dengan penyebaran t, rumus penyebaran rangkap tiga juga berlaku, jika satu sudut adalah jumlah atau perbedaan dari dua sudut lainnya.
Jadi kita bisa menemukan sebaran sudut berlipat ganda. Perhatikan bahwa ada dua solusi lagi. Kami menjadikan ini sebuah fungsi.

```
>$solve(triplespread(a,a,x),x), function doublespread(a) &= factor(rhs(%[1]))
```

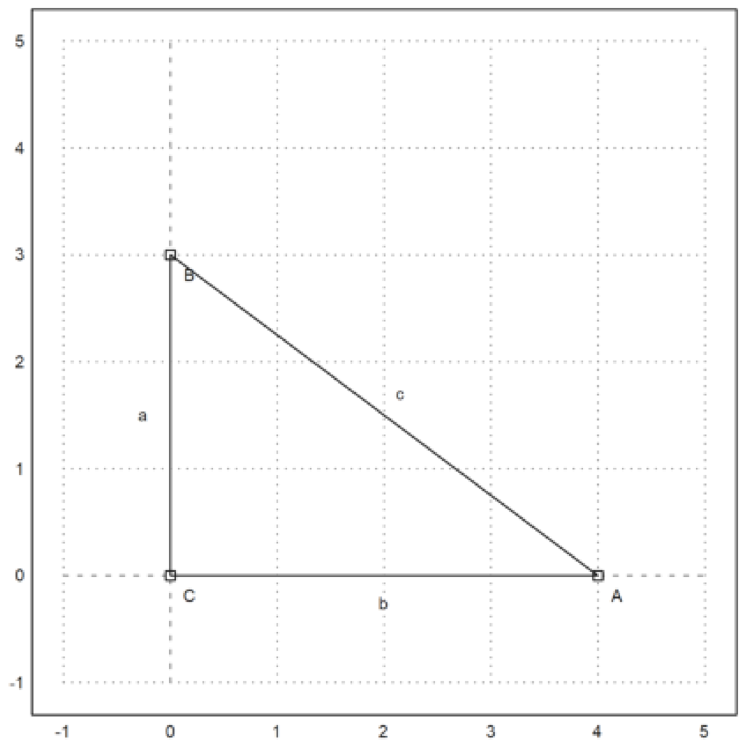
$$\left[x=4\,a-4\,a^2, x=0\right]$$

```
- 4 (a - 1) a
```

Angle Bisectors

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 bisektor sudut pada A. Tapi kita ingin menyelesaikannya untuk umum a, b, c.

```
>&remvalue(a,b,c);
```

Jadi pertama-tama kita menghitung sebaran sudut terbagi di A, menggunakan rumus sebaran rangkap tiga. Masalah dengan rumus ini muncul lagi. Ini memiliki dua solusi. Kami harus memilih yang benar. Solusi lainnya mengacu pada sudut terbagi 180 ? -wa.

```
>$triplespread(x,x,a/(a+b)), $solve(%,x), sa2 &= rhs(%[1]); $sa2
```

$$\frac{-\sqrt{b}\,\sqrt{b+a}+b+a}{2\,b+2\,a}$$

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

$$\frac{-\sqrt{b}\,\sqrt{b+a}+b+a}{2\,b+2\,a}$$

Mari kita periksa persegi panjang Mesir.

```
>$sa2 with [a=3^2,b=4^2]
```

$$\frac{1}{10}$$

Kami dapat mencetak sudut di Euler, setelah mentransfer penyebaran ke radian.

```
>wa2 := arcsin(sqrt(1/10)); degprint(wa2)
```

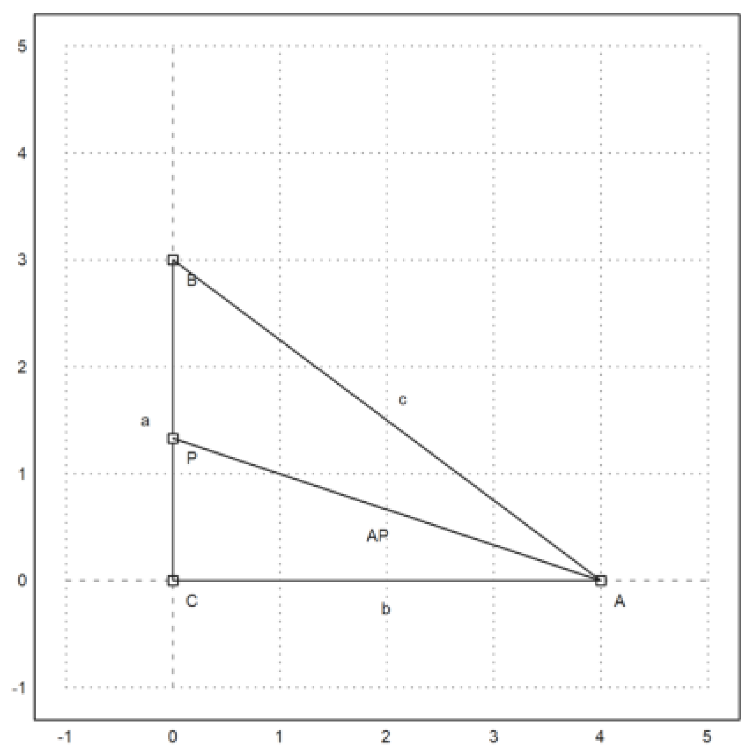
```
18°26'5.82''
```

Titik P adalah perpotongan dari 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 menghitung panjang bisektor AP.
Kami menggunakan teorema sinus di segitiga APC. Teorema ini menyatakan bahwa

$$\frac{BC}{\sin(w_a)} = \frac{AC}{\sin(w_b)} = \frac{AB}{\sin(w_c)}$$

memegang di segitiga apa pun. Persegi itu, itu diterjemahkan ke dalam apa yang disebut "hukum penyebaran"

$$\frac{a}{s_a} = \frac{b}{s_b} = \frac{c}{s_b}$$

dimana a, b, c menunjukkan qudrance.
Karena BPA sebaran adalah 1-sa2, kita dapatkan darinya bisa / 1 = b / (1-sa2) dan dapat menghitung bisa (kuadran garis-garis).

```
>&factor(ratsimp(b/(1-sa2))); bisa &= %; $bisa
```

$$\frac{2\,b\,\left(b+a\right)}{\sqrt{b}\,\sqrt{b+a}+b+a}$$

Mari kita periksa rumus ini untuk nilai Mesir kita.

```
>sqrt(mxmeval("at(bisa,[a=3^2,b=4^2])")), distance(A,P)
```

```
4.21637021356
4.21637021356
```

Kami juga dapat menghitung P menggunakan rumus spread.

```
>py&=factor(ratsimp(sa2*bisa)); $py
```

$$-\frac{b\,\left(\sqrt{b}\,\sqrt{b+a}-b-a\right)}{\sqrt{b}\,\sqrt{b+a}+b+a}$$

Nilainya sama dengan yang kita dapatkan dengan rumus trigonometri.

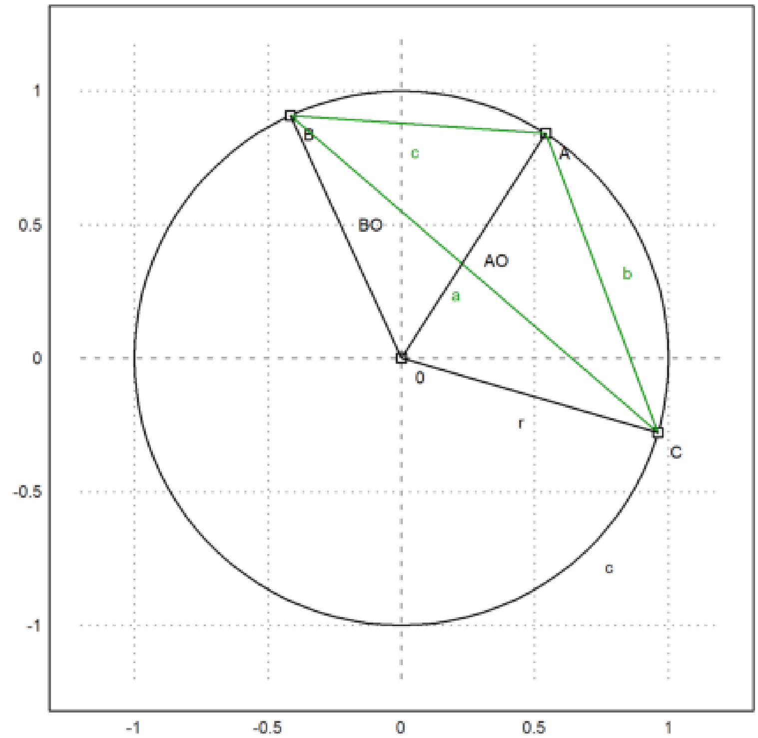
```
>sqrt(mxmeval("at(py,[a=3^2,b=4^2])"))
```

```
1.33333333333
```

The Chord Angle

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,"0"); ...
>plotSegment(A,O); plotSegment(B,O); plotSegment(C,O,"r"); ...
>insimg;
```



Kita bisa menggunakan Maxima untuk menyelesaikan rumus sebaran rangkap tiga untuk sudut di pusat O untuk r. Jadi kita mendapatkan rumus untuk jari-jari kuadrat dari keliling dalam hal kuadrat sisi.
Kali ini, Maxima menghasilkan beberapa angka nol yang 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
```

$$-\frac{a\,b\,c}{c^2-2\,b\,c+a\,\left(-2\,c-2\,b\right)+b^2+a^2}$$

Kita bisa menjadikannya sebagai fungsi Euler.

```
>function periradius(a,b,c) &= rabc;
```

Mari kita periksa hasilnya untuk poin A, B, C kita.

```
>a:=quadrance(B,C); b:=quadrance(A,C); c:=quadrance(A,B);
```

Radiusnya memang 1.

```
>periradius(a,b,c)
```

```
1
```

Faktanya, penyebaran CBA hanya bergantung pada b dan c. Ini adalah teorema sudut akord.

```
>$spread(b,a,c)*rabc | ratsimp
```

$$\frac{b}{4}$$

Sebenarnya sebarannya adalah b / (4r), dan kita melihat bahwa sudut akor b adalah setengah dari sudut tengah.

```
>$doublespread(b/(4*r))-spread(b,r,r) | ratsimp
```

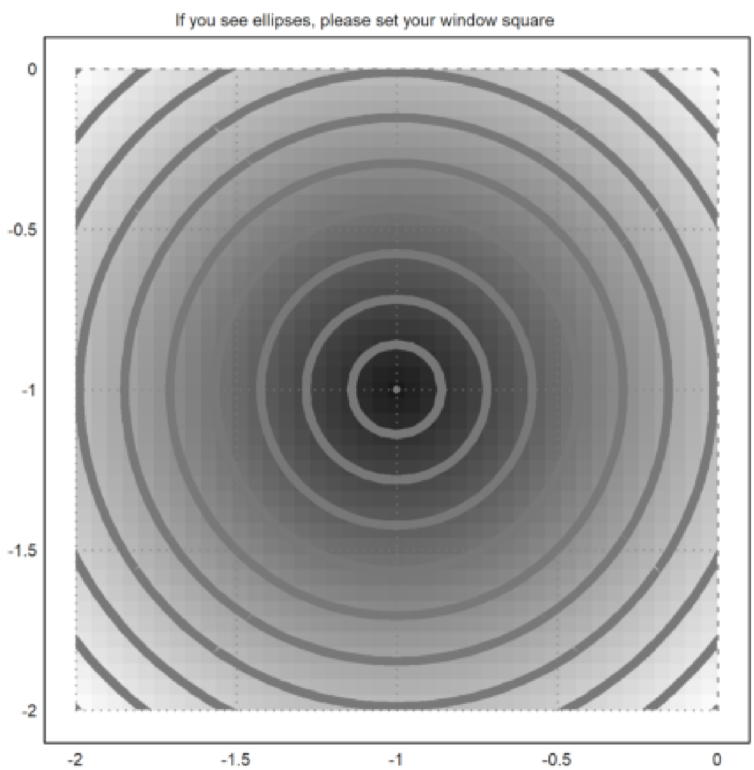
$$0$$

Contoh 6: Jarak Minimal pada Bidang

Ucapan awal

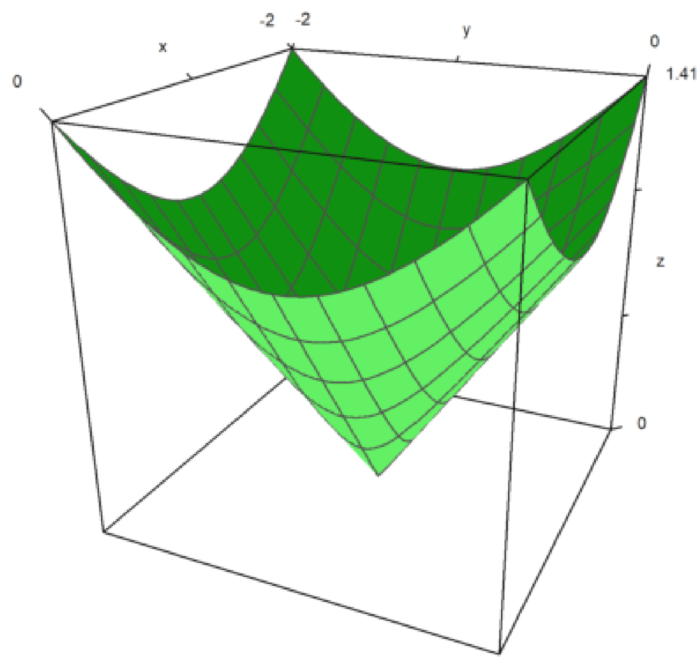
Fungsi yang, ke titik M di bidang, menetapkan jarak AM antara titik tetap A dan M, memiliki garis level yang agak sederhana: lingkaran berpusat di A.

```
>&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):
```

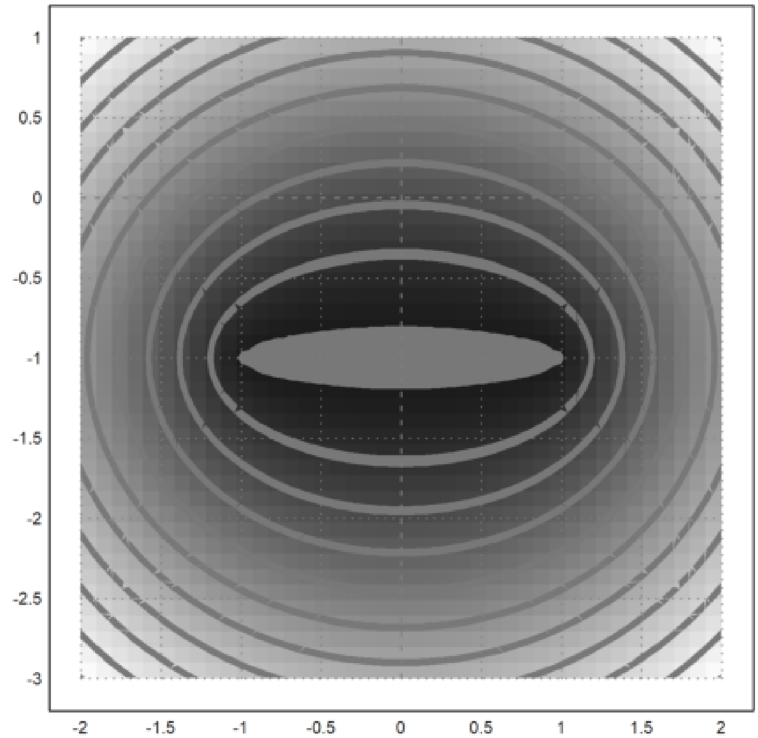


Tentu saja minimal 0 dicapai di A.

Two points

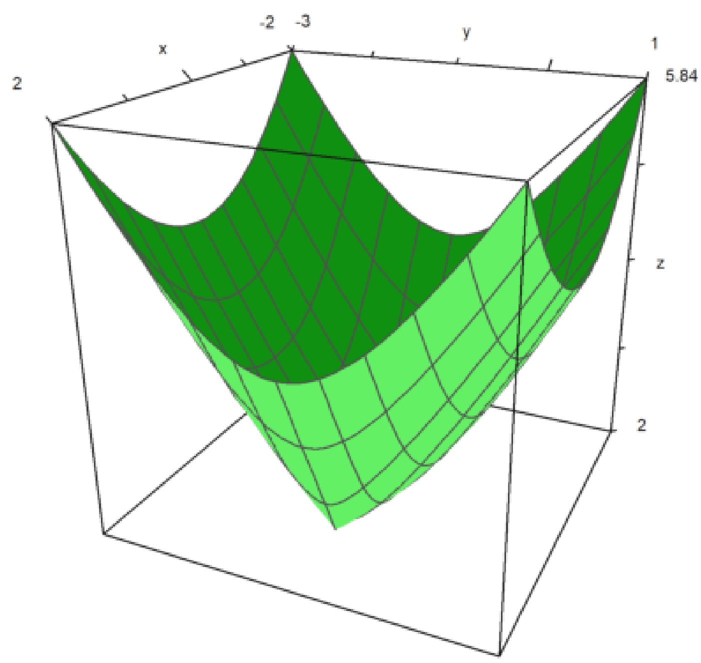
Sekarang kita melihat fungsi MA + MB dimana A dan B adalah dua titik (tetap). Ini adalah "fakta yang terkenal" bahwa kurva level adalah elips, titik fokusnya adalah A dan B; kecuali untuk minimum AB yang konstan pada segmen [AB]:

```
>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):
```



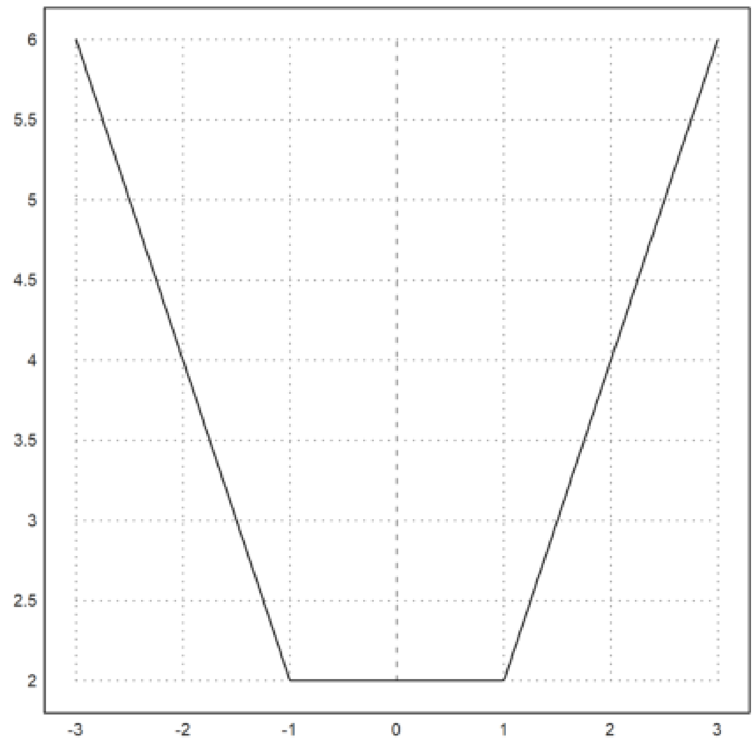
Grafiknya lebih menarik:

```
>plot3d("d2",xmin=-2,xmax=2,ymin=-3,ymax=1):
```



Batasan ke baris (AB) lebih terkenal:

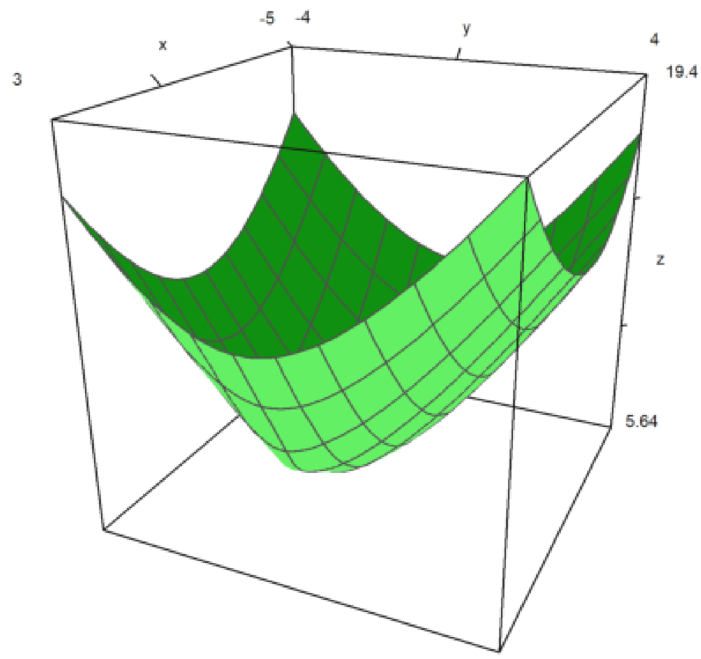
```
>plot2d("abs(x+1)+abs(x-1)",xmin=-3,xmax=3):
```



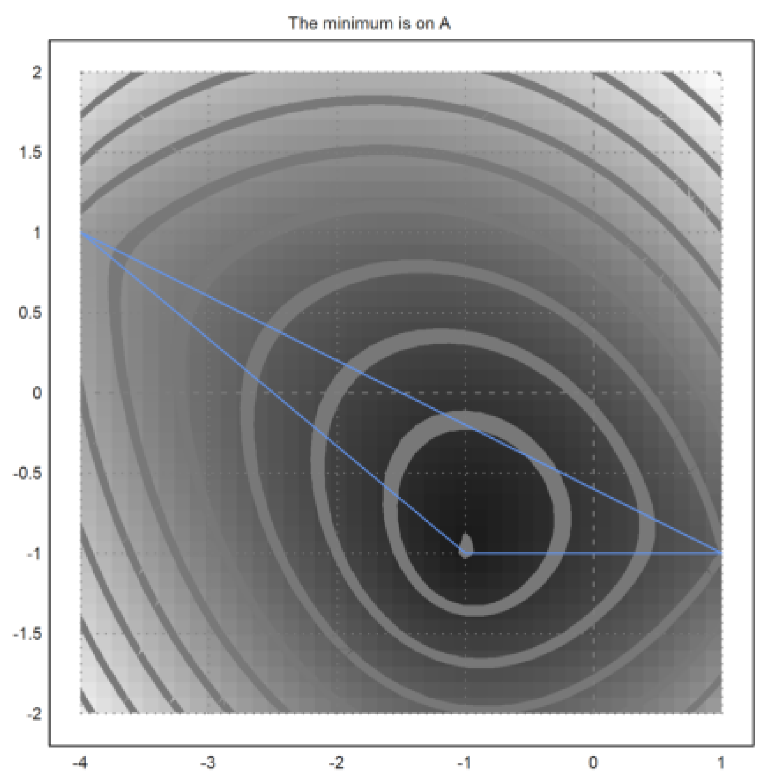
Three points

Sekarang hal-hal menjadi kurang sederhana: Sedikit kurang diketahui bahwa MA + MB + MC mencapai minimumnya pada satu titik bidang tetapi untuk menentukannya kurang sederhana:
1) Jika salah satu sudut segitiga ABC lebih dari 120 ? (katakanlah dalam A), maka minimum tercapai pada titik ini (katakanlah 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;
```

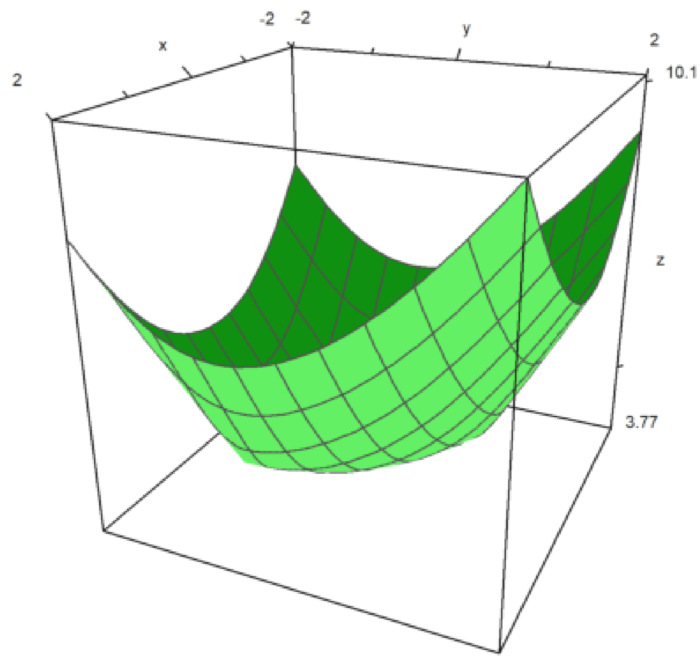



```
>fcontour("d3",xmin=-4,xmax=1,ymin=-2,ymax=2,hue=1,title="The minimum is on A");
>P=(A_B_C_A)'; plot2d(P[1],P[2],add=1,color=12);
>insimg;
```

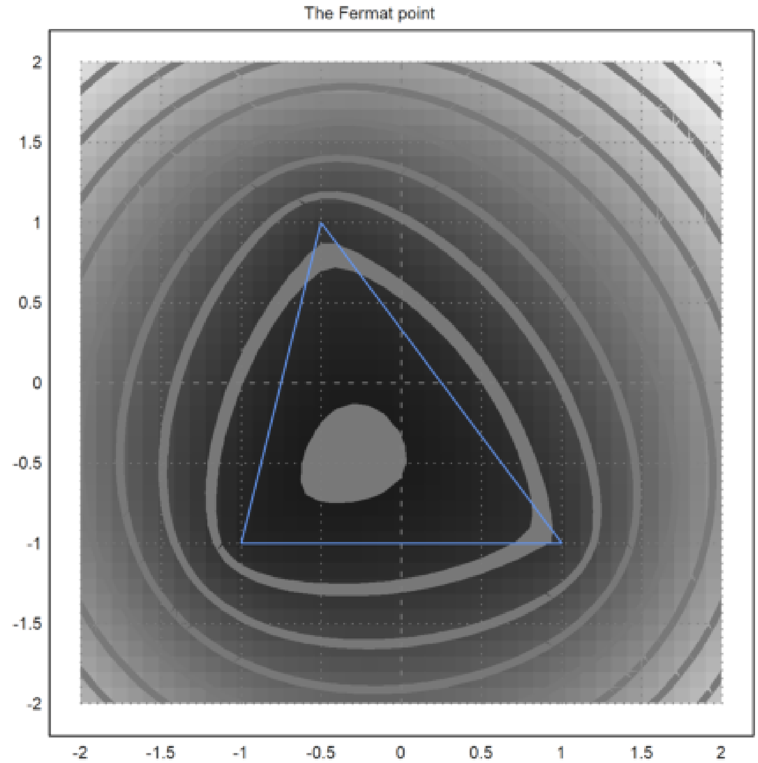


2) Tetapi jika semua sudut segitiga ABC kurang dari 120 ?, minimum berada pada titik F di bagian dalam segitiga, yang merupakan satu-satunya titik yang melihat sisi ABC dengan sudut yang sama (lalu masing-masing 120) :

```
>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;
```

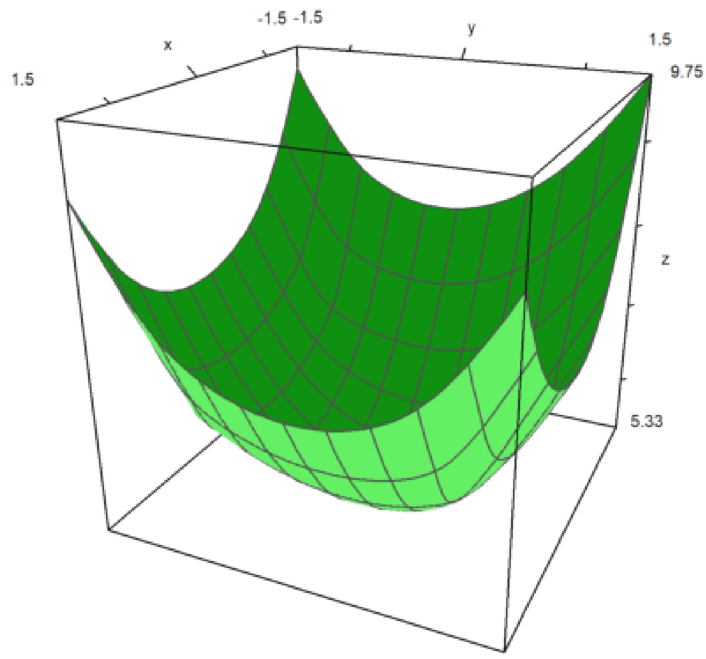


Merupakan kegiatan yang menarik untuk mewujudkan gambar di atas dengan perangkat lunak geometri; sebagai contoh, saya tahu soft tertulis di Java yang memiliki instruksi "garis kontur" ...
Semua ini di atas telah ditemukan oleh seorang hakim Prancis bernama Pierre de Fermat; dia menulis surat kepada para penggila lainnya seperti pendeta Marin Mersenne dan Blaise Pascal yang bekerja di bagian pajak penghasilan. Jadi titik unik F sehingga FA + FB + FC minimal disebut titik Fermat segitiga. Tetapi tampaknya beberapa tahun sebelumnya, Torriccelli Italia telah menemukan titik ini sebelum Fermat melakukannya! Pokoknya tradisinya adalah mencatat poin ini ...

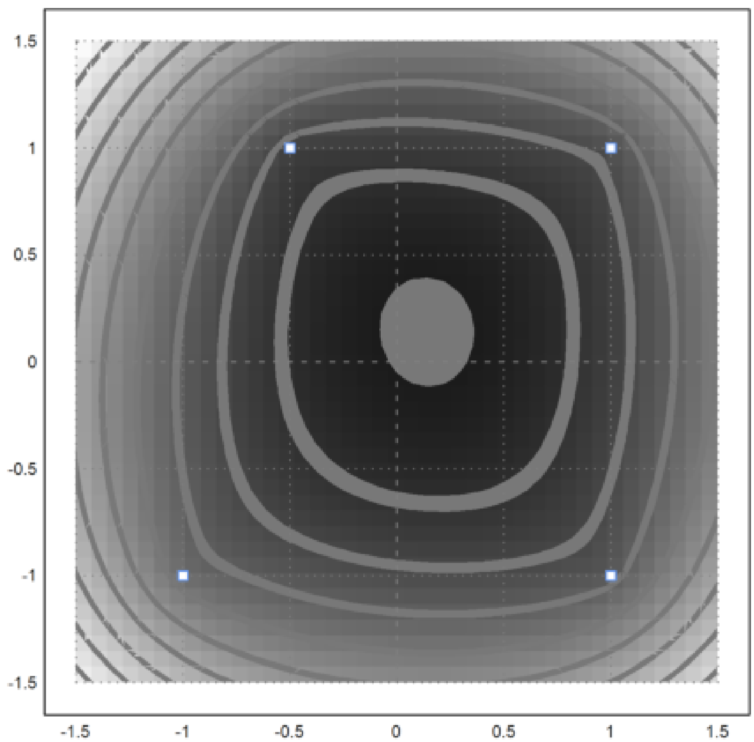
Four points

Langkah selanjutnya adalah menambahkan titik D ke-4 dan mencoba meminimalkan MA + MB + MC + MD; katakanlah 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 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;
```



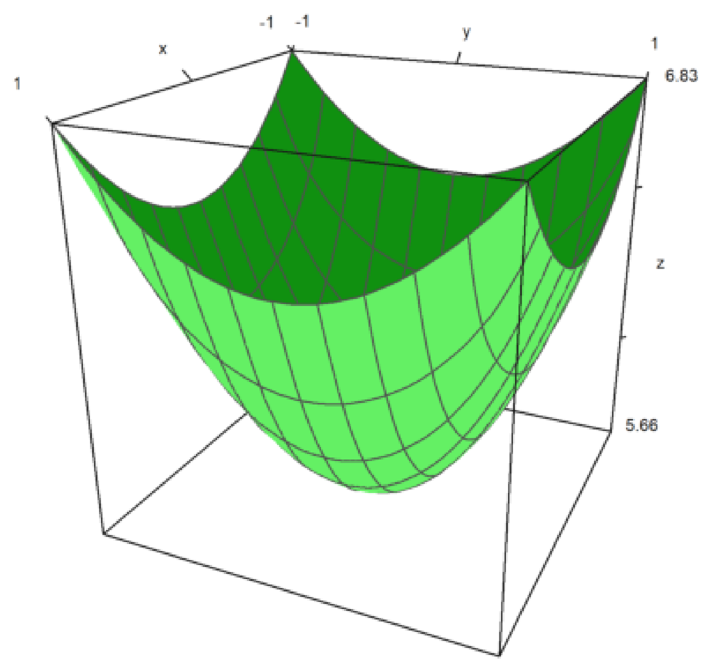
Masih ada minimum dan tidak ada yang dicapai pada simpul A, B, C atau D:

```
>function f(x):=d4(x[1],x[2])
>neldermin("f",[0.2,0.2])
```

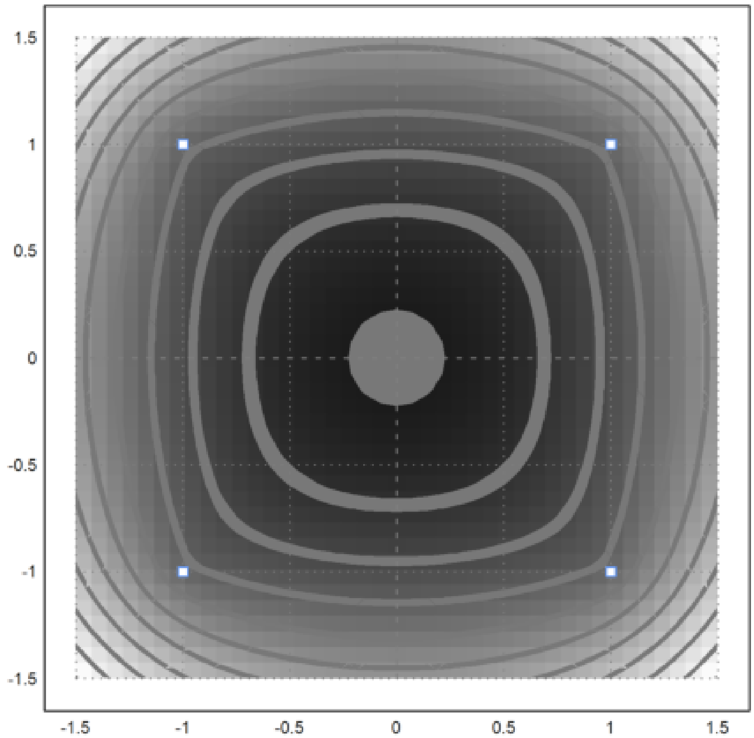
```
[0.142858,  0.142857]
```

Tampaknya dalam kasus ini, koordinat titik optimal rasional atau mendekati rasional ...
Sekarang ABCD adalah bujur sangkar, kami berharap bahwa titik optimal adalah 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 memiliki Povray diinstal, dan pvengine.exe di jalur program.
Pertama kami menghitung 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 geometry.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])
```

$$[- a, - 1, 0]$$

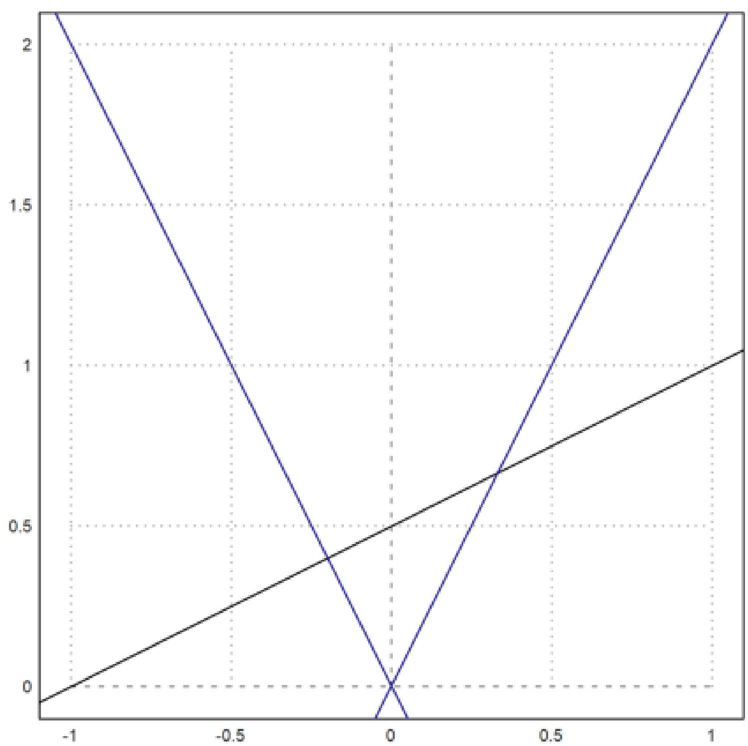
Lalu baris ketiga.

```
>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
```

$$\sqrt{\left(\frac{a^2\,u}{a^2+1}-u\right)^2+\frac{a^2\,u^2}{\left(a^2+1\right)^2}}$$

Hitung jarak ke g.

```
>d &= distance(P,projectToLine(P,g)); $d
```

$$\sqrt{\left(\frac{u+2}{5}-u\right)^2+\frac{\left(2\,u-1\right)^2}{25}}$$

Dan temukan pusat kedua lingkaran, di mana jaraknya sama.

```
>sol &= solve(d1^2=d^2,u); $sol
```

$$\left[u=\frac{-\sqrt{5}\,\sqrt{a^2+1}+2\,a^2+2}{4\,a^2-1} , u=\frac{\sqrt{5}\,\sqrt{a^2+1}+2\,a^2+2}{4\,a^2-1} \right]$$

Ada dua solusi.
Kami mengevaluasi solusi simbolis, dan menemukan kedua pusat, dan kedua jarak.

```
>u := sol()
```

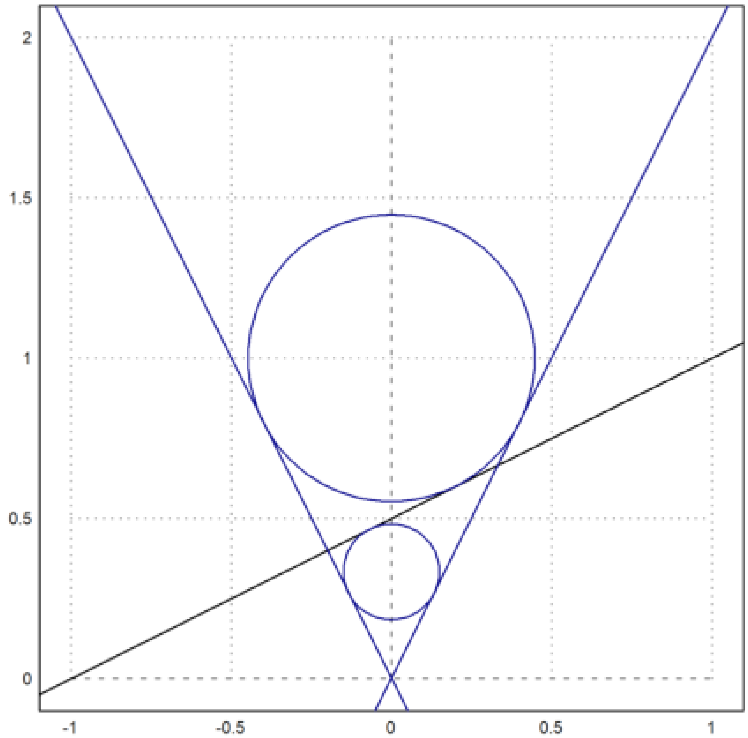
```
[0.333333,  1]
```

```
>dd := d()
```

```
[0.149071,  0.447214]
```

Plot lingkaran ke dalam gambar.

```
>color(blue);
>plotCircle(circleWithCenter([0,u[1]],dd[1]),"");
>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 kami menulis dua bidang ke file Povray.

```
>writeln(povsphere([0,0,u[1]],dd[1],povlook(red)));
>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 pesawat 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 kami 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]]);
>writeln(povpoint(P1,povlook(yellow)));
>P2=projectToLine([0,u[2]],g1()); P2=turnz([P2[1],0,P2[2]]);
>writeln(povpoint(P2,povlook(yellow)));
```

Kemudian kami menghasilkan dua titik di mana bola menyentuh bidang. Ini adalah fokus elips.

```
>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)));
```

Selanjutnya kita menghitung perpotongan P1P2 dengan bidang.

```
>t1=scalp(vp,P1)-dp; t2=scalp(vp,P2)-dp; P5=P1+t1/(t1-t2)*(P2-P1);
>writeln(povpoint(P5,povlook(yellow)));
```

Kami menghubungkan titik dengan segmen garis.

```
>writeln(povsegment(P1,P2,povlook(yellow)));
>writeln(povsegment(P5,P3,povlook(yellow)));
>writeln(povsegment(P5,P4,povlook(yellow)));
```

Sekarang kami membuat 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);
>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)));
```

Mulai program Povray.

```
>povend();
```

```
Function povend not found.
Try list ... to find functions!
Error in:
povend(); ...
        ^
```

Untuk mendapatkan Anaglyph ini, kita perlu memasukkan semuanya ke dalam fungsi scene. Fungsi ini akan digunakan dua kali nanti.

```
>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 / cyan untuk mengapresiasi efek berikut.

```
>povanaglyph("scene",zoom=11,center=[0,0,0.5],height=10°,angle=140°);
```

```
Function povanaglyph not found.
Try list ... to find functions!
Error in:
... ",zoom=11,center=[0,0,0.5],height=10°,angle=140°); ...
                                                     ^
```

Contoh 8: Geometri Bumi

Di notebook ini, kami ingin melakukan beberapa komputasi bola. Fungsi-fungsi tersebut terdapat dalam file "spherical.e" di folder contoh. Kita perlu memuat file itu dulu.

```
>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 bola).

```
>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 bola ke bola lainnya pada bola ideal. Vektor ini adalah [heading, distance] dalam radian. Untuk menghitung jarak di bumi, kita mengalikan dengan jari-jari bumi pada garis lintang 7°.

```
>br=svector(FMIPA,Solo); degprint(br[1]), br[2]*rearth(7°)->km // perkiraan jarak FMIPA-So
```

```
65°20'26.60''
53.8945384608
```

Ini adalah perkiraan yang bagus. Rutinitas berikut menggunakan perkiraan yang lebih baik. Pada jarak yang begitu dekat hasilnya hampir sama.

```
>esdist(FMIPA,Semarang)->km //perkiraan jarak FMIPA-Semarang
```

```
88.0114026318
```

Ada fungsi untuk heading, dengan mempertimbangkan bentuk bumi yang elips. Sekali lagi, kami mencetak dengan cara yang canggih.

```
>sdegprint(esdir(FMIPA,Solo))
```

```
     65.34°
```

```
1Sudut segitiga melebihi 80° pada bola.
```

```
>asum=sangle(Solo,FMIPA,Semarang)+sangle(FMIPA,Solo,Semarang)+sangle(FMIPA,Semarang,Solo);
```

```
180°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°)^2->" km^2";  // perkiraan luas segitiga FMIPA-Solo-Semarang
```

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 km^2
```

Kami juga dapat menambahkan vektor ke posisi. Vektor berisi heading dan jarak, keduanya dalam radian. Untuk mendapatkan vektor, kami menggunakan svector. Untuk menambahkan vektor ke posisi, kami menggunakan saddvector.

```
>v=svector(FMIPA,Solo); sposprint(saddvector(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(esadd(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 429,66 km. Kami mendapatkan perkiraan yang bagus.

```
>esdist(Tugu,Monas)->km // perkiraan jarak Tugu Jogja - Monas Jakarta
```

```
431.565659488
```

Judulnya sama dengan yang dihitung di Google Earth.

```
>degprint(esdir(Tugu,Monas))
```

```
294°17'2.85''
```

Namun kita tidak lagi mendapatkan posisi sasaran yang tepat, jika kita menambahkan heading dan jarak ke posisi semula. Hal ini terjadi karena kita tidak menghitung fungsi invers secara tepat, namun melakukan perkiraan jari-jari bumi di sepanjang lintasan.

```
>sposprint(esadd(Tugu,esdir(Tugu,Monas),esdist(Tugu,Monas)))
```

```
 S 6°10.500' E 106°48.717'
```

The error is not large, however.

```
>sposprint(Monas),
```

```
 S 6°10.500' E 106°48.717'
```

Tentunya kita tidak bisa berlayar dengan tujuan yang sama dari satu tujuan ke tujuan lainnya, jika kita ingin mengambil jalur terpendek. Bayangkan, Anda terbang NE mulai dari titik mana pun di bumi. Kemudian Anda akan berputar ke kutub utara. Lingkaran besar tidak mengikuti arah yang konstan!
Perhitungan berikut menunjukkan bahwa kami jauh dari tujuan yang benar, jika kami menggunakan tajuk yang sama selama perjalanan kami.

```
>dist=esdist(Tugu,Monas); hd=esdir(Tugu,Monas);
```

Sekarang kita tambahkan 10 kali sepersepuluh jaraknya, menggunakan heading ke Monas, kita sampai di Tugu.

```
>p=Tugu; loop 1 to 10; p=esadd(p,hd,dist/10); end;
```

Hasilnya masih jauh.

```
>sposprint(p), skmprint(esdist(p,Monas))
```

```
 S 6°11.250' E 106°48.372'
      1.529km
```

Sebagai contoh lain, mari kita ambil dua titik di bumi pada ketinggian yang sama.

```
>P1=[30°,10°]; P2=[30°,50°];
```

Jalur terpendek dari P1 ke P2 bukanlah lingkaran dengan garis lintang 30 ?, tetapi jalur yang lebih pendek mulai 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 tujuan kita di sepanjang jalan. Untuk tujuan kasar, kami menyesuaikannya pada 1/10 dari jarak total.

```
>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 tajuk yang sama terlalu lama.

```
>skmprint(esdist(p,P2))
```

```
0.203km
```

Kami mendapatkan perkiraan yang baik, jika kami menyesuaikan heading 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'
```

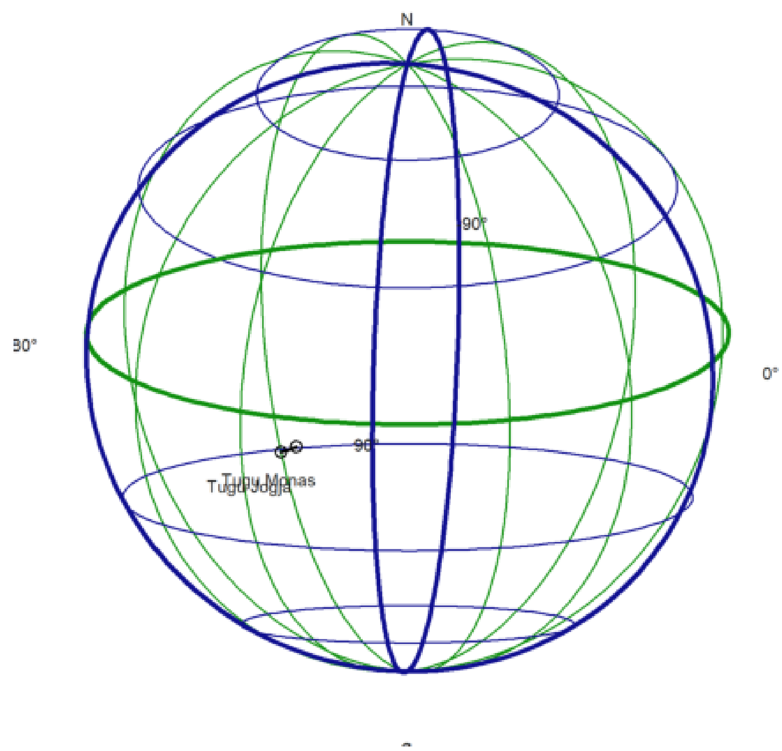
Kami menulis sebuah fungsi, yang menggambarkan bumi, dua posisi, dan posisi di antaranya.

```
>function testplot ...
```

```
useglobal;
plotearth;
plotpos(Tugu,"Tugu Jogja"); plotpos(Monas,"Tugu Monas");
plotposline(v);
endfunction
```

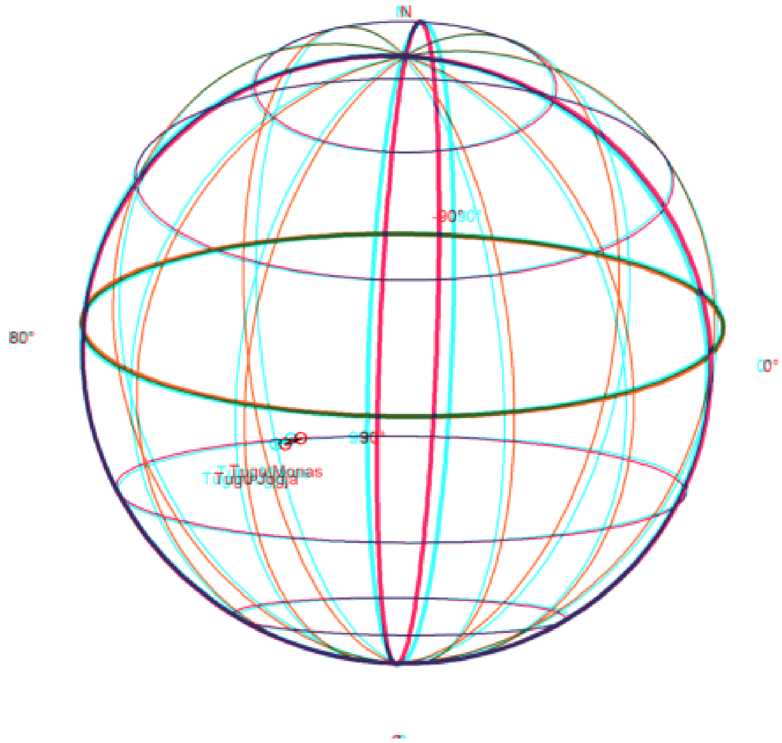
Now plot everything.

```
>plot3d("testplot",angle=25, height=6,>own,>user,zoom=4):
```



Atau gunakan plot3d untuk mendapatkan tampilan anaglyphnya. Ini terlihat sangat bagus dengan kacamata merah / cyan.

```
>plot3d("testplot",angle=25,height=6,distance=5,own=1,anaglyph=1,zoom=4):
```



Latihan Soal

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

```
>o &:= [0,0]; c=circleWithCenter(o,5);
```

```
Function circleWithCenter not found.
Try list ... to find functions!
Error in:
o &:= [0,0]; c=circleWithCenter(o,5); ...
                                     ^
```

```
>color(1); setPlotRange(5); plotPoint(o); plotCircle(c);
```

```
Function setPlotRange not found.
Try list ... to find functions!
Error in:
```

```
color(1); setPlotRange(5); plotPoint(o); plotCircle(c); ...
                               ^
```

```
>A=[-5,0]; plotPoint(A,"A");
```

```
Function plotPoint not found.
Try list ... to find functions!
Error in:
A=[-5,0]; plotPoint(A,"A"); ...
                            ^
```

```
>B=[5,0]; plotPoint(B,"B");
```

```
Function plotPoint not found.
Try list ... to find functions!
Error in:
B=[5,0]; plotPoint(B,"B"); ...
                           ^
```

```
>plotSegment(A,B,""):
```

```
Function plotSegment not found.
Try list ... to find functions!
Error in:
plotSegment(A,B,""): ...
                     ^
```

```
>c1=circleWithCenter(A,distance(A,o));
```

```
Function distance not found.
Try list ... to find functions!
Error in:
c1=circleWithCenter(A,distance(A,o)); ...
                                      ^
```

```
>c2=circleWithCenter(B,distance(B,o));
```

```
Function distance not found.
Try list ... to find functions!
Error in:
c2=circleWithCenter(B,distance(B,o)); ...
                                      ^
```

```
>k=circleCircleIntersections(c1,c);
```

```
Variable or function c1 not found.
Error in:
k=circleCircleIntersections(c1,c); ...
                                 ^
```

```
>l=circleCircleIntersections(c,c2);
```

```
Variable or function c not found.
Error in:
l=circleCircleIntersections(c,c2); ...
                             ^
```

```
>m=circleCircleIntersections(c2,c);
```

```
Variable or function c2 not found.
Error in:
m=circleCircleIntersections(c2,c); ...
                              ^
```

```
>n=circleCircleIntersections(c,c1);
```

```
Variable or function c not found.
Error in:
n=circleCircleIntersections(c,c1); ...
                             ^
```

```
>r=lineThrough(k,m); s=lineThrough(l,n);
```

```
Variable or function k not found.
Error in:
r=lineThrough(k,m); s=lineThrough(l,n); ...
               ^
```

```
>setPlotRange(8); plotPoint(o); plotCircle(c); plotPoint(A,"A"); plotPoint(B,"B"); plotSeg
```

```
Function setPlotRange not found.
Try list ... to find functions!
Error in:
setPlotRange(8); plotPoint(o); plotCircle(c); plotPoint(A,"A") ...
               ^
```

```
>color(4); plotCircle(c1); plotCircle(c2); plotPoint(k); plotPoint(l); plotPoint(m); plotP
```

```
Variable or function c1 not found.
Error in:
color(4); plotCircle(c1); plotCircle(c2); plotPoint(k); plotPo ...
                       ^
```

```
>color(5); plotLine(r); plotLine(s):
```

```
Variable or function r not found.
Error in:
color(5); plotLine(r); plotLine(s): ...
                    ^
```

```
>color(6); plotSegment(A,k,""); plotSegment(A,n,""); plotSegment(k,l,""); plotSegment(l,B,
```

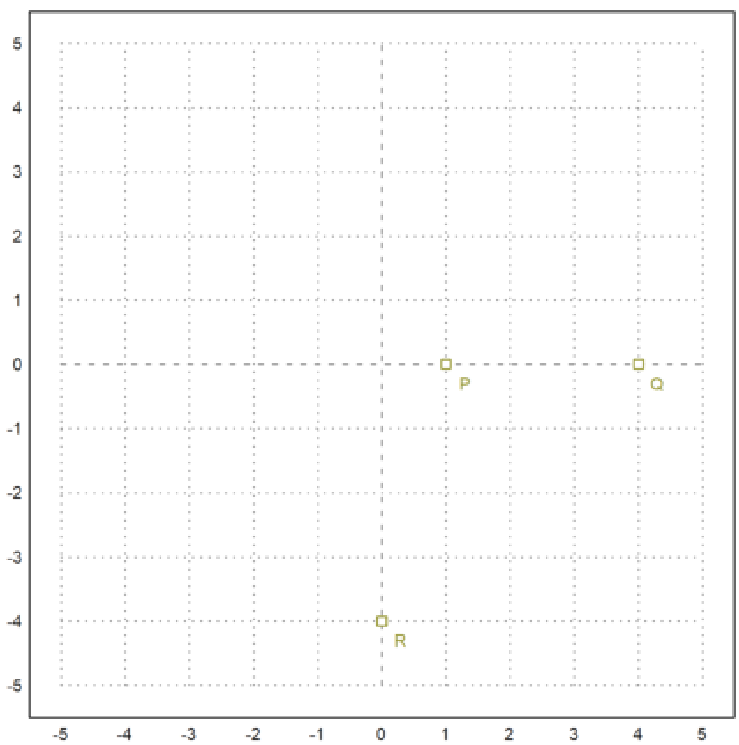
```
Variable or function k not found.
Error in:
color(6); plotSegment(A,k,""); plotSegment(A,n,""); plotSegmen ...
                         ^
```

---

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=[1,0]; Q=[4,0]; R=[0,-4];
>plotPoint(P,"P"); plotPoint(Q,"Q"); plotPoint(R,"R"):
```



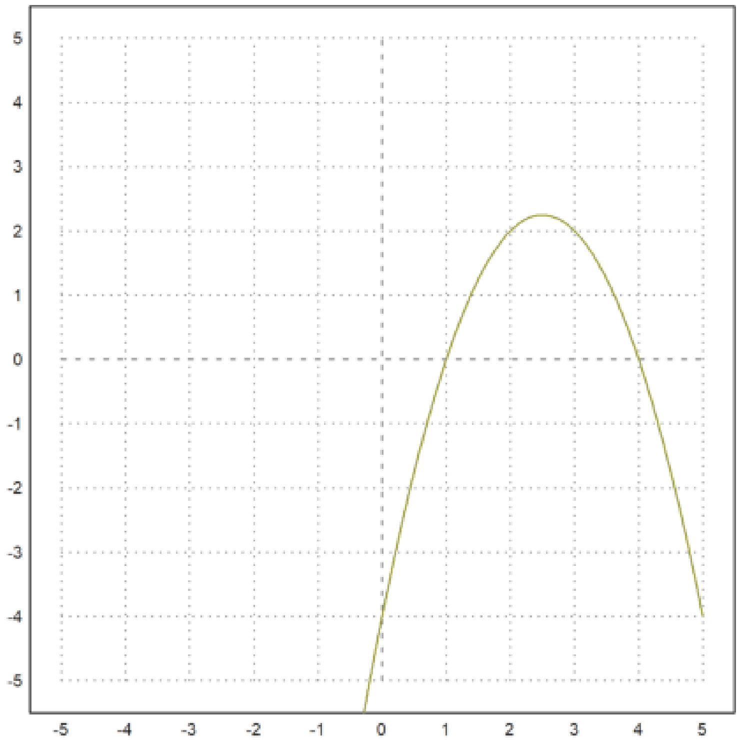
```
>sol &= solve([a+b=-c,16*a+4*b=-c,c=-4],[a,b,c])
```

```
[[a = - 1, b = 5, c = - 4]]
```

```
>function y&=-x^2+5*x-4
```

```
     2
- x   + 5 x - 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-sisintya 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
```

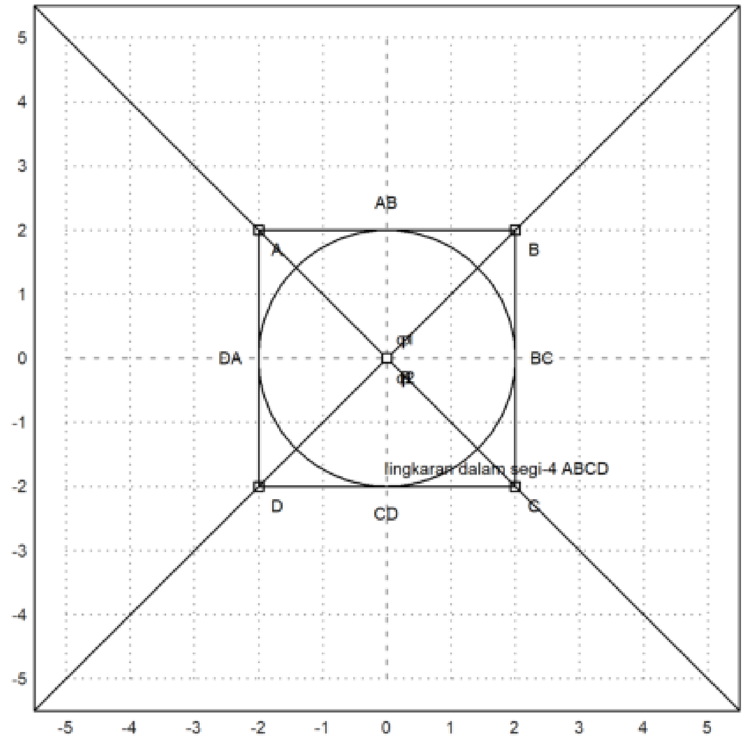
singgung apabila hasil kali panjang sisi-sisi yang berhadapan sama.

Penyelesaian

```
>setPlotRange(-5,5,-5,5);
>A=[-2,2]; plotPoint(A,"A");
>B=[2,2]; plotPoint(B,"B");
>C=[2,-2]; plotPoint(C,"C");
>D=[-2,-2]; plotPoint(D,"D");
>plotSegment(A,B);
>plotSegment(B,C);
>plotSegment(C,D);
>plotSegment(D,A);
>plotSegment(A,C,"q1");
>plotSegment(B,D,"q2");
>q1=lineThrough(A,C);
>q2=lineThrough(B,D);
>p=lineIntersection(q1,q2);
>plotLine(q1); plotLine(q2);
>plotPoint(p, "P");
>r=norm(p-projectToLine(p,lineThrough(A,B)))
```

```
2
```

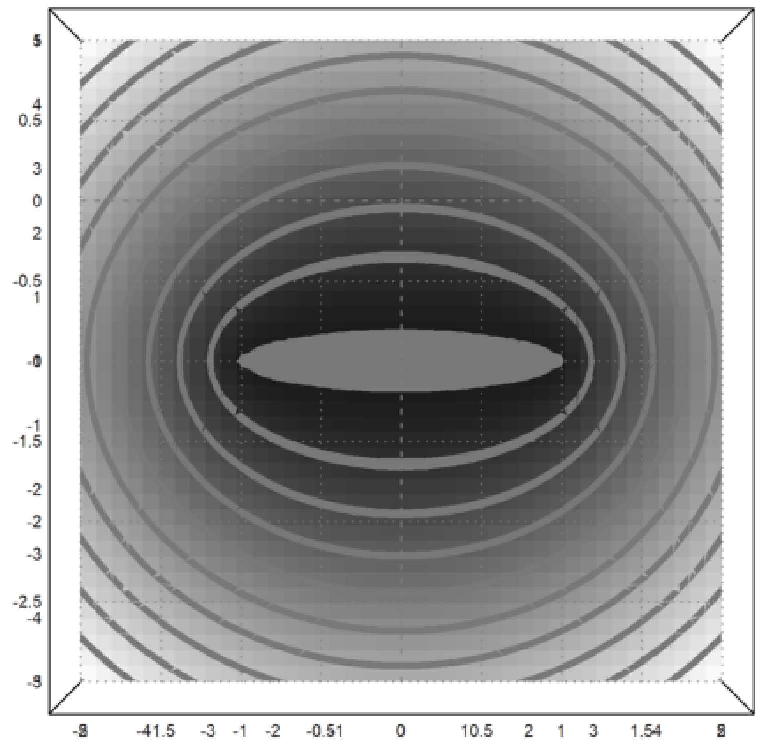
```
>plotCircle(circleWithCenter(p,r),"lingkaran dalam segi-4 ABCD"):
```



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).

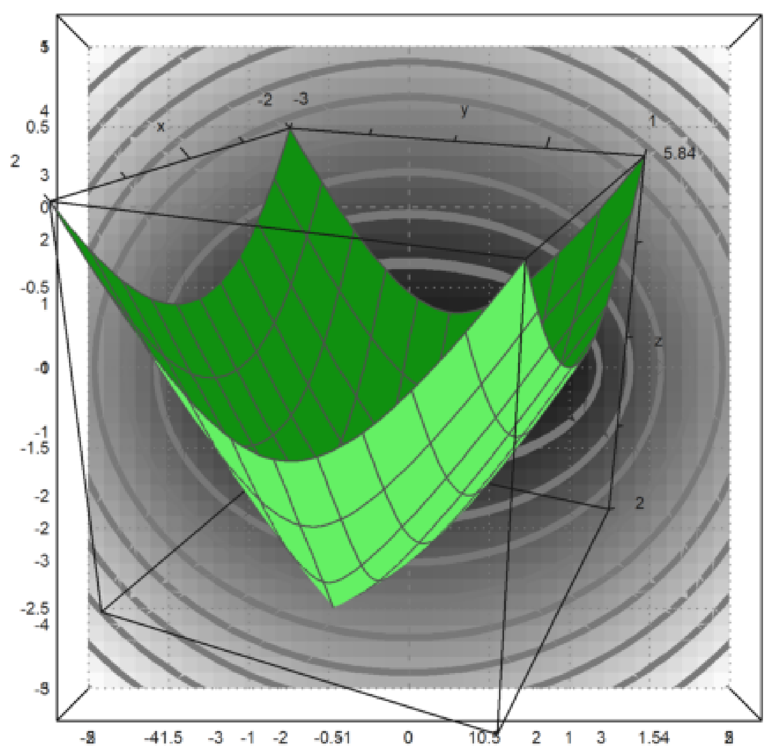
Penyelesaian

```
>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):
```



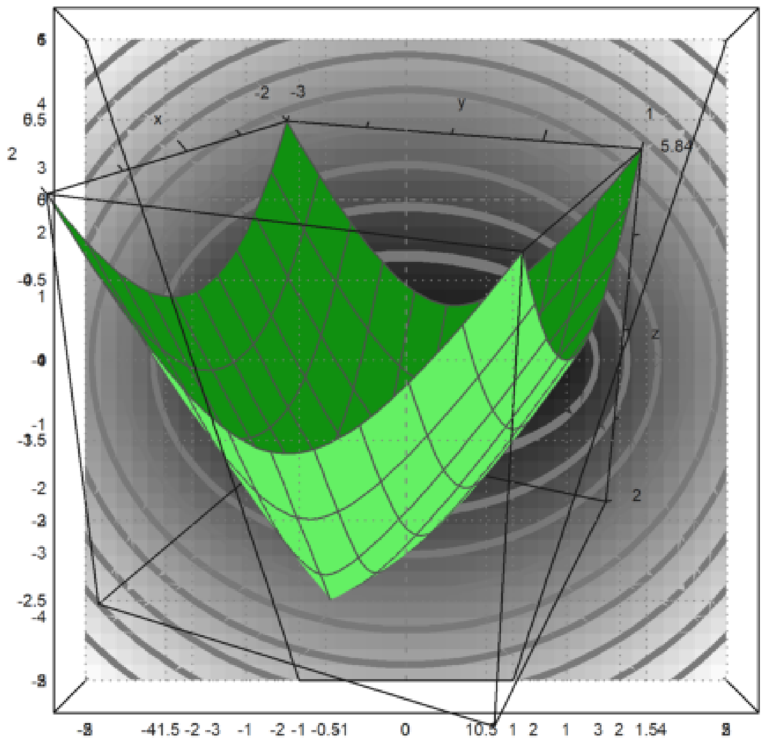
Grafik yang lebih menarik

```
>plot3d("d2",xmin=-2,xmax=2,ymin=-3,ymax=1):
```



Batasan ke garis PQ

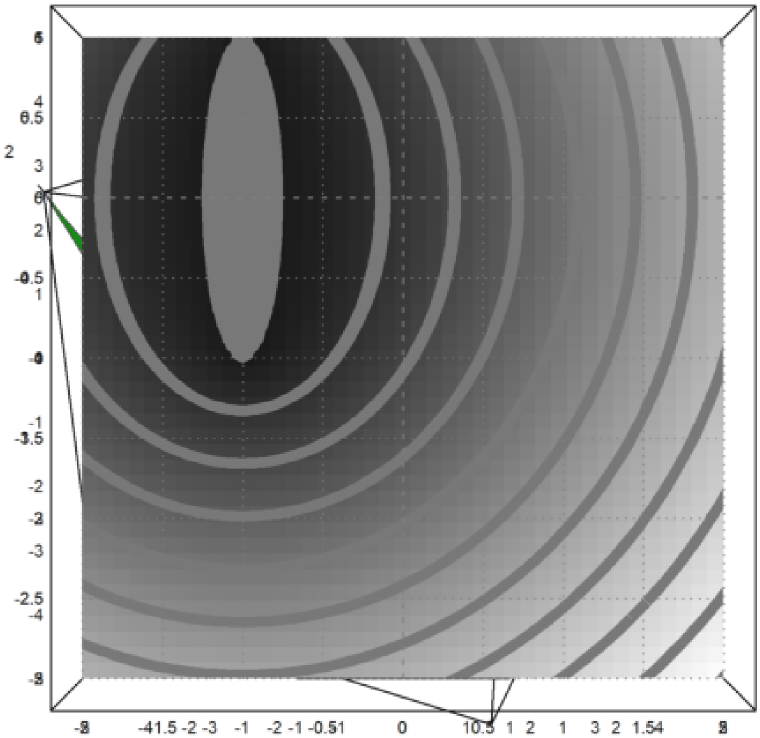
```
>plot2d("abs(x+1)+abs(x-1)",xmin=-3,xmax=3):
```



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).

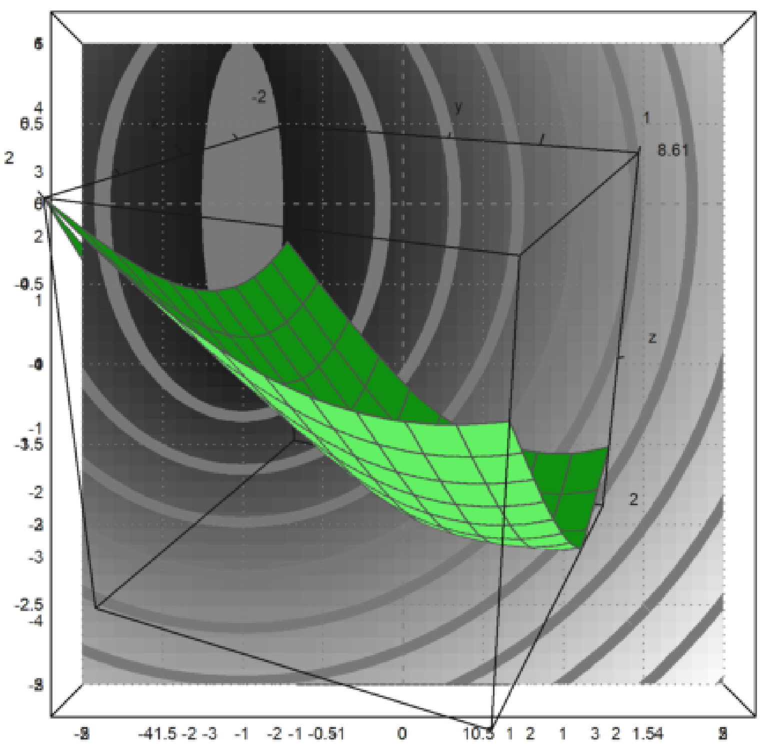
Penyelesaian

```
>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):
```

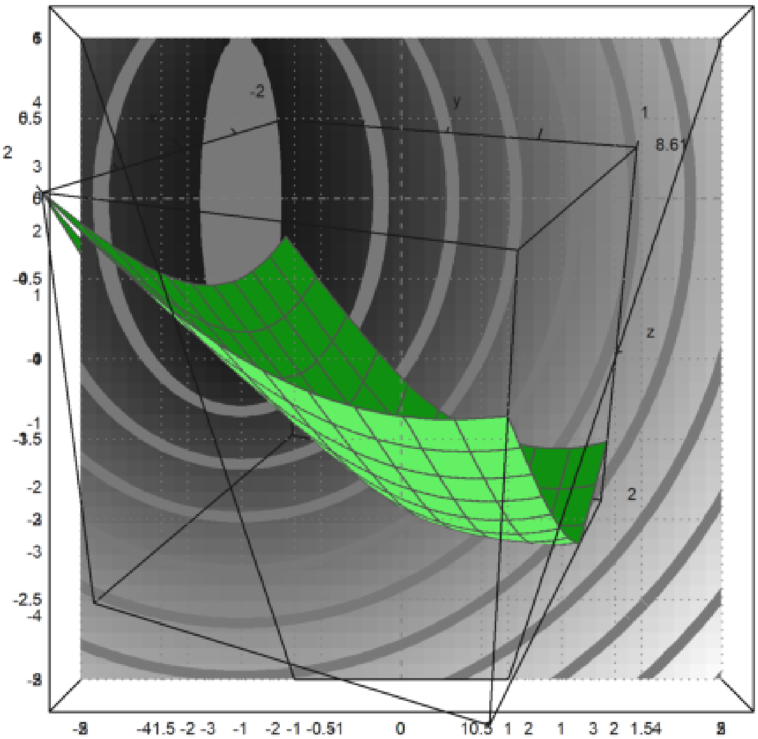


Grafik yang lebih menarik

```
>plot3d("d2",xmin=-2,xmax=2,ymin=-3,ymax=1):
```



```
>plot2d("abs(x+1)+abs(x-1)",xmin=-3,xmax=3):
```



BAB 6

KB PEKAN 10; MENGGUNAKAN EMT UNTUK STATISTIKA

[a4paper,10pt]article eumat

EMT Untuk Visualisasi Dan Komputasi Statistika

Kelompok 6
Nama : Ardan Andhirta (22305141045)

```
Nafisatul Iqima (22305144037)
Fransisca Renita Pejoresa (22305144012)
Oktavia Kusuma Wardhani (22305144013)
Dida Arkadia Ayu Jawata (22305144005)
Chintya Wijayanti (22305144029)
Bintang Mahija Aryacetta (22305144003)
Adib Brian Syuhada (22305144014)
```

Kelas : Matematika E 2022

Sub Topik 1: Menyimpan Data Dalam Bentuk Matriks

Array

Array adalah kumpulan-kumpulan variabel yang menyimpan data dengan tipe yang sama atau data-data yang tersusun secara linear dimana di dalamnya terdapat elemen dengan tipe yang sama.
Vektor digunakan untuk menggambarkan array angka satu dimensi. Vektor memiliki panjang, yang merupakan jumlah elemen dalam array.
Sedangkan matriks digunakan dalam mendeskripsikan susunan bilangan dua
dimensi yang disusun dalam baris dan kolom. matriks memiliki ukuran, yaitu jumlah baris dan kolom.
Hubungan antara array dan matriks adalah bahwa matriks adalah bentuk khusus dari array. Array dapat memiliki lebih dari dua dimensi, tetapi matriks selalu memiliki dua dimensi. Dalam pemrograman, array dan matriks sering digunakan untuk menyimpan data dalam jumlah besar dan memudahkan pengaksesan data tersebut.
Mari kita bahas beberapa hal terkait vektor terlebih dahulu

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

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

```
>w=intrandom(10,12)
```

```
[11,  4,  9,  3,  6,  4,  11,  3,  6,  2]
```

Untuk mengurutkan angka acak

```
>sort(v)
```

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

Selanjutnya mengurutkan angka acak dengan menyederhanakan angka yang sama

```
>unique(v)
```

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

Menemukan banyaknya setiap elemen dengan bantuan interval

```
>s=intrandom(10,20)
```

```
[12,  9,  15,  10,  7,  11,  11,  4,  6,  18]
```

```
>x=[5,10,15,20]
```

```
[5,  10,  15,  20]
```

```
>find(x,s)
```

```
[2,  1,  3,  2,  1,  2,  2,  0,  1,  3]
```

Berikutnya adalah cara mencari indeks dari sebuah vektor dengan contoh vekUntuk indeks pada EMT berbeda dengan indeks pada Phyton yang kita pelajari sebelumnya di Algoritma dan pemrograman. Perbedaannya juka sebelumnya untk menentukan indeks akan dimulalai dari nol namun di mmenentukan indeks di EMT akan dimulai dari angka satu, berikut penjelasannya

```
>indexof(w,1:10)
```

```
[0,  10,  4,  2,  0,  5,  0,  0,  3,  0]
```

```
>x= sort(intrandom(10,12))
```

```
[1,  1,  3,  6,  7,  8,  8,  12,  12,  12]
```

```
>indexofsorted(x,1:15)
```

```
[2,  0,  3,  0,  0,  4,  5,  7,  0,  0,  0,  10,  0,  0,  0]
```

```
>z=intrandom(1000,10); multofsorted(sort(z),1:10), sum(%)
```

```
[81,  96,  121,  101,  102,  91,  115,  98,  100,  95]
1000
```

Sampai disini pembahasan terkait dnegan vektor
Selanjutnya kita akan membahas beberapa hal terkait matriks terkait
Untuk Menyimpan Data dalam bentuk Matrik
Pertama, buat sebuah variabel yang akan menampung data matrik, misal X. Variabel ini bebas dengan syarat tidak sama dengan nama fungsi atau konstanta yang sudah ada dalam software.
Selanjutnya,kita akan membuat matrik berordo mxn yang berisi angka

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

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

```
>shortformat; A=random(3,4)
```

```
  0.50136   0.58172   0.02845   0.72032
  0.27668    0.1313   0.84982   0.77608
  0.32956   0.68574   0.42373   0.77217
```

```
>shortformat; A=intrandom(5,4,20)
```

```
        3        18         4        18
        7        19        12         8
        5        17        11        10
       15         1        20         7
       11        13         9         2
```

```
>shortformat; A=redim(1:15,4,4)
```

```
        1         2         3         4
        5         6         7         8
        9        10        11        12
       13        14        15         0
```

```
>(1:5)_2
```

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

```
>random(3,3)_random(2,2)
```

```
    0.74041    0.41322    0.25054
    0.15543    0.10655    0.98859
    0.48475   0.078167    0.57911
    0.46856    0.22056          0
    0.22837    0.47473          0
```

```
>for k=1 to prod(size(A)); A{k}=k; end; short A
```

```
          1          2          3          4
          5          6          7          8
          9         10         11         12
         13         14         15         16
```

```
>B=zeros(size(A))
```

```
          0          0          0          0
          0          0          0          0
          0          0          0          0
          0          0          0          0
```

```
>B=ones(size(A))
```

```
          1          1          1          1
          1          1          1          1
          1          1          1          1
          1          1          1          1
```

Berikutnya operasi penjumlahan dam pengurangan matriks

```
>shortformat; I=intrandom(3,4,10)
```

```
          5          5          7          5
          9          8          3          8
          5          5          1          1
```

```
>shortformat; J=intrandom(3,4,8)
```

```
          7          8          1          3
          3          7          6          7
          1          1          1          6
```

```
>C= I-J
```

```
         -2         -3          6          2
          6          1         -3          1
          4          4          0         -5
```

```
>C= I+J
```

```
        12        13        8        8
        12        15        9       15
         6         6        2        7
```

Dalam materi matriks yang pernah kita pelajari ada sebutan transpose, Invers dan juga determinan, jika menggunakan EMt sebagai berikut secera berurutan:

```
>T = transpose(I)
```

```
         5         9         5
         5         8         5
         7         3         1
         5         8         1
```

```
>T = I'
```

```
         5         9         5
         5         8         5
         7         3         1
         5         8         1
```

```
>K = J^(-1)
```

```
   0.14286     0.125         1   0.33333
   0.33333   0.14286   0.16667   0.14286
         1         1         1   0.16667
```

```
>shortformat; L=intrandom(3,3,7)
```

```
         7         4         4
         3         1         6
         3         7         6
```

```
>det(L)
```

```
  -180
```

Selanjutnya adalah cara ekstraksi baris dan kolom, atau sub-matriks,yang mirip dengan R sebagai berikut:

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

```
         4         4
         1         6
         7         6
```

```
>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
```

```
>function setmatrixvalue (M, i, j, v) ...
```

```
  loop 1 to max(length(i),length(j),length(v))
     M[i{#},j{#}] = v{#};
  end;
  endfunction
```

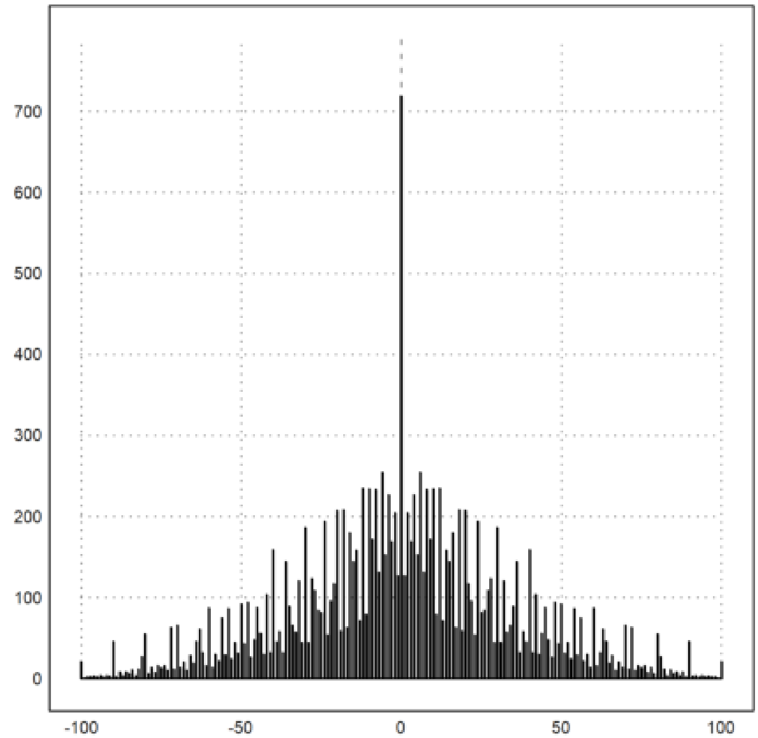
```
>setmatrixvalue(X,1:4,4:-1:1,0); X,
```

```
           1           2           3           0           5
           6           7           0           9          10
          11           0          13          14          15
           0          17          18          19          20
```

```
>(1:4)*(1:4)'
```

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

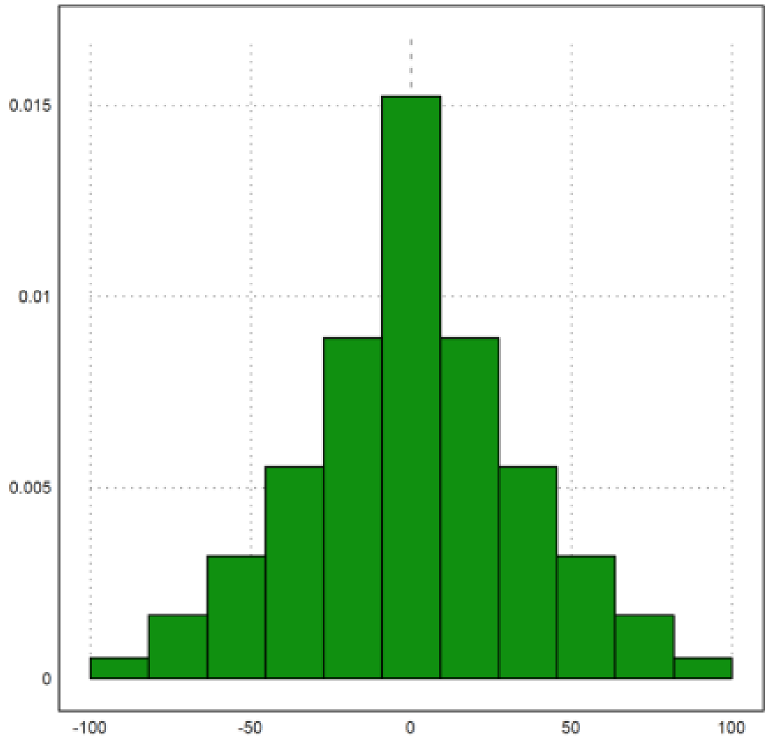
```
>a=0:10; b=a'; p=flatten(a*b); q=flatten(p-p'); ...
>u=sort(unique(q)); f=getmultiplicities(u,q); ...
>statplot(u,f,"h"):
```



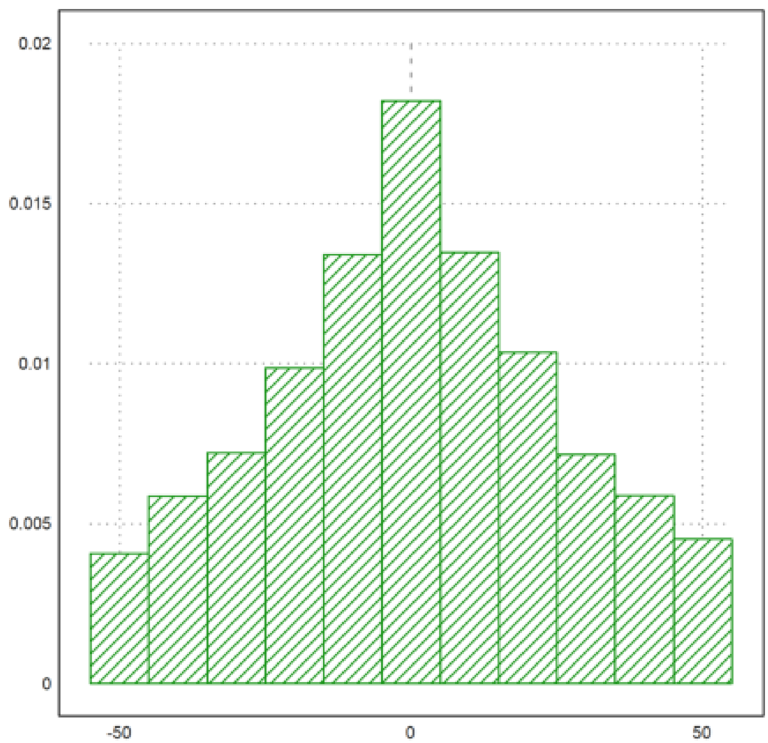

```
>getfrequencies(q,-50:10:50)
```

```
  [613,  814,  1088,  1404,  1904,  2389,  1431,  1109,  841,  680]
```

```
>plot2d(q,distribution=11):
```




```
>{x,y}=histo(q,v=-55:10:55); y=y/sum(y)/differences(x);
>plot2d(x,y,>bar,style="/"):
```



Sub Topik 2 : Menghasilkan Data Acak Menggunakan Fungsi Distribusi

CAKUPAN MATERI MELIPUTI DIANTARANYA:
-Definisi Bilangan Acak dan Data Acak
-Pengertian Distribusi Diskrit dan Konsep yang Terkait
-Metode Menentukan Distribusi Diskrit
1. Definisi Bilangan Acak dan Data Acak

```
Bilangan Acak adalah bilangan yang tidak dapat diprediksi
```

kemunculannya. Sehingga, tidak ada komputasi yang benar-benar menghasilkan deret bilangan acak secara sempurna.

```
Bilangan acak sendiri dapat dibangkitkan dengan pola tertentu
```

yang dinamakan dengan distribusi, dengan catatan mengikuti fungsi distribusi yang ditentukan.

```
Data acak merupakan hasil dari suatu percobaan acak. Sedangkan
```

percobaan acak adalah suatu proses yang dilakukan sedemikian rupa sehingga hasilnya tidak dapat ditentukan dengan pasti sebelum percobaan tersebut selesai dilakukan
contoh :

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

```
[2,  4,  6,  7,  3,  3,  2,  9,  10,  2]
```

---

2. Pengertian Distribusi Diskrit dan Konsep yang Terkait

```
Distribusi diskrit dalam statistika adalah distribusi data yang
```

memiliki nilai-nilai yang terpisah dan dapat dihitung. Contohnya adalah jumlah anak dalam sebuah keluarga, jumlah mata dadu yang muncul, atau jumlah pelanggan yang datang ke sebuah toko.

```
Distribusi diskrit merujuk pada distribusi
```

probabilitas yang melibatkan variabel acak diskrit. Variabel acak diskrit adalah variabel acak yang hanya dapat mengambil nilai-nilai terpisah, bukan nilai-nilai kontinu seperti pada variabel acak kontinu. Distribusi diskrit memberikan probabilitas masing-masing nilai yang mungkin dari variabel acak tersebut. Berikut adalah beberapa konsep kunci yang terkait dengan distribusi diskrit dalam
statistika:
1. Fungsi Probabilitas Diskrit (Probability Mass Function - PMF):
-Fungsi probabilitas diskrit, atau PMF, memberikan probabilitas bahwa variabel acak diskrit akan mengambil nilai tertentu.
PMF umumnya dilambangkan dengan P(X=x), di mana X adalah variabel acak dan x adalah nilai yang mungkin dari variabel tersebut.

2. Ruang Sampel (Sample Space):
-Ruang sampel adalah himpunan semua hasil mungkin dari suatu percobaan acak yang dapat diukur.
-Setiap elemen dalam ruang sampel merupakan hasil yang mungkin dari variabel acak.

3. Hukum Probabilitas untuk Distribusi Diskrit:
Probabilitas suatu kejadian adalah bilangan yang berada dalam rentang 0 hingga 1, atau 0 <= P(A)<=1 untuk setiap kejadian A.
Probabilitas total dari semua hasil dalam ruang sampel adalah 1, atau P (S)= 1, di mana S adalah ruang sampel.

4. Fungsi Distribusi Kumulatif (Cumulative Distribution Function - CDF):
-Fungsi distribusi kumulatif memberikan probabilitas bahwa variabel acak diskrit kurang dari atau sama dengan nilai tertentu.
-Notasi matematisnya sering kali disimbolkan sebagai F(x)-P(X<=x)

5. Harapan (Expectation) dan Varians:
-Harapan atau nilai rata-rata (E(X)) dari distribusi diskrit adalah jumlah tertimbang dari nilai-nilai mungkin berdasarkan probabilitas masing-masing nilai.
-Varians Var(X)) mengukur sejauh mana nilai-nilai distribusi tersebar dari nilai rata-ratanya.

```
>
```

---

3. Metode Menentukan Distribusi Diskrit

```
Untuk menentukan distribusi diskrit sendiri, dapat menggunakan
```

metode berikut. Pertama kita mengatur fungsi distribusi, fungsi distribusi adalah fungsi yang menggambarkan kemungkinan suatu variabel acak untuk memiliki nilai tertentu atau dalam rentang waktu tertentu.
Langkah mengatur fungsi distribusi:
-Menentukan jenis var acak yg akan diteliti, apakah diskrit atau kontinu
-Menentukan parameter-parameter yang berkaitan dengan fungsi distribusi, spt probabilitas
-Menentukan bentuk fungsi distribusi yg sesuai dg variabel acak dan parameter yg sudah ditentukan

```
>wd = 0|((1:6)+[-0.01,0.01,0,0,0,0])/5
```

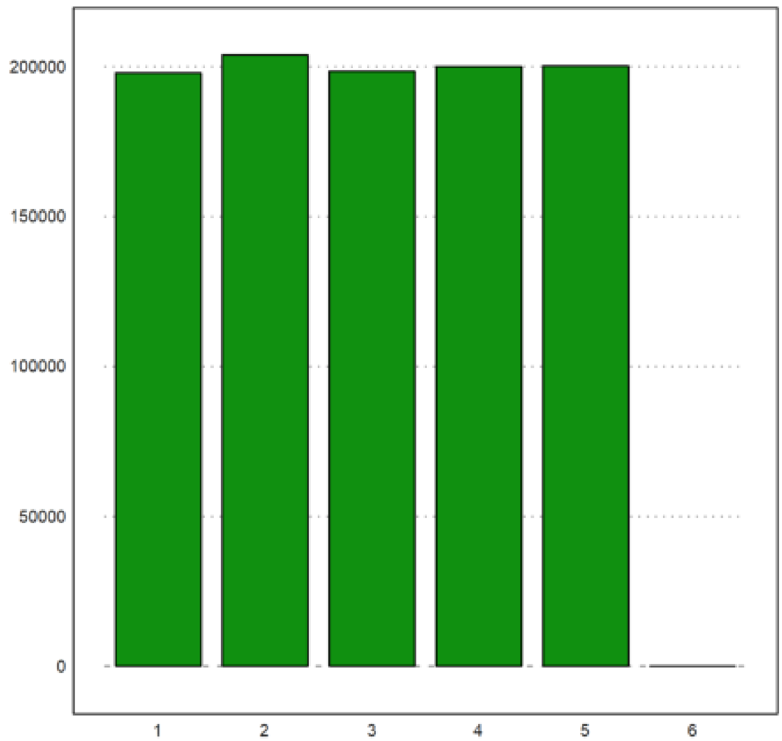
```
[0,  0.198,  0.402,  0.6,  0.8,  1,  1.2]
```

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 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. 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)
```

```
0.7514
```

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,  0.44121]
```

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))
```

```
0.7514
```

invbinsum() menghitung kebalikan dari binomialsum().

```
>2*hypergeomsum(1,5,13,26)
```

```
0.32174
```

Ada juga simulasi distribusi multinomial.Distribusi diskrit dalam statistika adalah distribusi data yang memiliki nilai-nilai yang terpisah dan dapat dihitung. Contohnya adalah jumlah anak dalam sebuah keluarga, jumlah mata dadu yang muncul, atau jumlah pelanggan yang datang ke sebuah toko.

```
>randmultinomial(10,1000,[0.4,0.1,0.5])
```

```
        385         95        520
        407         88        505
        388        106        506
        381        114        505
        413        113        474
        388         99        513
        396         91        513
        399         93        508
        390         89        521
        402        103        495
```

Contoh Soal
Simulasikan 1000 data acak dengan distribusi normal dengan mean 1 dan simpangan baku 2. Hitung rata-rata!
Jawab :
// Simulasi data acak dengan distribusi normal

```
>data = randnormal(1,1000,2)
```

```
[3.9181,  1.9527,  -0.33732,  2.2526,  1.221,  1.5064,  1.0475,
2.1997,  2.5056,  1.9128,  1.727,  2.1483,  2.2224,  3.6901,  1.1749,
1.9995,  0.24513,  2.3104,  3.1337,  0.38286,  2.8783,  1.2136,
2.0146,  -0.097474,  2.1818,  0.0020967,  3.6067,  2.2221,  1.6357,
0.73029,  3.1022,  2.1763,  2.6671,  2.3198,  2.326,  1.5059,  1.4371,
2.7432,  0.1283,  1.6684,  1.9932,  0.76937,  2.134,  1.7466,  1.3792,
1.5807,  0.83863,  2.5515,  3.2569,  4.4061,  1.9229,  3.8785,
1.5262,  2.3406,  1.8594,  2.4003,  2.8752,  2.5498,  2.2527,  2.7602,
2.6761,  1.6431,  3.4518,  2.2219,  1.7896,  2.519,  2.2191,  2.1538,
2.4901,  1.8535,  3.1297,  1.7501,  4.6754,  3.3252,  1.7295,  1.4201,
2.515,  2.5155,  0.51684,  1.727,  1.4341,  3.1193,  1.0781,  1.1891,
0.44626,  3.7094,  1.1403,  0.064038,  0.497,  2.4713,  2.0817,
2.2472,  4.3022,  3.5434,  2.9278,  2.5287,  1.1453,  1.9166,  2.1864,
1.8073,  2.3455,  0.32874,  2.8625,  1.8259,  1.6132,  1.9987,
2.8166,  4.1364,  1.2717,  3.2232,  1.1259,  1.2758,  1.5482,  3.7335,
1.509,  3.0431,  0.60999,  1.0187,  1.8762,  1.9697,  1.6101,  1.8664,
2.1309,  2.2315,  2.0468,  1.559,  3.853,  1.89,  1.5104,  2.1613,
1.5506,  5.3184,  2.3794,  2.3148,  2.5744,  1.7659,  2.6473,
0.66693,  2.6216,  4.014,  3.6484,  -0.083378,  3.5684,  2.7952,
2.1936,  2.381,  1.8078,  0.78504,  3.4552,  1.2495,  0.88303,
 ... ]
```

```
>mean(data)
```

```
1.9987
```

Sub Topik 3: Membaca Data Yang Tersimpan

Membaca Data yang Tersimpan di dalam Berkas dengan Berbagai Format(teks biasa, CSV) untuk di analisis lebih lanjut
membaca data dari teks biasa pertama-tama, kita akan mencoba membaca data dengan teks biasa yang terdapat pada contoh yang telah di berikan pada besmart yaitu dengan menuliskan fungsi "printfile(nama file teks biasa, berapa baris yang akan di print)"
seperti di bawah kita akan print tabel pada data yang terdapat pada buku online "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");
```

pada kasus ini saya telah menunjukan terdapat 4 baris yang telah di print pada emt yang berisi angka dan token(string)
kali ini kita akan membaca tabel dengan lebih mudah atau dengan bahasa kita sendiri. Untuk ini, kita mendefinisikan set token. Fungsi strtokens () mendapatkan vektor string token dari string tertentu.

```
>mf:=["m","f"]; yn:=["y","n"]; ev:=strtokens("g vg m b vb");
```

sekarang kita dapat membacanya dengan cara kita sendiri
Argumen tok2, tok4, dll. Adalah definisi dari kolom tabel. Argumen ini tidak ada dalam daftar parameter readtable (), jadi Anda perlu memberinya ": =" untuk mendefinisikannya.

```
>{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;
```

lalu kita akan print tabel sesuai dengan tabel awal namun dengan bentuk tabel yang berbeda

```
>writetable(MT[1:6],labc=hd,wc=5,tok2:=mf,tok4:=yn,tok5:=ev,tok7:=yn);
```

```
MT is not a variable!
Error in:
writetable(MT[1:6],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 menjadi angka.
String khusus NA = "." diartikan sebagai "Tidak Tersedia", dan mendapatkan NAN (bukan angka) di tabel.
Terjemahan ini dapat diubah dengan parameter NA, dan NAval.

```
>MT[1]
```

```
MT is not a variable!
Error in:
MT[1] ...
     ^
```

Berikut adalah isi tabel dengan bilangan yang belum diterjemahkan.

```
>writetable(MT[1:6],wc=5)
```

```
MT is not a variable!
Error in:
writetable(MT[1:6],wc=5) ...
                  ^
```

Untuk kenyamanan, Anda bisa memasukkan keluaran 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;
```

Dengan menggunakan kolom token yang sama dan token dibaca dari file, kita dapat mencetak tabel. Kita dapat menentukan ctok, tok, dll. Atau menggunakan Tabel daftar.

```
>writetable(Table,ctok=ctok,wc=5);
```

```
Variable or function Table not found.
Error in:
writetable(Table,ctok=ctok,wc=5); ...
                ^
```

tabel sudah dapat di analisis lebih lanjut

membaca data dari CSV

pertama tama kita download file csv yang telah di sediakan di besmart, setelah itu kita jadi satukan dalam 1 folder dengan file emt kita. lalu masukan file tersebut dengan definisi file="nama file csv"

```
>file="test.csv";  ...
>M=random(3,3); writematrix(M,file)
```

M mendefinisikan sebagai matrix
random(n,m) mendefinisikan matrix dengan variabel acak yang akan di keluarakan
writematrix digunakan untuk menuliskan matriks yang ada
lalu kita print datanya dengan

```
>printfile(file)
```

```
0.4191983703672241,0.4504034185243261,0.8530870403686531
0.7175631662797505,0.6559204322999515,0.883095901954524
0.07277738544441656,0.8168773103666358,0.5703787256414963
```

titik desimal pada data tersebut dapat di jadikan pada format EMT dengan cara menggunakan readmatrix()

```
>readmatrix(file)
```

```
    0.4192     0.4504    0.85309
   0.71756    0.65592     0.8831
  0.072777    0.81688    0.57038
```

```
>
```

Di Excel atau spreadsheet serupa, Anda dapat mengekspor matriks sebagai CSV (nilai 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.
Berikut ini contohnya.

```
>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 EMT. Cara termudah untuk membaca ini ke dalam Euler adalah readmatrix (). Semua koma diganti dengan titik dengan parameter> koma. Untuk CSV bahasa Inggris, cukup abaikan 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;
```

data siap di analisis lebih lanjut

```
>reset;
```

Latihan nomer 1

```
>file="sample.csv"
```

```
sample.csv
```

```
>printfile(file,7)
```

```
Could not open the file
sample.csv
for reading!
Try "trace errors" to inspect local variables after errors.
printfile:
    open(filename,"r");
```

```
>ctok=[1]; {MT,hd,tok}=readtable("sample.csv",ctok=ctok);
```

```
Could not open the file
sample.csv
for reading!
Try "trace errors" to inspect local variables after errors.
readtable:
    if filename!=none then open(filename,"r"); endif;
```

```
>tok
```

```
Variable tok not found!
Error in:
tok ...
   ^
```

```
>MT[2]
```

```
MT is not a variable!
Error in:
MT[2] ...
     ^
```

```
>writetable(MT[1:6],wc=5)
```

```
MT is not a variable!
Error in:
writetable(MT[1:6],wc=5) ...
                  ^
```

```
>Table={{readtable(file,ctok=ctok)}};
```

```
Could not open the file
sample.csv
for reading!
Try "trace errors" to inspect local variables after errors.
readtable:
    if filename!=none then open(filename,"r"); endif;
```

```
>writetable(Table,ctok=ctok,wc=10);
```

```
Variable or function Table not found.
Error in:
writetable(Table,ctok=ctok,wc=10); ...
                ^
```

```
>reset;
```

---

nomer 2

```
>file="test.dat"
```

```
test.dat
```

```
>printfile(file,7)
```

```
Could not open the file
test.dat
for reading!
Try "trace errors" to inspect local variables after errors.
printfile:
    open(filename,"r");
```

```
>mf:=["m","f"];
>{MT,hd}=readtable("table1.dat",tok2:=mf);
```

```
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;
```

```
>load over statistics;
>writetable(MT[1:6],labc=hd,wc=5,tok2:=mf);
```

```
MT is not a variable!
Error in:
writetable(MT[1:6],labc=hd,wc=5,tok2:=mf); ...
                  ^
```

```
>reset;
```

Sub Topik 4: Membaca data dari internet

Membaca data dari internet
Situs web atau file dari URL dapat dibuka dengan menggunakan EMT dan dapat dibaca baris demi baris.
Berikut contoh penggunaan EMT untuk membuka url untuk mengetahui versi dari EMT

```
>function readversion () ...
```

```
urlopen("http://www.euler-math-toolbox.de/Programs/Changes.html"); //membuka url
repeat //loop yang berlangsung sampai akhir file url
until urleof();
s=urlgetline();  //membaca baris teks
k=strfind(s,"Version",1);  //mencari substring"Version". jika ditemukan akan disimpan di
if k>0 then substring(s,k,strfind(s,"<",k)-1), break; endif; //berhenti sebelum <
end;
urlclose();
endfunction
```

```
> readversion
```

```
Version 2022-05-18
```

```
>function readdataurl () ...
```

```
urlopen("https://kumparan.com/berita-terkini/3-contoh-soal-desil-data-tunggal-beserta-ku
repeat
until urleof();
s=urlgetline();
k=strfind(s,"Tentukan persentil",1);
if k>0 then substring(s,k,strfind(s,".",k)-1), break; endif;
end;
urlclose();
endfunction
```

Selanjutnya kita mencoba dengan cara yang sama untuk mengambil soal dari website yang ada di internet

```
>readdataurl
```

Sub Topik 5: Perhitungan terkait analisis data statistika deskriptif

Rata-rata, simpangan baku, jangkauan, modus, ukuran data,varians dan median.

Analisis data statistika deskriptif

Statistika deskriptif adalah bidang ilmu statistika yang mempelajari cara-cara untuk pengumpulan, penyusunan, dan penyajian data sehingga memberikan informasi yang berguna. Perlu diketahui juga bahwa statistika deskriptif memberikan informasi hanya mengenai data yang dipunyai dan sama sekali tidak menarik inferensia atau kesimpulan apapun tentang gugus data induknya yang lebih besar.
Dalam praktiknya,analisis data statistika deskriptif bisa dilakukan dengan menerapkan sejumlah metode statistik, seperti :

1. Mencari rata rata/mean

Metode pertama yang digunakan untuk melakukan analisis statistika adalah mean atau sering disebut rata-rata. Saat akan menghitung rata-rata, kita bisa melakukan dengan cara menambahkan daftar angka kemudian membagi angka tersebut dengan jumlah item dalam daftar. Metode ini memungkinkan penentuan tren keseluruhan dari kumpulan data dan mampu mendapatkan tampilan data yang cepat dan ringkas. Manfaat dari metode ini juga termasuk perhitungan yang sederhana dan cepat.

a. Rata-rata hitung data tunggal

Misalkan

$$x_1, x_2, x_3, ..., x_n$$

adalah data yang dikumpulkan dari suatu sampel atau populasi maka rata-rata hitung untuk sampel disimbolkan dengan

$$\bar{x}$$

dan rata-rata hitung untuk populasi disimbolkan dengan

$$\mu$$

Sehingga, untuk mencari rata-rata hitung data tunggal terdapat 2 jenis rumus sebagai berikut :
1. Rata-rata hitung sampel

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$

2. Rata-rata hitung populasi,

$$\mu = \frac{\sum_{i=1}^{n} X_i}{n}$$

Untuk menghitung rata-rata data tunggal dengan EMT, kita dapat menggunakan sintaks
> mean ([data])
Contoh Soal:
1. Diketahui data usia(dalam tahun) penduduk suatu daerah adalah sebagai berikut:
60,70,66,75,77,68,45,30,15,71,69,84,13
hitunglah rata-rata usia penduduk tersebut.
Jawab :

```
>mean([60,70,66,75,77,68,45,30,15,71,69,84,13])
```

```
57.1538461538
```

Jadi, rata rata data tersebut adalah 57.1538461538
2. Nilai ulangan matematika dari 10 siswa adalah 80, 88, 70, 60, 90, 75, 92, 78, 67, 90. Tentukan rata-rata dari data tersebut!
Jawab :

```
>mean([80, 88, 70, 60, 90, 75, 92, 78, 67, 90])
```

```
79
```

Jadi, rata-rata dari data tersebut yaitu 79

b. Rata-rata data tabel distribusi

Jika diberikan data

$$x_1, x_2, ..., x_n$$

yang memiliki frekuensi berturut- turut

$$f_1, f_2, ..., f_n$$

maka, rataan hitung dari data yang disajikan dalam daftar distribusi tersebut ditentukan dengan 2 jenis rumus sebagai berikut :
1. Rata-rata hitung sampel
Untuk rata-rata hitung sampel,

$$\bar{x} = \frac{\sum_{i=1}^{n} f_i x_i}{\sum_{i=1}^{n} f_i}$$

2.Rata-rata hitung populasi
Untuk rata-rata hitung populasi,

$$\mu = \frac{\sum_{i=1}^{n} f_i x_i}{\sum_{i=1}^{n} f_i}$$

Cara diatas adalah beberapa perhitungan untuk mencari rata-rata data tabel distribusi menggunakan metode yang ada dalam statistika. Dengan menggunakan EMT kita juga bisa menghitung rata-rata data tabel distribusi dengan mudah, yaitu dengan cara berikut:
1. Mendeskripsikan data dan frekuensi
2. Menghitung rata-rata menggunakan perintah berikut :
> mean(data,frekuensi)
Contoh soal:
Diberikan data berat badan siswa kelas V SD yang memiliki jumlah siswa sebanyak 35 orang anak. anak dengan berat 30kg terdapat 5 orang, anak dengan berat 35kg terdapat 11 orang, anak dengan berat 40kg terdapat 4 orang, anak dengan berat 38kg terdapat 7 orang, anak dengan berat 44kg terdapat 7 orang, dan anak dengan berat 50kg terdapat 1 orang. Tentukan rata-rata berat siswa kelas V SD tersebut!
Jawab :

```
>printfile("tabel berat badan kelas V SD.dat",7); //meringkas informasi pada soal dengan m
```

```
Could not open the file
tabel berat badan kelas V SD.dat
for reading!
Try "trace errors" to inspect local variables after errors.
printfile:
    open(filename,"r");
```

```
>data=[30,35,38,40,44,50]//mendefinisikan data sebagai berat siswa dalam satuan kilogram
```

```
[30,  35,  38,  40,  44,  50]
```

```
>frekuensi=[5,11,7,4,7,1]//mendefinisikan frekuensi sebagai banyak siswa
```

```
[5,  11,  7,  4,  7,  1]
```

```
>mean(data,frekuensi) //menghitung rata-rata
```

```
37.6857142857
```

Jadi, rata-rata berat badan siswa SD kelas V adalah 37.6857142857 **c. Rata-rata hitung data kelompok**

Misalkan suatu data kelompok terdiri dari n kelas dengan nilai tengah masing-masing kelas secara berturut-turut adalah

$$t_1, t_2, ..., t_n$$

dan masing-masing frekuensinya adalah

$$f_1, f_2, ..., f_n$$

Untuk mencari rata rata hitung data tersebut terdapat 2 jenis rumus sebagai berikut :
1. Rata-rata hitung sampel
untuk rata-rata hitung sampel,

$$\bar{x} = \frac{\sum_{i=1}^{n} t_i f_i}{\sum_{i=1}^{n} f_i}$$

2. Rata-rata hitung populasi
untuk rata-rata hitung populasi,

$$\mu = \frac{\sum_{i=1}^{n} t_i f_i}{\sum_{i=1}^{n} f_i}$$

Untuk menghitung rata-rata data kelompok di EMT dapat dilakukan dengan langkah berikut :
1. Menentukan tepi bawah kelas(Tb), panjang kelas(P), dan tepi atas kelas(Ta) dengan rumus :

$$Tb = a - 0,5$$

$$P = (b - a) + 1$$

$$Ta = b + 0,5$$

Keterangan :
a = batas bawah kelas
b = batas atas kelas
2. Membuat data menjadi bentuk tabel, dengan perintah
> r= tepi bawah terkecil : panjang kelas : tepi atas terbesar;
f=[frekuensi];
>T:r[1:jumlah kelas]' | r[2:jumlah kelas + 1]' |f';
writetable(T, labc=["tepi bawah", "tapi atas", "frekuensi"])
3. Menghitung nilai tengah kelas, dengan perintah
>T[,1]+T[,2]/2
4. Mengubah baris menjadi kolom
>t=fold(r,[0.5,0.5])
5. Menghitung rata-rata, dengan perintah
>mean(t,f)
Contoh soal :
1. Disajikan data kelompok seperti berikut :

```
>printfile("Tabel rata-rata data kelompok.dat",7)
```

```
Could not open the file
Tabel rata-rata data kelompok.dat
for reading!
Try "trace errors" to inspect local variables after errors.
printfile:
    open(filename,"r");
```

```
>31-0.5  //Tepi bawah terkecil
```

```
30.5
```

```
>(40-31)+1  //Panjang kelas
```

```
10
```

```
>90+0.5 //Tepi atas kelas
```

```
90.5
```

```
>r=30.5:10:90.5; f=[3, 5, 10, 11, 8, 3];
>T:=r[1:6]' | r[2:7]' | f' ; writetable(T,labc=["tepi bawah", "tepi atas", "frekuensi"])
```

```
 tepi bawah tepi atas frekuensi
       30.5      40.5         3
       40.5      50.5         5
       50.5      60.5        10
       60.5      70.5        11
       70.5      80.5         8
       80.5      90.5         3
```

```
>t=(T[,1]+T[,2])/2  //menghitung nilai tengah kelas
```

```
        35.5
        45.5
        55.5
        65.5
        75.5
        85.5
```

```
>t=fold(r,[0.5,0.5]) // mengubah tampilan data kolom menjadi baris dan sebaliknya
```

```
[35.5,  45.5,  55.5,  65.5,  75.5,  85.5]
```

```
>mean(t,f)
```

```
61.75
```

Jadi, rata-rata data kelompok tersebut adalah 61,75

2. Diberikan data kelompok berikut yang mewakili jumlah jam belajar per minggu dari sekelompok siswa :

```
>printfile("Tabel data kelompok conso 2.dat",5)
```

```
Could not open the file
Tabel data kelompok conso 2.dat
for reading!
Try "trace errors" to inspect local variables after errors.
printfile:
    open(filename,"r");
```

Hitunglah rata-rata jumlah jam belajar per minggu dari data kelompok tersebut!
Jawab :

```
>10-0.5  //tepi bawah terkecil
```

```
9.5
```

```
>(14-10)+1  //panjang kelas
```

```
5
```

```
>29+0.5  //tepi atas terbesar
```

```
29.5
```

```
>r=9.5:5:29.5; f=[5, 8, 12, 6];
>T:=r[1:4]' | r[2:5]' | f'; writetable(T,labc=["tepi bawah", "tepi atas", "frekuensi"])
```

```
 tepi bawah tepi atas frekuensi
        9.5      14.5         5
       14.5      19.5         8
       19.5      24.5        12
       24.5      29.5         6
```

```
>t=(T[,1]+T[,2])/2  // menghitung nilai tengah kelas
```

```
          12
          17
          22
          27
```

```
>t=fold(r,[0.5,0.5]) // mengubah tampilan data kolom menjadi baris dan sebaliknya
```

```
[12,  17,  22,  27]
```

```
>mean(t,f)
```

```
20.064516129
```

Jadi, rata-rata data kelompok tersebut yaitu 20.064516129

2. Mencari median

Median (Me) adalah nilai tengah dari suatu data yang telah disusun dari data terkecil sampai data terbesar atau sebaliknya. Selain sebagai ukuran pemusatan data, median juga dijadikan sebagai ukuran letak data dan dikenal sebagai kuartil 2 (Q2). Rumus perhitungan median dibedakan untuk data tak berkelompok dan data berkelompok.

a. Median data tunggal

Median data tunggal adalah mengurutkan data berdasarkan nilainya, misalkan data yang telah terurut dari data terkecil ke data terbesar adalah

$$x_1, x_2, ..., x_n$$

```
untuk menentukan letak median dengan menggunakan rumus :
```

1. Jika jumlah suatu data(n) berjumlah ganjil maka nilai mediannya adalah sama dengan data yang memiliki nilai di urutan paling tengah yang memiliki nomor urut k, dimana untuk menentukan nilai k dapat dihitung menggunakan rumus:

$$k = \frac{n+1}{2}$$

2. Jika jumlah suatu data (n) berjumlah genap, maka untuk menghitung mediannya dengan menggunakan rumus :

$$k = \frac{n}{2}$$

$$Median = \frac{1}{2}(x_k + x_{k+1})$$

Diatas adalah rumus untuk mencari median secara statistika. Dengan menggunakan EMT kita bisa menentukan median dengan menggunakan perintah
> median([data])
perintah tersebut dapat berjalan dengan baik apabila data sudah diurutkan terlebih dahulu dari data terkecil hingga terbesar.
Contoh soal :
Diketahui data hasil tes SKD calon PNS adalah sebagai berikut :
487, 300, 450, 500, 521, 440
Tentukan nilai median dari data tersebut!
Jawab :

```
>data=[487, 300, 450, 500, 521, 440]; //mendeskripsikan data
>urutan=sort(data)  //mengurutkan data
```

```
[300,  440,  450,  487,  500,  521]
```

```
>median([urutan])
```

```
468.5
```

Jadi, nilai median dari data hasil tes SKD adalah 468.5

b. Median data kelompok

Menghitung median data kelompok dapat menggunakan rumus di bawah ini :

$$M_e = Tb + p\frac{\frac{1}{2}n - F}{f}$$

Keterangan:
Tb = tepi bawah kelas median, ialah kelas dimana median terletak
p = panjang kelas median
n = ukuran sampel / banyak data
F = jumlah semua frekuensi dengan tanda kelas lebih kecil dari tanda kelas median.
f = frekuensi kelas median
Untuk menghitung median data berkelompok di EMT, dapat dilakukan dengan cara berikut:
1. Menentukan tepi bawah kelas (Tb), panjang kelas (P), dan tepi atas kelas (Ta) dengan rumus :

$$T_b = a - 0,5$$

$$P = (b - a) + 1$$

$$T_a = b + 0.5$$

2. Mendeskripsikan data dalam bentuk tabel, dengan perintah
> r=tepi bawah terkecil:panjang kelas:tepi atas terbesar; f=[frekuensi];
> T:=r[1:jumlah kelas]' | r[2:jumlah kelas + 1]' | f'; writetable(T,labc=["tepi bawah","tepi atas","frekuensi"]))
3. Mendeskripsikan batas bawah kelas median, panjang kelas median, banyak data, jumlah frekuensi sebelum kelas median, frekuensi median
> Tb=(tepi bawah kelas median), p=(panjang kelas median), n=(banyak data), F=(jumlah frekuensi sebelum kelas median), f=(frekuensi kelas median)
4. Menghitung median data dengan perintah:
> Tb+p*(1/2*n-F)/f
Contoh soal :
Berikut adalah data hasil dari pengukuran berat badan 20 siswa SD kelas V. Dari ke 20 siswa, siswa yang mempunyai berat badan dalam rentang 21-26 kg sebanyak 5 orang, yang mempunyai berat badan dalam rentang 27-32 kg sebanyak 4 orang, yang mempunyai berat badan dalam rentang 33-38 kg sebanyak 3 orang, yang mempunyai berat badan dalam rentang 39-44 kg sebanyak 2 orang, yang mempunyai berat badan dalam rentang 45-50 kg sebanyak 3 orang, dan yang mempunyai berat badan 51-56 kg sebanyak 3 orang. Tentukan median dari
data hasil pengukuran berat badan 20 siswa di SD tersebut!
Penyelesaian:
Menentukan tepi bawah kelas yang terkecil

```
>21-0.5 // menentukan tepi bawah kelas terkecil
```

```
20.5
```

```
> (26-21)+1 // menentukan panjang kelas
```

```
6
```

```
> 56+0.5 // tepi atas kelas terbesar
```

```
56.5
```

```
>r=20.5:6:56.5; f=[5, 4, 3, 2, 3, 3];
>T :=r[1:6]' | r[2:7]' | f' ; writetable(T, labc=["Tb", "Ta", "frekuensi"])
```

```
          Tb          Ta  frekuensi
        20.5        26.5          5
        26.5        32.5          4
        32.5        38.5          3
        38.5        44.5          2
        44.5        50.5          3
        50.5        56.5          3
```

```
>Tb=32.5, p=6, n=20, F=9, f=3
```

```
32.5
6
20
9
3
```

```
>Tb+p*(1/2*n-F)/f
```

```
34.5
```

Jadi, median dari data hasil pengukuran berat badan 20 siswa SD kelas V adalah 34.5

3. Mencari Modus

Modus adalah area fokus dalam analisis statistika deskriptif yang termasuk dalam ukuran pusat data. Ini adalah nilai yang paling sering muncul dalam kumpulan data atau nilai yang memiliki frekuensi tertinggi dalam distribusi data. Modus dapat dibagi menjadi dua jenis, yaitu modus untuk data tunggal dan modus untuk data kelompok.

a.Modus untuk data tunggal:

Menentukan modus untuk data tunggal cukup sederhana. Pertama, data diurutkan dari nilai terkecil ke terbesar sehingga data dengan nilai yang sama berdekatan satu sama lain. Selanjutnya, frekuensi masing-masing data dihitung, dan data yang memiliki frekuensi tertinggi dipilih sebagai modus.

b.Modus untuk data kelompok

Berikut rumus untuk mencari modus data kelompok :

$$M_o = Tb + \frac{d_1}{d_1 + d_2} c$$

Keterangan :
Tb = Tepi bawah
d1 = selisih f modus dengan f sebelumnya
d2 = selisih f modus dengan f sesudahnya

```
c = Panjang kelas
```

Untuk menghitung modus data berkelompok di EMT, dapat dilakukan dengan cara berikut:
1. Menentukan tepi bawah kelas (Tb), panjang kelas (P), dan tepi atas kelas (Ta) dengan rumus :

$$T_b = a - 0,5$$

$$P = (b - a) + 1$$

$$T_a = b + 0.5$$

dimana a = batas bawah kelas dan b = batas atas kelas
2. Mendeskripsikan data dalam bentuk tabel, dengan perintah
> r=tepi bawah terkecil:panjang kelas:tepi atas terbesar; v=[frekuensi];
> T:=r[1:jumlah kelas]' | r[2:jumlah kelas + 1]' | f'; writetable(T,labc=["tepi bawah","tepi atas","frekuensi"]))
3. Mendeskripsikan tepi bawah kelas modus, panjang kelas modus,selisih frekuensi modus dengan frekuensi sebelumnya, selisih frekuensi modus dengan frekuensi sesudahnya
> Tb=(tepi bawah kelas modus), p=(panjang kelas modus), d1=(selisih frekuensi modus dengan frekuensi sebelumnya), d2=(selisih frekuensi dengan frekuensi sesudahnya)
4. Menghitung modus dengan perintah:
> Tb+p*d1/(d1+d2)
Contoh soal
Diketahui sebuah data kelompok sebagai berikut :

```
>printfile("Tabel modus data kelompok.dat",8)
```

```
Could not open the file
Tabel modus data kelompok.dat
for reading!
Try "trace errors" to inspect local variables after errors.
printfile:
    open(filename,"r");
```

Berapakah modus dari data tersebut?

```
>20-0.5  //menentukan tepi bawah kelas
```

```
19.5
```

```
>(29-20)+1 //menentukan panjang kelas
```

```
10
```

```
>89+0.5 //menentukan tepi atas
```

```
89.5
```

```
>r=19.5:10:89.5; f=[3, 7, 8, 12, 9, 6, 5];
>T:=r[1:7]' | r[2:8]' | f'; writetable(T,labc=["Tb", "Ta", "frekuensi"])
```

```
       Tb         Ta  frekuensi
     19.5       29.5          3
     29.5       39.5          7
     39.5       49.5          8
     49.5       59.5         12
     59.5       69.5          9
     69.5       79.5          6
     79.5       89.5          5
```

Berdasarkan tabel di atas, modus berada pada kelas 49.5-59.5

```
>Tb=49.5, p=10, d1=12-8, d2=12-9
```

```
49.5
10
4
3
```

```
>Tb+p*d1/(d1+d2)
```

```
55.2142857143
```

Jadi, modus dari data kelompok di atas adalah 55.2142857143

4. Mencari varians/ragam

Varians digunakan untuk mengetahui bagaimana sebaran data terhadap mean atau nilai rata-rata. Sederhananya, varians adalah ukuran statistik jauh dekatnya penyebaran data dari nilai rata-ratanya. Dalam mencari ragam dapat dikelompokkan menjadi 2 jenis yaitu sebagai berikut :

a.Varians data tunggal

Rumus untuk varians data tunggal berikut :
1) Untuk populasi

$$\sigma^2 = \frac{\sum_{i=1}^{n}(x-\mu)^2}{n}$$

2) Untuk sampel

$$S^2 = \frac{\sum_{i=1}^{n}(x-\bar{x})^2}{n-1}$$

Pada EMT, untuk menemukan suatu Ragam data tunggal dapat menggunakan perintah berikut:
> mean(dev^2)
Contoh soal
Hitunglah nilai varians dari data sampel nilai siswa: 9, 10, 6, 7!

```
>data=[9, 10, 6, 7]; //mendefinisikan data
>urut=sort(data) //mengurutkan data
```

```
[6,  7,  9,  10]
```

```
>xbar=mean(urut) //menghitung rata rata dari data
```

```
8
```

```
>dev= urut-xbar
```

```
[-2,  -1,  1,  2]
```

```
>varians=mean(dev^2) //menghitung varians
```

```
2.5
```

Jadi, varians dari data sampel tersebut adalah 2.5

b. Varians data kelompok

Rumus untuk varians data kelompok sebagai berikut :
1) Untuk populasi

$$\sigma^2 = \frac{\sum_{i=1}^{n} f_i(x_i - \mu)^2}{\sum_{i=1}^{n} f_i}$$

2) Untuk sampel

$$S^2 = \frac{\sum_{i=1}^{n} f_i(x_i - \bar{x})^2}{\sum_{i=1}^{n} f_i - 1}$$

Pada EMT, untuk menemukan Ragam data berkelompk dapat menggunakan perintah berikut:
1. Menentukan tepi bawah kelas (Tb), panjang kelas (P), dan tepi atas kelas (Ta) dengan rumus :

$$T_b = a - 0,5$$

$$P = (b - a) + 1$$

$$T_a = b + 0.5$$

dengan a = batas bawah kelas dan b = batas atas kelas
2. Mendeskripsikan data dalam bentuk tabel, dengan perintah
> r=tepi bawah terkecil:panjang kelas:tepi atas terbesar; f=[frekuensi];
> T:=r[1:jumlah kelas]' | r[2:jumlah kelas + 1]' | f'; writetable(T,labc=["tepi bawah","tepi atas","frekuensi"])
3. Menghitung Ragam dengan perintah
> (T[,1]+T[,2])/2; t=fold(r,[0.5,0.5]);m=mean(t,f);
> sum(f*(t-m)^2)/sum(f) //untuk populasi
> sum(f*(t-m)^2)/(sum(f)-1) //untuk sampel
Contoh soal
Tentukan varians data sampel dari tabel berikut :

```
>printfile("Tabel data kelompok varians.dat",7)
```

```
Could not open the file
Tabel data kelompok varians.dat
for reading!
```

```
Try "trace errors" to inspect local variables after errors.
printfile:
    open(filename,"r");
```

```
>63-0.5  //tapi bawah terkecil
```

```
62.5
```

```
>(67-63)+1  //panjang kelas
```

```
5
```

```
>92+0.5  //tepi atas terbesar
```

```
92.5
```

```
>r=62.5:5:92.5; f=[3,2,7,3,4,1];
>T:=r[1:6]' | r[2:7]' | f'; writetable(T, labc=["tepi bawah", "tepi atas", "frekuensi"])
```

```
 tepi bawah tepi atas frekuensi
       62.5      67.5         3
       67.5      72.5         2
       72.5      77.5         7
       77.5      82.5         3
       82.5      87.5         4
       87.5      92.5         1
```

```
>(T[,1]+T[,2])/2; t=fold(r,[0.5,0.5])
```

```
[65,  70,  75,  80,  85,  90]
```

```
>m=mean(t,f)
```

```
76.5
```

```
>sum(f*(t-m)^2)/(sum(f)-1)
```

```
52.8947368421
```

Jadi, varians dari data kelompok dari tabel di atas adalah 52.8947368421

```
>
```

5. Mencari Simpangan Baku

Standar Deviasi atau simpangan baku adalah akar dari ragam/varians. Untuk nenetukan nilai standar deviasi, caranya:

$$\sigma = \sqrt{\sigma^2}$$

$$atau$$

$$S = \sqrt{S^2}$$

a. Simpangan baku data tunggal

Untuk data tunggal, simpangan baku populasi atau sampel dapat dirumuskan sebagai berikut:
1) Untuk populasi

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x-\mu)^2}{n}}$$

2) Untuk sampel

$$S = \sqrt{\frac{\sum_{i=1}^{n}(x-\bar{x})^2}{n-1}}$$

Pada EMT, untuk menemukan suatu Ragam data tunggal dapat menggunakan perintah berikut:
> sqrt(mean(dev^2))
Contoh soal :
1. Simpangan baku untuk data 70,80,40,25,65,87,97,59,24,77,45 adalah
Jawab :

```
>data=[70,80,40,25,65,87,97,59,24,77,45];
>urut=sort(data)
```

```
[24,  25,  40,  45,  59,  65,  70,  77,  80,  87,  97]
```

```
>x=mean(urut)
```

```
60.8181818182
```

```
>dev=urut-x
```

```
[-36.8182,  -35.8182,  -20.8182,  -15.8182,  -1.81818,  4.18182,
9.18182,  16.1818,  19.1818,  26.1818,  36.1818]
```

```
>varians=mean(dev^2)
```

```
550.148760331
```

```
>simpanganbaku= sqrt(varians)
```

```
23.4552501656
```

Jadi, simpang baku data tersebut adalah 23.4552501656

b. Simpangan baku data kelompok

```
Untuk data berkelompok dapat dirumuskan seperti berikut:
```

1) Untuk populasi

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} f_i(x_i - \mu)^2}{\sum_{i=1}^{n} f_i}}$$

2) Untuk sampel

$$S = \sqrt{\frac{\sum_{i=1}^{n} f_i(x_i - \bar{x})^2}{\sum_{i=1}^{n} f_i - 1}}$$

Pada EMT, untuk menemukan Ragam data berkelompk dapat menggunakan perintah berikut:
1. Menentukan tepi bawah kelas (Tb), panjang kelas (P), dan tepi atas kelas (Ta) dengan rumus :

$$T_b = a - 0,5$$

$$P = (b - a) + 1$$

$$T_a = b + 0.5$$

dengan a = batas bawah kelas dan b = batas atas kelas
2. Mendeskripsikan data dalam bentuk tabel, dengan perintah
> r=tepi bawah terkecil:panjang kelas:tepi atas terbesar; f=[frekuensi];
> T:=r[1:jumlah kelas]' | r[2:jumlah kelas + 1]' | f'; writetable(T,labc=["tepi bawah","tepi atas","frekuensi"])
3. Menghitung Ragam dengan perintah
> (T[,1]+T[,2])/2; t=fold(r,[0.5,0.5]);m=mean(t,f);
> sqrt(sum(f*(t-m)^2)/sum(f)) // untuk populasi
> sqrt(sum(f*(t-m)^2)/(sum(f)-1)) // untuk sampel
Contoh soal :
Simpangan baku dari tabel dibawah ini adalah

```
>printfile("Tabel simpangan baku data kelompok.dat",7)
```

```
Could not open the file
Tabel simpangan baku data kelompok.dat
for reading!
Try "trace errors" to inspect local variables after errors.
printfile:
    open(filename,"r");
```

```
>41-0.5 //tepi bawah terkecil
```

```
40.5
```

```
>(45-41)+1 //panjang kelas
```

```
5
```

```
>70+0.5 //tepi atas terbesar
```

```
70.5
```

```
>r=40.5:5:70.5; f=[10,12,18,34,20,6];
>T:=r[1:6]' | r[2:7]' | f'; writetable(T,labc=["tepi bawah", "tepi atas", "frekuensi"])
```

```
 tepi bawah tepi atas frekuensi
       40.5      45.5        10
       45.5      50.5        12
       50.5      55.5        18
       55.5      60.5        34
       60.5      65.5        20
       65.5      70.5         6
```

```
>(T[,1]+T[,2])/2; t=fold(r,[0.5,0.5]); m=mean(t,f);
```

karena data tersebut merupakan data sampel, maka menggunakan rumus berikut

```
>sqrt(sum(f*(t-m)^2)/(sum(f)-1))
```

```
6.81649810861
```

Jadi, simpangan baku data kelompok tersebut adalah 6.81649810861

6.Mencari Jangkauan/Range

Jangkauan, atau biasa disebut range, merupakan perbedaan antara nilai data tertinggi dan nilai data terendah dalam suatu set data. Metode pencarian jangkauan berbeda antara data tunggal dan data kelompok.

a. Jangkauan/Range Data Tunggal

Bila ada sekumpulan data tunggal terurut dari yang terkecil sampai terbesar adalah

$$x_1, x_2, ..., x_n$$

maka jangkauannya adalah:

$$Jangkauan = x_n - n_1$$

Untuk menemukan jangkauan data tunggal di EMT dapat menggunakan perintah berikut:
> x=[data]; max(x)-min(x)
Contoh soal
Jangkauan dari data 30,60,87,55,87,98,22,75,81,70,69,84,75 adalah...

```
>x=[30,60,87,55,87,98,22,75,81,70,69,84,75]; max(x)- min(x)
```

```
76
```

Jadi, jangkauan dari data tersebut adalah 76

b. Jangkauan data kelompok

Jangkauan pada data berkelompok adalah selisih antara batas atas dari kelas tertinggi dengan batas bawah dari kelas terendah.
Pada EMT, untuk menemukan jangkauan dari data berkelompok dapat menggunakan perintah berikut:
1. Menentukan tepi bawah kelas (Tb), panjang kelas (P), dan tepi atas kelas (Ta) dengan rumus :

$$T_b = a - 0,5$$

$$P = (b - a) + 1$$

$$T_a = b + 0.5$$

dengan a = batas bawah kelas dan b = batas atas kelas
2. Mendeskripsikan data dalam bentuk tabel, dengan perintah
> r=tepi bawah terkecil:panjang kelas:tepi atas terbesar; f=[frekuensi];
> T:=r[1:jumlah kelas]' | r[2:jumlah kelas + 1]' | f'; writetable(T,labc=["tepi bawah","tepi atas","frekuensi"])
3. Menghitung jangkauan data berkelompok
> max(transpose(T[,2]))-min(transpose(T[,1]))
Contoh soal :
Berikut adalah data hasil dari pengukuran berat badan 20 siswa SD kelas V. Dari ke 20 siswa,siswa yang mempunyai berat badan dalam rentang 21-26 kg sebanyak 5 orang, yang mempunyai berat badan dalam rentang 27-32 kg sebanyak 4 orang, yang mempunyai berat badan dalam rentang 33-38 kg sebanyak 3 orang, yang mempunyai berat badan dalam rentang 39-44 kg sebanyak 2 orang, yang mempunyai berat badan dalam rentang 45-50 kg sebanyak 3 orang, dan yang mempunyai berat badan 51-56 kg sebanyak 3 orang. Tentukan jangkauan dari
data hasil pengukuran berat badan 20 siswa di SD tersebut!
Jawab :

```
>printfile("Tabel jangkauan data kelompok.dat",7) //menyederhanakan informasi
```

```
Could not open the file
Tabel jangkauan data kelompok.dat
for reading!
Try "trace errors" to inspect local variables after errors.
printfile:
    open(filename,"r");
```

```
>21-0.5 //tepi bawah terkecil
```

```
20.5
```

```
>(26-21)+1 //panjang kelas
```

```
6
```

```
>56+0.5 //tepi atas terbesar
```

```
56.5
```

```
>r=20.5:6:56.5; f=[5,4,3,2,3,3];
>T:=r[1:6]' | r[2:7]' | f'; writetable(T,labc=["tepi bawah","tepi atas","frekuensi"])
```

```
 tepi bawah tepi atas frekuensi
       20.5      26.5         5
       26.5      32.5         4
       32.5      38.5         3
       38.5      44.5         2
       44.5      50.5         3
       50.5      56.5         3
```

```
>max(transpose(T[,2]))-min(transpose(T[,1]))
```

```
36
```

Jadi, Jangkauan dari data kelompok tersebut adalah 36

7.Menentukan ukuran letak

Ukuran letak merupakan ukuran untuk melihat dimana letak salah satu data dari sekumpulan banyak data yang ada. Yang termasuk ukuran ukuran letak antara lain adalah kuartil(Q), desil(D) dan persentil(P). Dalam menentukan ke-3 nya yang harus diingat adalah mengurutkan distribusi data dari yang terkecil sampai terbesar
1. Kuartil
Dalam EMT untuk menghitung kuartil bisa dilakukan dengan perintah
>quartiles(data)
perintah tersebut akan menghasilkan nilai Q1, Q2, Q3, nilai minimum dan nilai maksimum dari suatu data
2. Desil
Dalam EMT untuk menghitung desil bisa dilakukan dengan perintah
>quantile(data)
3. Persentil
Dalam EMT untuk menghitung persentil bisa dilakukan dengan perintah
>quantile(data)
perintah ">quantile(data)" dapat digunakan untuk menentukan desil dan persentil perbedaannya tergantung pada nilai dari pembaginya
Contoh soal
1. Tentukan Q1,Q2 dan Q3 dari data : 7,3,8,5,9,4,8,3,10,2,7,6,8,7,2,6,9.

```
>data=[7,3,8,5,9,4,8,3,10,2,7,6,8,7,2,6,9];
>urut=sort(data)
```

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

```
>quartiles(urut)
```

```
[2,  3.5,  7,  8,  10]
```

dari hasil di atad diperoleh nilai sebagai berikut :
Nilai minimal data = 2
Q1=3.5
Q2=7
Q3=8
Nilai maksimal data = 10
2. Tentukan D8 dari data : 6,3,8,9,5,9,9,7,5,7,4,5,8,3,7,6

```
>data=[6,3,8,9,5,9,9,7,5,7,4,5,8,3,7,6];
>urut=sort(data)
```

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

```
>quantile(urut,0.8)  //nilai 0.8 diapatkan karena kita akan mencari D8
```

```
8
```

Jadi, nilai dari D8 berdasarkan perhitungan di atas adalah 8
3.Tentukan persentil ke-65 dari data : 6,5,8,7,9,4,5,8,4,7,8,5,8,4,5

```
>data=[6,5,8,7,9,4,5,8,4,7,8,5,8,4,5];
>urut=sort(data)
```

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

```
>quantile(urut,65%)
```

```
7.1
```

Sub Topik 6: Menggambar Grafik Statistika

Diagram Kotak

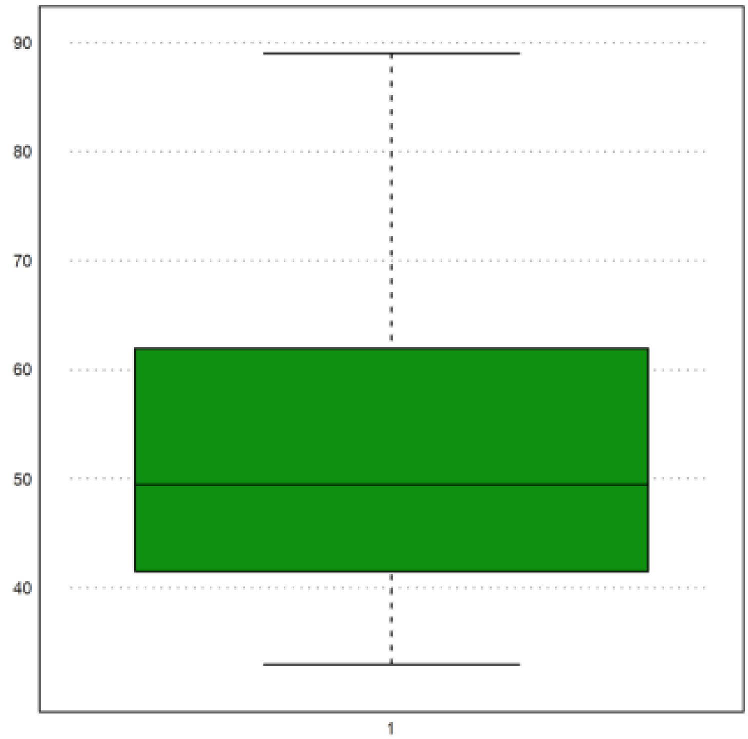
Diagram kotak atau box plot merupakan ringkasan distribusi sampel yang disajikan secara grafis yang bisa menggambarkan bentuk distribusi data (skewness), ukuran tendensi sentral dan ukuran penyebaran (keragaman) data pengamatan. Diagram kotak sering digunakan ketika jumlah distribusi data perlu dibandingkan. Diagram kotak menyajikan informasi tentang nilai–nilai inti dalam distribusi data termasuk juga pencilan. Pencilan adalah titik data yang terpaut jauh dari titik data lainnya.
Contoh:
Diketahui data berat badan mahasiswa di Universitas A sebagai berikut.

```
>A=[55,50,33,42,44,37,63,74,56,34,51,43,45,39,64,77,60,35,53,43,48,41,65,87,61,36,54,44,49
```

```
[55,  50,  33,  42,  44,  37,  63,  74,  56,  34,  51,  43,  45,  39,
64,  77,  60,  35,  53,  43,  48,  41,  65,  87,  61,  36,  54,  44,
49,  41,  66,  89]
```

Buatlah diagram kotak (box plot) kemudian tuliskan interpretasinya.

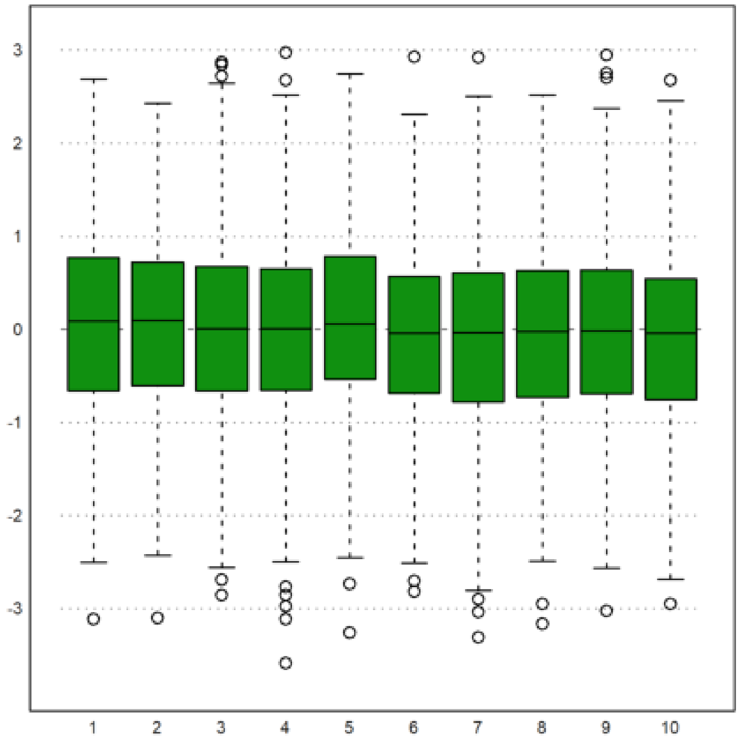
```
>boxplot(A):
```



Dari gambar box plot berat badan mahasiswa Universitas A, sepintas kita bisa menentukan beberapa ukuran statistik, meskipun tidak persis sekali. Nilai statistik pada badan boxplot berkisar pada: Nilai Minimum = 33 , Q1 = 41.5 , Median (Q2) = 49.5 , Q3 = 62 , Nilai Maksimum = 89 . Sebaran data tidak simetris, melainkan menjulur ke arah kanan (postively skewness). Karena nilai jarak Q1 dengan Q2 lebih pendek dari jarak Q2 dengan Q3, maka data lebih terpusat di kiri. Akan tetapi data tersebut tergolong cenderung mesokurtik karena jarak IQR dengan panjang hampir sama, dengan data berpusat di angka 49.5

Adapun contoh perbandingan 10 simulasi 500 nilai terdistribusi normal menggunakan box plot dan terdapat pencilan sebagai berikut.

```
> p=normal(10,500); boxplot(p):
```

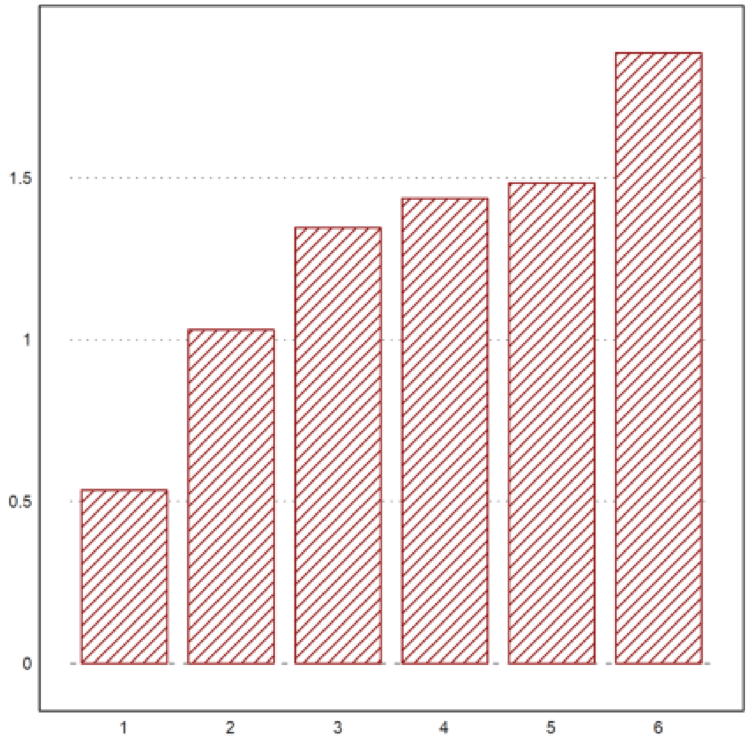


pada diagram diatas, adalah membuat boxplot distribusi normal dengan rata-rata 10 dan standar deviasi 500. Boxplot adalah representasi grafis dari lokalitas, penyebaran, dan kecondongan sekelompok data numerik melalui kuartil mereka
2

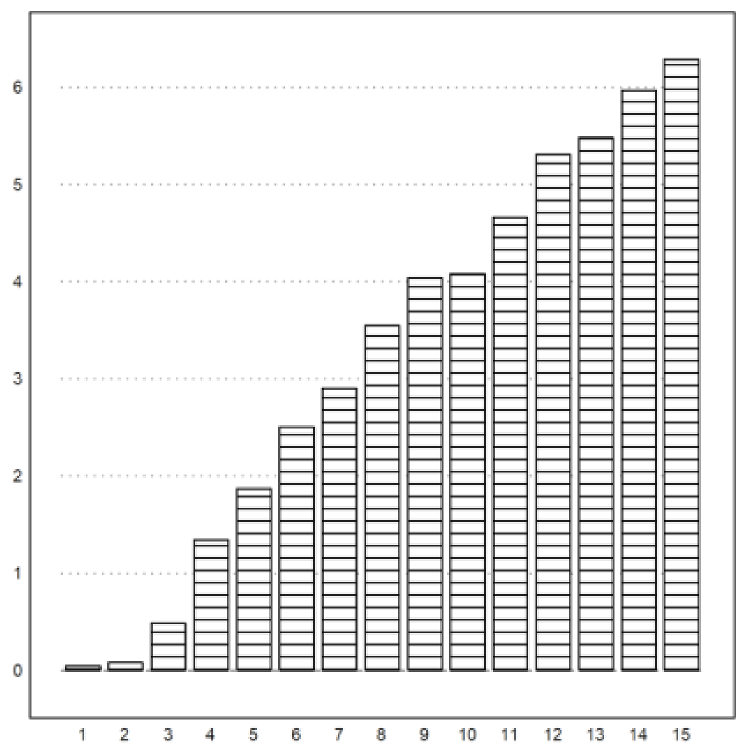
Diagram Batang

Diagram batang adalah representasi visual dari data yang menggunakan balok atau kolom vertikal untuk mewakili kategori, nilai atau variabel tertentu. Setiap kolom yang ada pada diagram batang memiliki frekuensi atau jumlah dalam kategori tersebut.
Contoh:
Kita akan membuat diagram batang secara random.

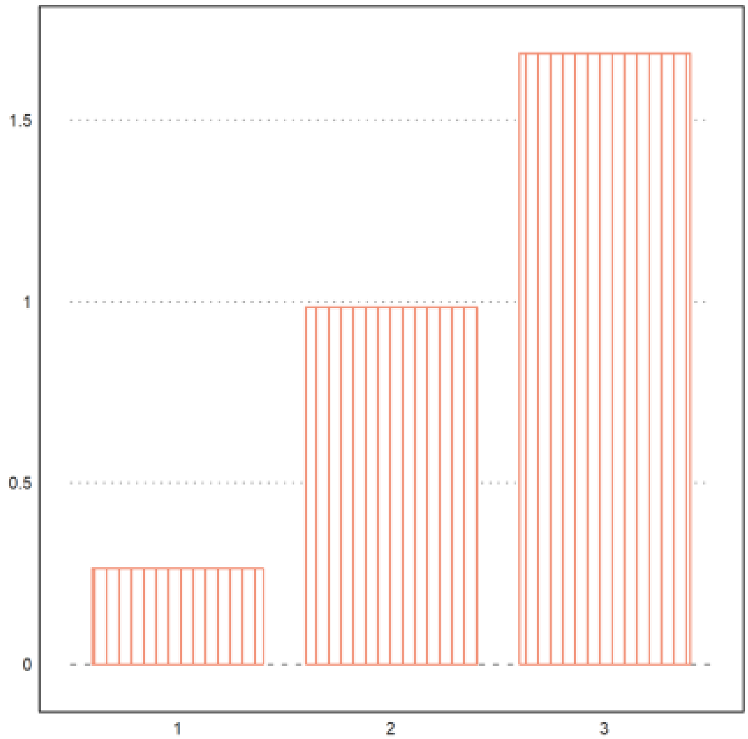
```
>columnsplot(cumsum(random(6)),style="/",color=red):
```




```
>columnsplot(cumsum(random(15)),style="-",color=black):
```

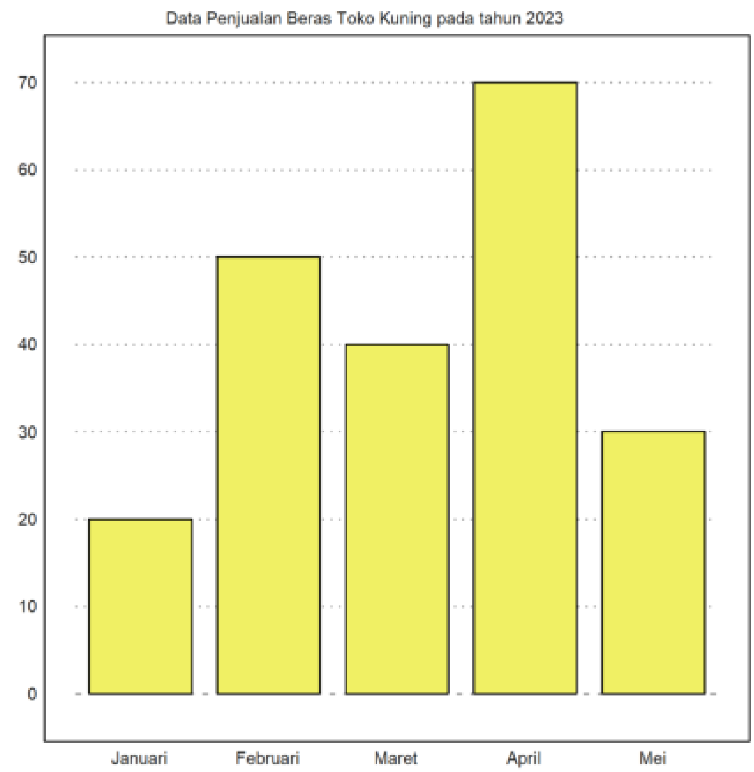


```
>columnsplot(cumsum(random(3)),style="|",color=orange):
```



Selanjutnya kita akan mencoba membuat diagram batang penjualan yang menggunakan variabel.

```
>months=["Januari","Februari","Maret","April","Mei"];
>values=[20,50,40,70,30];
>columnsplot(values,lab=months,color=yellow);
>title("Data Penjualan Beras Toko Kuning pada tahun 2023"):
```



Perintah "columnsplot(values,lab=months,color=yellow);" merupakan sintaks untuk membuat diagram batang dengan menggunakan nilai dari variabel "values", label bulan dari variabel "months", dan warna kuning Dari diagram batang tersebut kita bisa mengetahui data penjualan toko kuning selama lima bulan pada tahun 2023 yaitu, pada bulan Januari, Februari, Maret , April, Mei. Januari terjual 20 ton beras, Februari terjual 50 ton beras, Maret terjual 40 ton beras, April terjual 70 ton beras, dan Mei terjual 30 ton beras.

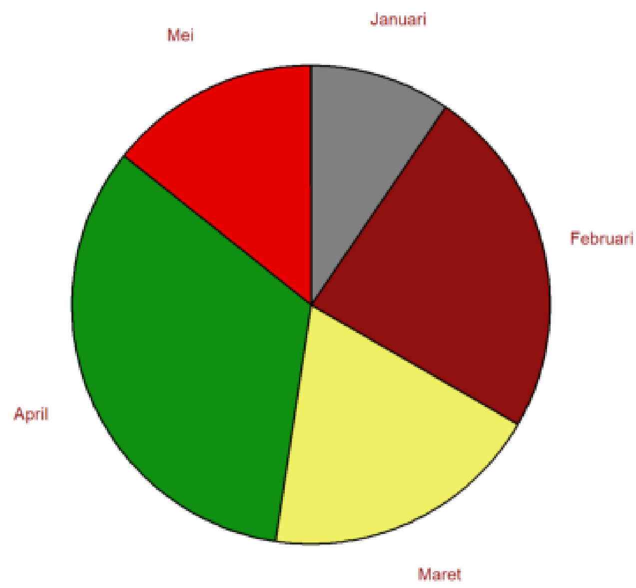
Diagram Lingkaran

Diagram lingkaran merupakan penyajian statistik data tunggal dalam bentuk lingkaran yang dibagi menjadi beberapa juring atau sektor yang menggambarkan banyak frekuensi untuk setiap data.Diagram lingkaran tidak menampilkan informasi frekuensi dari masing-masing data secara detail.

```
>CP:=[rgb(0.5,0.5,0.5),red,yellow,green,rgb(0.9,0,0)]
```

```
[5.87532e+07,   2,   15,   3,   6.54049e+07]
```

```
>i=[1,2,3,4,5]; piechart(values[i],color=CP[i],lab=months[i]):
```

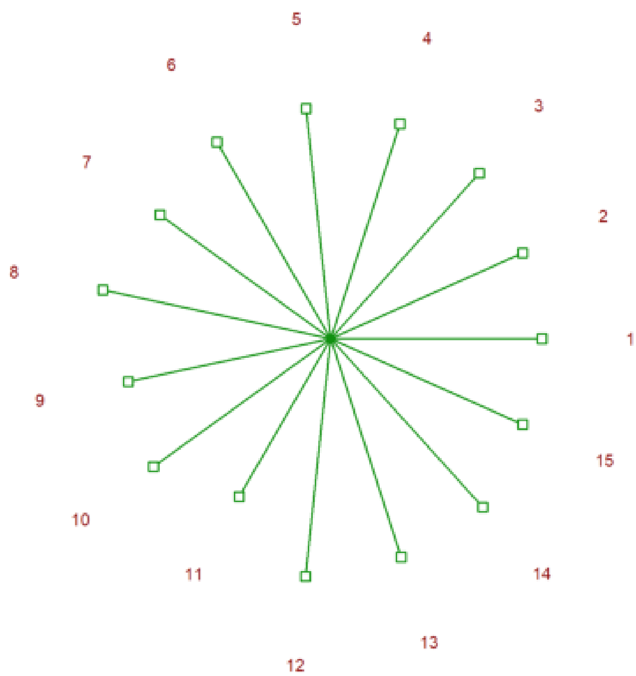


RGB adalah singkatan dari Red, Green, and Blue, dan setiap parameter mendefinisikan intensitas warna dengan nilai antara 0 dan 1. Warna pertama dalam daftar adalah warna abu-abu dengan jumlah merah, hijau, dan biru yang sama. Warna kedua merah, ketiga kuning, dan keempat hijau. Warna terakhir adalah warna merah dengan lebih banyak merah daripada hijau atau biru.

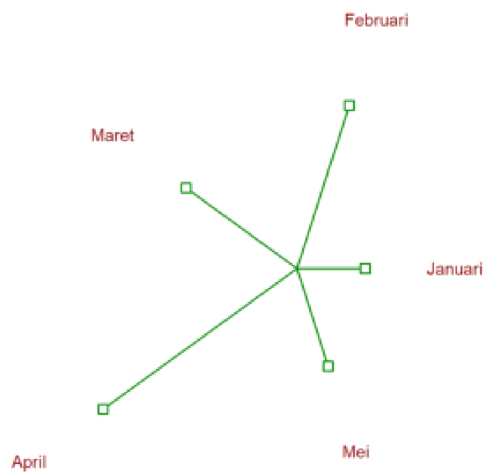
Diagram Bintang

Diagram bintang, terkadang disebut diagram radar atau diagram web, adalah metode perangkat grafis yang digunakan untuk menampilkan data multivariat. Multivariat dalam pengertian ini mengacu pada memiliki banyak karakteristik untuk diamati. Variabelnya juga harus berupa nilai yang berkisar.
Diagram bintang terdiri dari rangkaian jari-jari bersudut sama, yang disebut jari-jari, dengan masing-masing jari mewakili salah satu variabel. Panjang jari-jari data sebanding dengan besaran variabel pada titik data relatif terhadap besaran maksimum variabel di seluruh titik data.

```
>starplot(normal(1,15)+16,lab=1:15,>rays):
```




```
>starplot(values,lab=months,>rays):
```

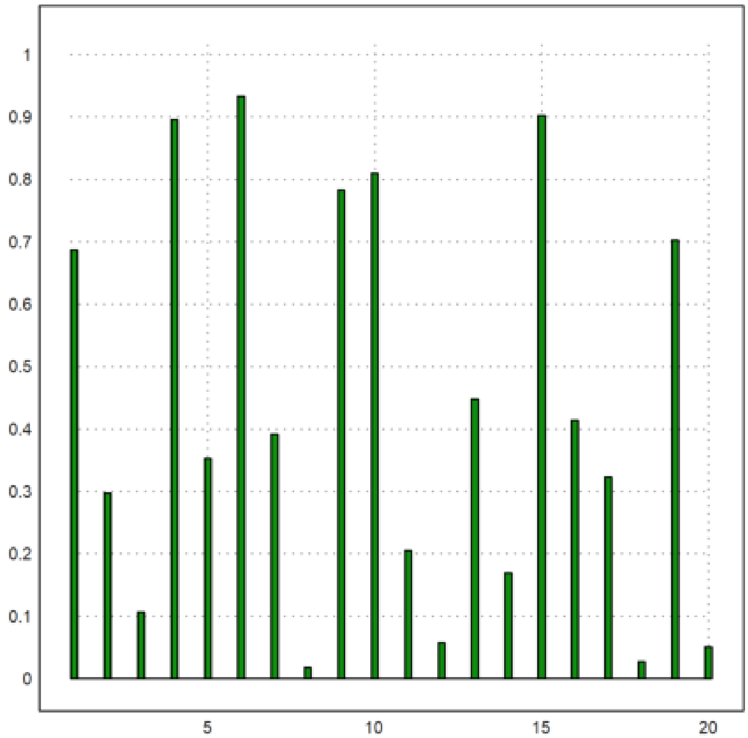


Syntax starplot(values,lab=months,rays) adalah perintah untuk membuat grafik bintang (star plot) dengan menggunakan nilai-nilai yang diberikan dalam vektor values, label sumbu yang diberikan dalam vektor months, dan jumlah rays yang menentukan jumlah garis radial yang digunakan dalam grafik

Diagram Impuls

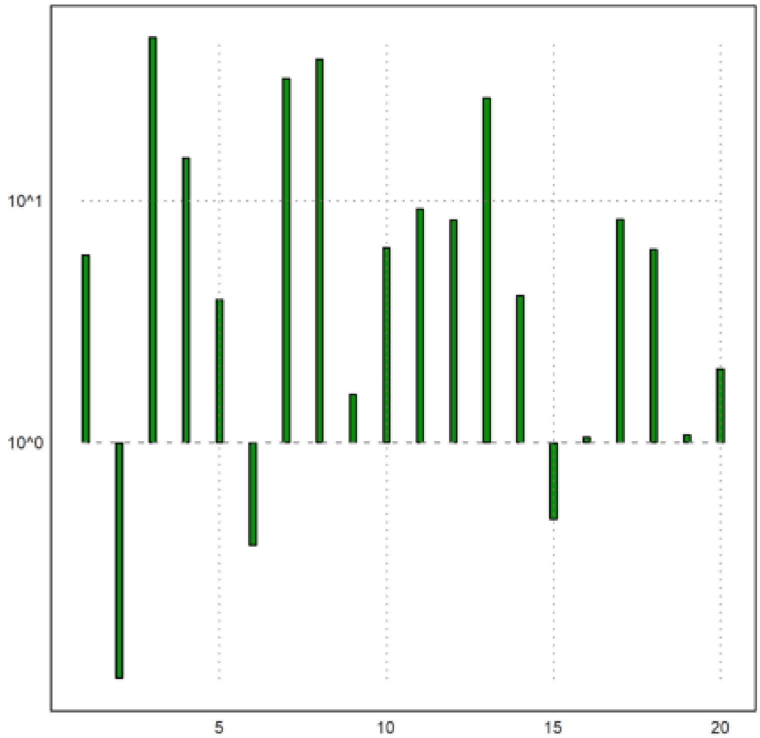
Impuls (impulse) adalah perubahan momentum. Contohnya adalah sebuah bola bermassa yang tengah ditendang, bola menggelinding yang dihentikan, bola jatuh yang memantul, mobil yang menabrak tembok, telur jatuh yang pecah.
Berikut adalah plot impuls dari data acak 1 sampai 20, terdistribusi secara merata di [0,1].

```
>plot2d(makeimpulse(1:20,random(1,20)),>bar):
```



Tetapi untuk data yang terdistribusi secara eksponensial, kita mungkin memerlukan plot logaritmik.

```
> logimpulseplot(1:20,-log(random(1,20))*10):
```

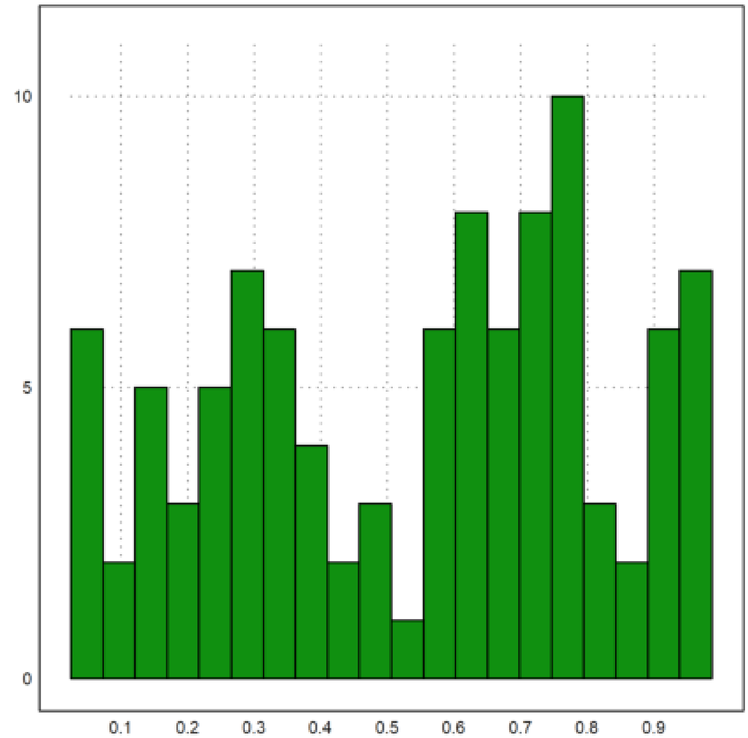


Jadi gambar grafiknya terlihat naik turun (mengalami perubahan).

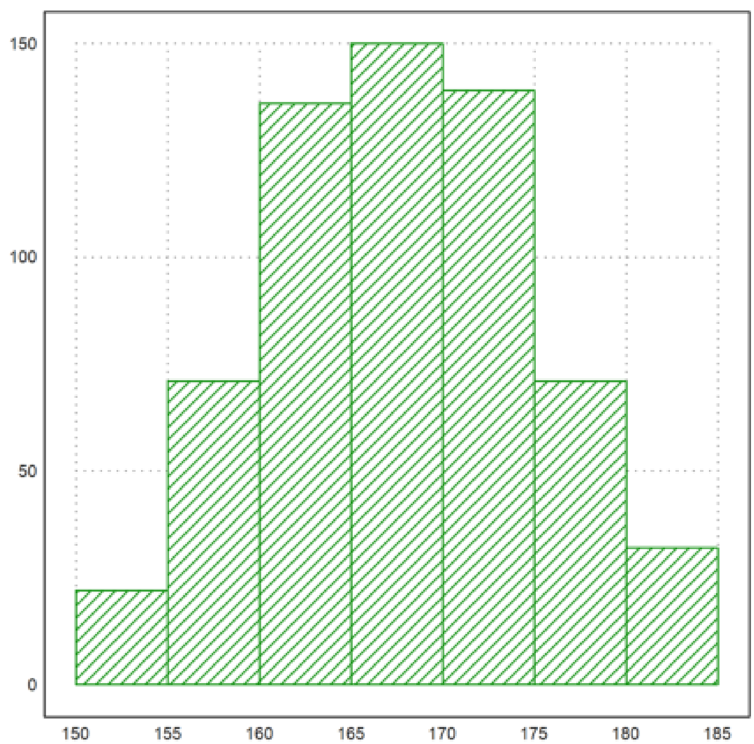
Histogram

Histogram adalah representasi grafis (diagram) yang mengatur dan menampilkan frekuensi data sampel pada rentang tertentu. Frekuensi data yang ada pada masing-masing kelas direpresentasikan dengan bentuk grafik diagram batang atau kolom.

```
>aspect(1); plot2d(random(100),>histogram):
```




```
>r=150:5:185; v=[22,71,136,150,139,71,32];
>plot2d(r,v,a=150,b=185,c=0,d=150,bar=1,style="/"):
```

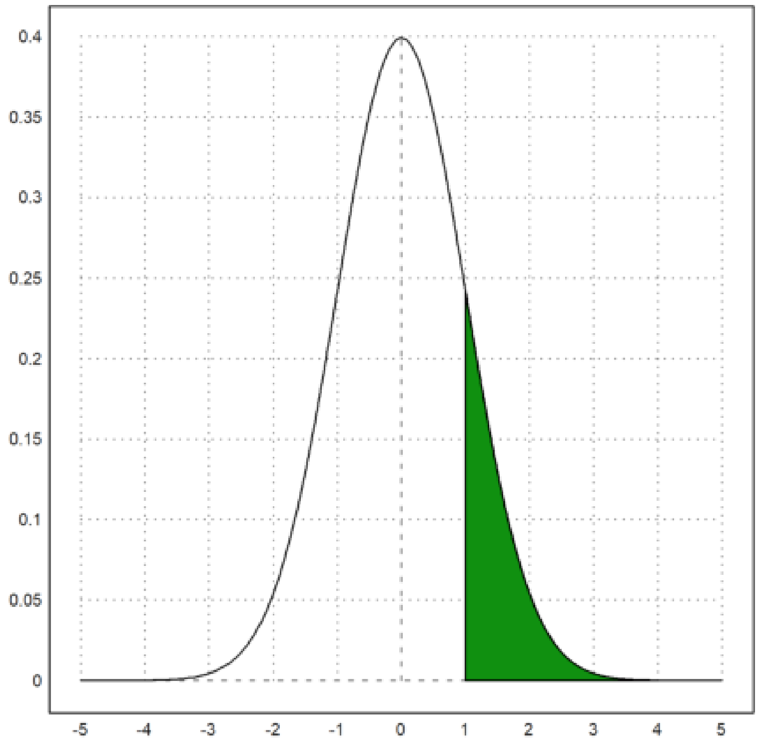


Pola "r=150:5:185" berarti bahwa nilai r dimulai dari 150, kemudian bertambah 5 setiap kali, dan berakhir saat mencapai atau melebihi 185. Dengan pola ini, kita dapat menentukan nilai-nilai r yang sesuai.
Dari data yang diperoleh dapat diketahui bahwa dari rentang kelas 150-155 memiliki frekuensi 22, rentang kelas 155-160 memiliki frekuensi 71, dan seterusnya.

Kurva Fungsi Kerapatan Probabilitas

Secara teoritis kurva probabilitas populasi diwakili oleh poligon frekuensi relatif yang dimuluskan (variabel acak kontiniu diperlakukan seperti variabel acak diskrit yang rapat).Karena itu fungsi dari variabel acak kontiniu merupakan fungsi kepadatan probabilitas (probability density function – pdf). Pdf menggambarkan besarnya probabilitas per unit interval nilai variabel acaknya.

```
>plot2d("qnormal(x,0,1)",-5,5);  ...
>plot2d("qnormal(x,0,1)",a=1,b=4,>add,>filled):
```



Perintah "plot2d("qnormal(x,0,1)",-5,5)" digunakan untuk membuat plot dari distribusi normal dengan mean 0 dan standard deviation 1 di rentang -5 hingga 5
Probabilitas variabel acak x yang terletak antara 1 dan 4 memenuhi
P(1<X<4)= luas daerah hijau

Kurva Fungsi Distribusi Kumulatif

Cumulative Distribution Function (CDF) atau fungsi distribusi kumulatif adalah fungsi matematika yang digunakan untuk menghitung probabilitas variabel acak diskrit atau kontinu. CDF memberikan probabilitas bahwa variabel acak akan menghasilkan nilai kurang dari atau sama dengan nilai tertentu. Dalam hal ini, CDF dapat digunakan untuk menghitung probabilitas kumulatif dari variabel acak.
Berikut merupakan contoh kurva fungsi distribusi kumulatif kontinu:

```
>splot2d("normaldis",-3,5):
```

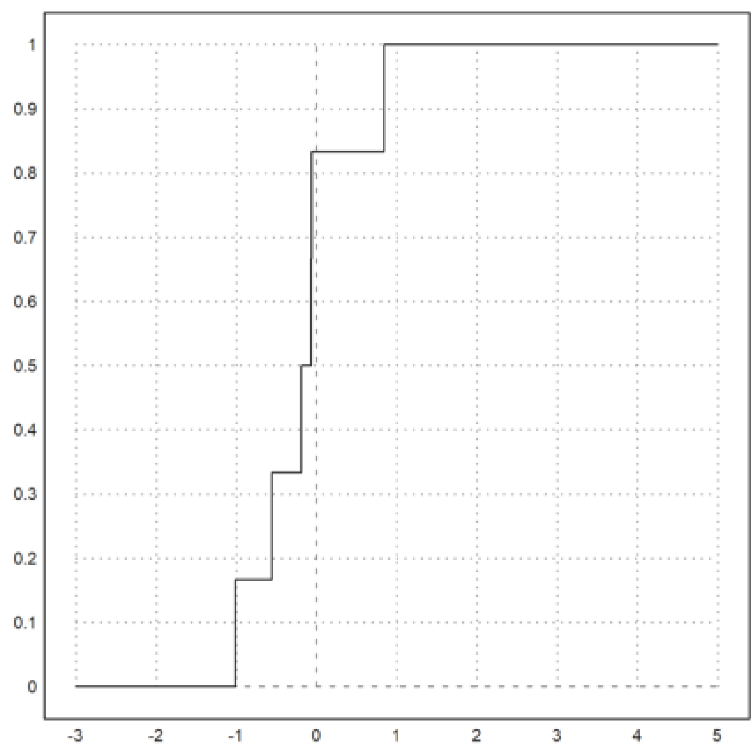
```
Function splot2d not found.
Try list ... to find functions!
Error in:
splot2d("normaldis",-3,5): ...
                              ^
```

Dapat kita lihat dalam kurva fungsi distribusi kumulatif kontinu terdiri atas tiga bagian yaitu:
1. Bernilai 0 untuk x di bawah minimal dari daerah rentang.
2. Merupakan fungsi monoton naik pada daerah rentang.
3. Mempunyai nilai konstan 1 di atas batas maksimum daerah rentangnya.
Adapun contoh kurva fungsi distribusi kumulatif diskrit sebagai berikut.

```
>x=normal(1,6);
```

Baris kode tersebut akan menghasilkan suatu nilai acak dari distribusi normal dengan mean 1 dan deviasi standar 6, dan nilai tersebut disimpan dalam variabel x. Variabel x kemudian dapat digunakan dalam perhitungan atau analisis selanjutnya
Fungsi empdist(x,vs) membutuhkan array nilai yang diurutkan. Jadi kita harus mengurutkan x sebelum kita dapat menggunakannya.

```
>xs=sort(x);
>plot2d("empdist",-3,5;xs):
```



Grafik fungsi distribusi kumulatif peubah acak diskrit merupakan fungsi tangga naik dengan nilai terendah 0 dan nilai tertinggi 1.

Sub topik 7 : Menampilkan Tabel Frame Data

Cakupan Materi 1) Diagram Titik
2) Diagram Garis
3) Kurva Regresi
4) Menampilkan Tabel Data Frame

Diagram Titik Diagram titik atau bisa disebut Scatter Plot adalah tipe grafik yang digunakan untuk menampilkan nilai-nilai dua variabel pada sumbu horizontal dan vertikal. Setiap titik dalam diagram mewakili satu observasi atau data point. Diagram titik sangat berguna untuk menemukan pola atau hubungan antara dua variabel, serta untuk mengevaluasi distribusi data.
Dalam scatter plot, sumbu horizontal umumnya digunakan untuk variabel independen, sementara sumbu vertikal digunakan untuk variabel dependen. Dengan melihat pola penyebaran titik-titik, kita akan mendapatkan wawasan tentang apakah ada korelasi antara dua variabel dan jenis korelasi apa yang mungkin ada (positif, negatif, atau tidak ada korelasi).

```
>x=normal(1,150); plot2d(x,x+rotright(x),>points,style=".."):
```

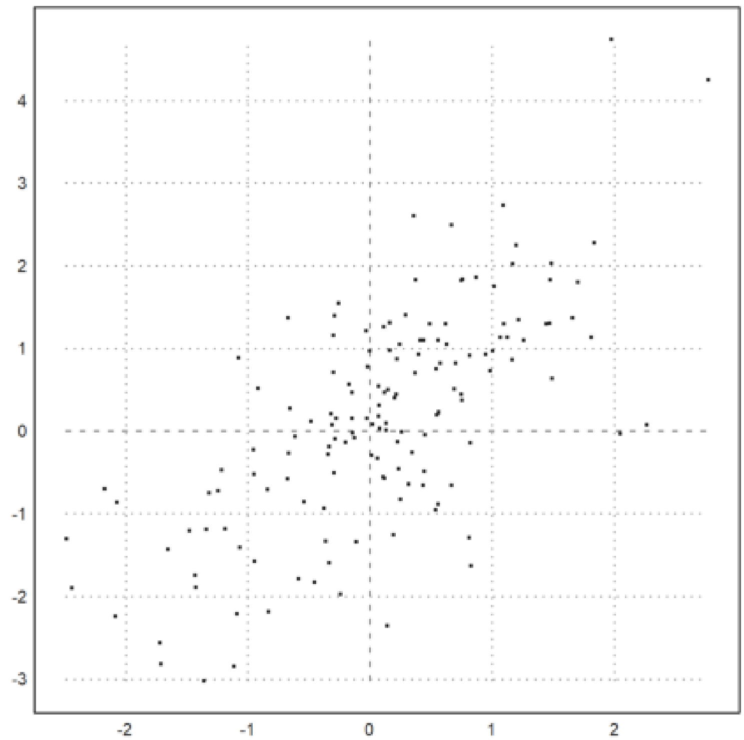


Diagram titik di atas memvisualisasikan data yang dihasilkan oleh fungsi normal dengan parameter mean 1 dan standar deviasi 150. Diagram di atas juga menunjukkan adanya korelasi yang positif karena pergeseran data tersebut ke kanan.
Berikut adalah penjelasan dari setiap perintah dalam sintaks tersebut:
- x=normal(1,150) menghasilkan data dari distribusi normal dengan mean 1 dan standar deviasi 150.
- plot2d(x,x+rotright(x),>points,style="..") digunakan untuk membuat plot 2D dari data x dan x+rotright(x) sebagai sumbu x dan y. >points digunakan untuk menunjukkan bahwa plot yang dihasilkan berupa titik-titik, dan style=".." menentukan gaya dari titik-titik tersebut.

```
>plot2d(normal(1500),normal(1500),>points,grid=6,style=".."):
```

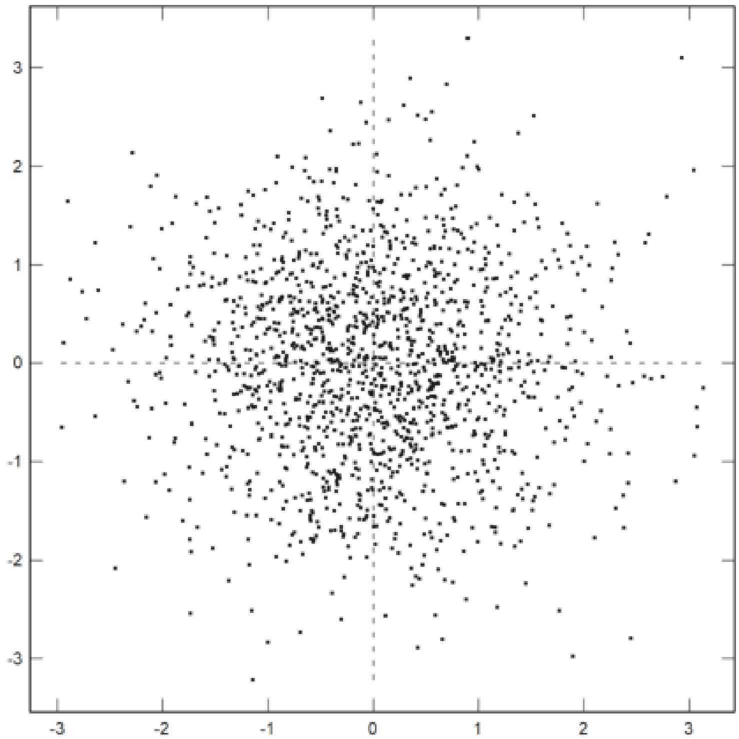


Diagram titik di atas memvisualisasikan distribusi dari dua set data yang dihasilkan oleh distribusi normal.
Berikut adalah penjelasan dari setiap perintah dalam sintaks tersebut:
plot2d(normal(1500),normal(1500),>points,grid=6,style="..") digunakan untuk membuat plot 2D dari dua distribusi normal yang menghasilkan masing-masing 1500 data. >points menunjukkan bahwa plot yang dihasilkan berupa titik-titik, grid=6 menentukan ukuran grid, dan style=".." menentukan gaya dari titik-titik tersebut.

```
>{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;
```

Tabel di atas memvisualisasikan data yang berisi mengenai survei anak, jenis kelamin mereka, usia mereka, usia orang tua mereka dan jumlah saudara kandung mereka.
Berikut adalah penjelasan dari setiap perintah dalam sintaks tersebut:
- {MS,hd}:=readtable("table1.dat",tok2:=["m","f"]) digunakan untuk membaca data dari file "table1.dat" dan menyimpannya dalam variabel MS dan hd. tok2:=["m","f"] menunjukkan bahwa data dalam file tersebut memiliki format "m" dan "f".
- writetable(MS,labc=hd,tok2:=["m","f"]) digunakan untuk menulis kembali data ke file dengan menggunakan label hd sebagai nama kolom dan format data "m" dan "f".

```
>scatterplots(tablecol(MS,3:5),hd[3:5]):
```

```
Variable or function MS not found.
Error in:
scatterplots(tablecol(MS,3:5),hd[3:5]): ...
                        ^
```

Diagram titik di atas memvisualisasikan hubungan antara tiga variabel dalam tabel data "MS", yaitu usia anak, ibu, dan ayah.
Berikut adalah penjelasan dari setiap perintah dalam sintaks tersebut:
- scatterplots(tablecol(MS,3:5),hd[3:5]) digunakan untuk membuat scatter plot (diagram titik) dari tiga kolom data dalam tabel MS dengan menggunakan label hd sebagai nama kolom.
- tablecol(MS,3:5) digunakan untuk memilih tiga kolom data dari tabel MS.
- hd[3:5] digunakan untuk memilih tiga label kolom dari hd.

Contoh Soal

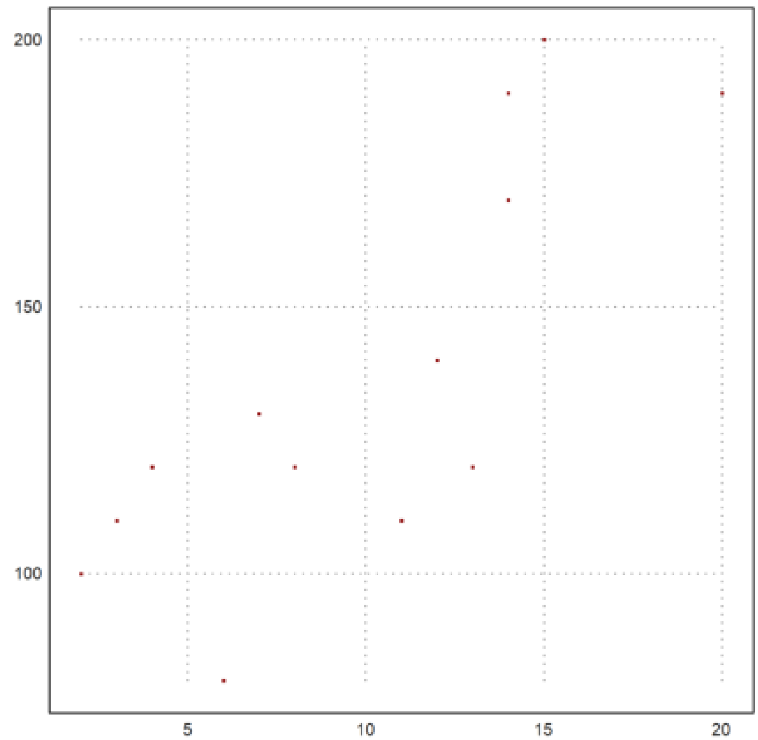
Tabel berikut ini memberikan informasi mengenai kandungan gula (gram) dan jumlah kalori dalam satu sajian dari 13 sampel merek sereal.

```
>DATA:=[ ...
>1,4,120; ...
>2,15,200; ...
>3,12,140; ...
>4,11,110; ...
>5,8,120; ...
>6,6,80; ...
>7,14,170; ...
>8,2,100; ...
>9,7,130; ...
>10,14,190; ...
>11,20,190; ...
>12,3,110; ...
>13,13,120];
>VAR:= ["MERK", "GULA(gram)", "KALORI"];
>writetable(DATA,labc=VAR);
```

```
 MERK  GULA(gram)    KALORI
    1           4       120
    2          15       200
    3          12       140
    4          11       110
    5           8       120
    6           6        80
    7          14       170
    8           2       100
    9           7       130
   10          14       190
   11          20       190
   12           3       110
   13          13       120
```

a. Gambarkan diagram titik atau scatter plot dari data di atas.
b. Bagaimana pola penyebaran titik-titik yang telah digambar pada diagram di atas?
c. Kesimpulan seperti apa yang dapat kalian ambil mengenai hubungan antara gula (gram) dan jumlah kalori?
Penyelesaian:

```
>GULA:=[4,15,12,11,8,6,14,2,7,14,20,3,13];
>KALORI:=[120,200,140,110,120,80,170,100,130,190,190,110,120];
>plot2d(GULA,KALORI,>points,color=red,style=".."):
```



a. Diagram titik atau scatter plot di atas merupakan hasil visualisasi dari tabel data yang telah disebutkan sebelumnya.
b. Pada diagram di atas dapat terlihat bahwa pola penyebaran titiknya mempunyai kecenderungan semakin naik ke atas jika dilihat dari kiri bawah ke kanan atas.
c. Kesimpulan yang dapat diambil yaitu semakin tinggi kandungan gula maka semakin tinggi jumlah kalorinya.

Diagram Garis Diagram garis atau bisa disebut line chart adalah jenis grafik statistik yang digunakan untuk menunjukkan perubahan atau tren sepanjang waktu atau variabel independen lainnya. Ini adalah salah satu cara yang paling umum digunakan untuk memvisualisasikan data berurutan dalam statistika.
Dalam diagram garis, data diplotkan sebagai titik-titik dan kemudian dihubungkan dengan garis lurus. Ini membantu untuk melihat perubahan atau tren dari satu titik ke titik berikutnya dengan jelas. Diagram garis sangat berguna untuk menyoroti pola, tren, atau fluktuasi dalam data sepanjang periode waktu tertentu.
Contoh umum penggunaan diagram garis melibatkan waktu di sumbu horizontal (x) dan nilai atau frekuensi di sumbu vertikal (y). Misalnya, diagram garis dapat digunakan untuk menunjukkan perubahan suhu sepanjang waktu, pertumbuhan populasi, atau kinerja keuangan perusahaan sepanjang beberapa kuartal.
Untuk memetakan data, kita mencoba hasil pemilu Jerman sejak tahun 1990, 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];
```

Sintaks tersebut digunakan untuk membuat matriks BW yang berisi data-data tahun dan jumlah kursi untuk beberapa partai politik pada setiap tahun.

```
>P:=["CDU/CSU","SPD","FDP","Gr","Li"];
```

Sintaks tersebut digunakan untuk membuat array P yang berisi label-label untuk setiap kolom pada matriks BW.

```
>BT:=BW[,3:7]; BT:=BT/sum(BT); YT:=BW[,1]';
```

Berikut adalah penjelasan dari setiap perintah dalam sintaks tersebut:
- BT:=BW[,3:7]; BT:=BT/sum(BT) digunakan untuk mengambil kolom ke-3 hingga ke-7 dari matriks BW, kemudian membagi setiap nilai dalam matriks tersebut dengan jumlah total nilai dalam matriks tersebut. Kolom 3 sampai 7 adalah jumlah kursi masing-masing partai, dan kolom 2 adalah jumlah kursi seluruhnya. Sedangkan kolom 1 adalah tahun pemilihan.
- YT:=BW[,1]' digunakan untuk mengambil transpose dari kolom pertama matriks BW dan menyimpannya dalam variabel YT.

```
>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
```

Sintaks tersebut digunakan untuk menulis matriks BT yang telah dikalikan dengan 100 ke dalam sebuah file. Parameter wc=6 menunjukkan lebar atau sekat kolom, dc=0 menunjukkan jumlah digit di belakang koma, >fixed menunjukkan bahwa jumlah digit di belakang koma tetap, labc=P menunjukkan label untuk kolom, dan labr=YT menunjukkan label untuk baris.

```
>BT1:=(BT.[1;1;0;0;0])'*100
```

```
 [84.29,  81.25,  81.1659,  82.7529,  72.9642,  61.8971,  79.8732]
```

Sintaks di atas digunakan untuk mengalikan baris pertama dari matriks BT dengan 100 dan menyimpan hasilnya dalam variabel BT1. Angka-angka dalam tanda kurung siku menunjukkan indeks baris yang ingin diambil dari matriks BT (pada sintaks tersebut indeks baris yang ingin diambil yaitu baris pertama), sedangkan tanda kutip di sebelah kanan menunjukkan bahwa kita ingin mengalikan baris tersebut dengan 100.

```
>statplot(YT,BT1,"b"):
```

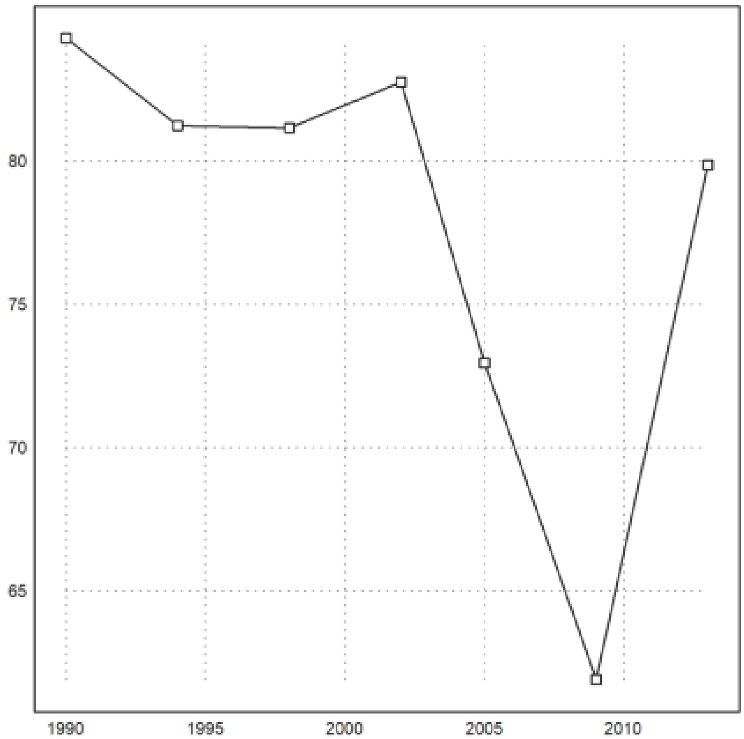


Diagram di atas memvisualisasikan persentase perolehan jumlah kursi setiap partai di setiap tahunnya, yaitu dari tahun 1990-2013.
Tipe plotplot yang tersedia adalah sebagai berikut:
- 'p' : plot titik
- 'l' : plot garis
- 'b' : keduanya (titik dan garis)
- 'h' : alur histogram
- 's' : plot permukaan

Contoh Soal

Perhatikan tabel di bawah ini!

```
>BARANG:=[ ...
>2015,13000,10000,2700,10700; ...
>2020,15000,12500,4000,12300];
>VRB := ["TAHUN","SUSU(kaleng)","GULA(kg)","JAGUNG(kg)","BERAS(kg)"];
>writetable(BARANG,labc=VRB);
```

```
TAHUN  SUSU(kaleng)   GULA(kg)  JAGUNG(kg)  BERAS(kg)
 2015         13000      10000        2700      10700
 2020         15000      12500        4000      12300
```

Hitunglah persentase kenaikan harga dari tahun 2015 hingga tahun 2020 (jika penghitungan indeks harga menggunakan metode agregatif sedehana) dan gambarkan grafik kenaikan harganya menggunakan diagram garis!
Penyelesaian:

```
>HARGA:=BARANG[,2:5]; TAHUN:=BARANG[,1]';
>Po:=(13000+10000+2700+10700) // total harga barang di tahun 2015
```

```
36400
```

```
>Pn:=(15000+12500+4000+12300) // total harga barang di tahun 2020
```

```
43800
```

```
>Persentase:=(Pn/Po)*100
```

```
120.32967033
```

```
>Kenaikan:=(120.3-100)
```

```
20.3
```

Berdasarkan penghitungan tersebut, dapat disimpulkan bahwa dari tahun 2015 ke tahun 2020 terdapat kenaikan harga sebesar 20,3%.

```
>P1:=[36400,43800];
>statplot(TAHUN,P1,"b"):
```

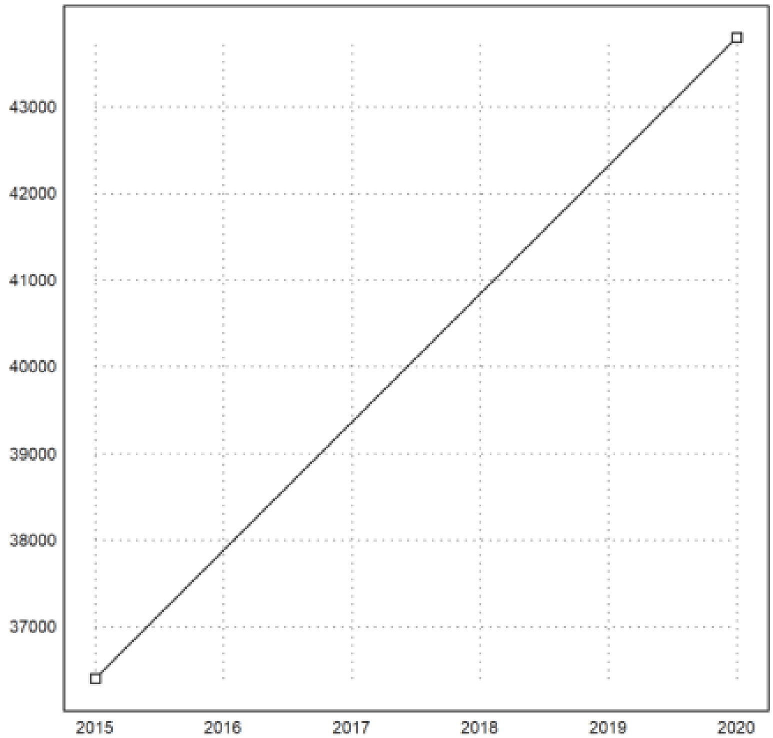


Diagram garis di atas memvisualisasikan kenaikan harga barang dari tahun 2015 hingga tahun 2020.

Kurva Regresi Kurva regresi adalah representasi grafis dari model regresi yang digunakan untuk memodelkan hubungan antara variabel independen (X) dan variabel dependen (Y). Persamaan regresi dapat digunakan untuk memprediksi atau mengestimasi nilai dari variabel tak bebas berdasarkan informasi dari variabel bebas. Persamaan regresi linear merupakan suatu persamaan yang berupa garis lurus, sedangkan persamaan regresi nonlinear bukan merupakan persamaan garis lurus.

Persamaan regresi linier umumnya ditulis sebagai Y=a+bX, di mana:
Y = garis regresi/ variable response
a = konstanta (intercept), perpotongan dengan sumbu-y
b = konstanta regresi (slope), kemiringan
X = variabel bebas/ predictor

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
```

Berikut adalah penjelasan dari setiap perintah dalam sintaks tersebut:
- x=1:10; y=[2,3,1,5,6,3,7,8,9,8];: Perintah ini digunakan untuk membuat vektor x yang berisi bilangan bulat dari 1 hingga 10, dan vektor y yang berisi nilai-nilai acak.
- writetable(x'|y',labc=["x","y"]): Perintah ini digunakan untuk membuat tabel dua kolom dari dua vektor x dan y. Parameter x'|y' menunjukkan bahwa vektor x dan y akan disusun secara vertikal, sedangkan labc=["x","y"] mengatur label kolom tabel.
Kami ingin membandingkan kecocokan yang tidak berbobot dan berbobot. Pertama koefisien kecocokan linier.

```
>p=polyfit(x,y,1)
```

```
[0.733333,  0.812121]
```

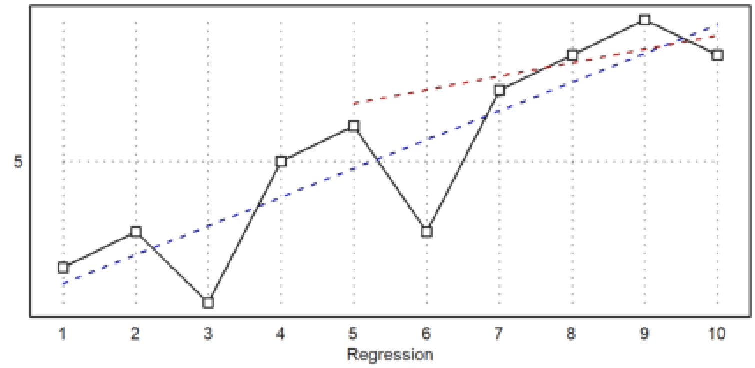
Sintaks di atas digunakan untuk melakukan fitting kurva polinomial orde satu pada data yang diberikan. Parameter x dan y adalah data yang akan di-fit, sedangkan 1 menunjukkan orde polinomial yang digunakan. Hasil fitting akan disimpan dalam variabel p.
Sekarang koefisien dengan bobot yang menekankan nilai terakhir.

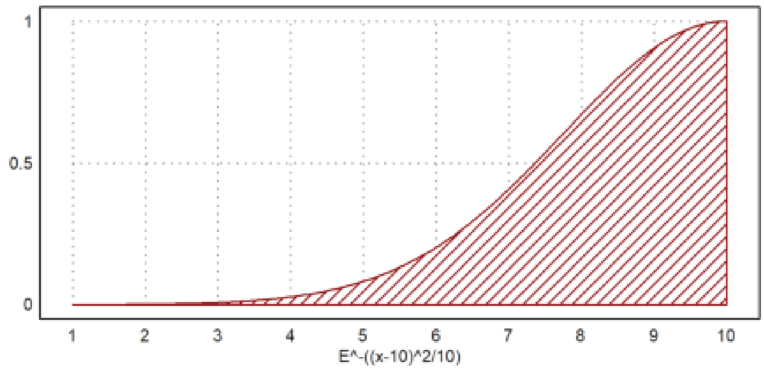
```
>w &= "exp(-(x-10)^2/10)"; pw=polyfit(x,y,1,w=w(x))
```

```
[4.71566,  0.38319]
```

Berikut adalah penjelasan dari setiap perintah dalam sintaks tersebut:
- w &= "exp(-(x-10)^2/10)": digunakan untuk mendefinisikan fungsi berat (weight function) w sebagai exp(-(x-10)^2/10). Fungsi berat digunakan dalam fitting polinomial untuk memberikan bobot pada setiap titik data, sehingga titik-titik data yang lebih dekat dengan nilai 10 akan diberi bobot yang lebih tinggi.
- pw=polyfit(x,y,1,w=w(x)): digunakan untuk melakukan fitting polinomial pada data yang diberikan dengan menggunakan fungsi berat w. Fungsi polyfit digunakan untuk menemukan koefisien polinomial yang sesuai dengan data. Dalam kasus ini, fitting dilakukan dengan polinomial derajat 1, dengan menggunakan fungsi berat yang telah didefinisikan sebelumnya.
Kami memasukkan semuanya ke dalam satu plot untuk titik 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):
```





Berikut adalah penjelasan dari setiap perintah dalam sintaks tersebut:
- figure(2,1); digunakan untuk mengatur tampilan grafik dengan berbeda nomor tampilan (handle).
- statplot(x,y,"b",xl="Regression"): digunakan untuk menampilkan data (x,y) dengan label nama sumbu-x nya yaitu "Regression".
- plot2d("evalpoly(x,p)",>add,color=blue,style="–"): fungsi evalpoly digunakan untuk menghitung nilai polinomial pada titik-titik data yang diberikan. Dalam hal ini, evalpoly(x,p) menghitung nilai polinomial pada titik-titik data yang diberikan dari polinomial pertama (p). Hasilnya kemudian digunakan untuk menambahkan plot polinomial pertama ke grafik dengan warna biru dan gaya putus-putus.
- plot2d("evalpoly(x,pw)",5,10,>add,color=red,style="–"): fungsi evalpoly digunakan untuk menghitung nilai polinomial yang diperoleh dari polyfit dengan fungsi berat. Dalam hal ini, evalpoly(x,pw) menghitung nilai polinomial dengan fungsi berat pada titik-titik data dari 5 hingga 10. Hasilnya kemudian digunakan untuk menambahkan plot polinomial dengan fungsi berat ke grafik dengan warna merah dan gaya putus-putus.
- plot2d(w,1,10,>filled,style="/",fillcolor=red,xl=w): sintaks tersebut digunakan untuk membuat plot dari area di bawah kurva polinomial dengan fungsi berat w pada rentang 1 hingga 10, dengan gaya pengisian warna merah dan label sumbu x yang diberi nilai w

---

Menampilkan Data Frame Data frame biasanya merujuk pada struktur data tabular dua dimensi yang digunakan untuk menyimpan dan mengorganisir data. Data frame adalah konsep yang umumnya terkait dengan pemrograman statistik, terutama dalam bahasa seperti R dan Python (menggunakan pustaka pandas).

Di direktori buku catatan ini kita akan menemukan file dengan tabel. Data tersebut merupakan hasil survei. Berikut adalah empat baris pertama file tersebut. Datanya berasal dari buku online Jerman "Einführung in die Statistik mit R" yang dibuat 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");
```

Sintaks tersebut digunakan untuk mencetak isi file "table.dat" pada empat baris pertama.

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");
```

Berikut adalah penjelasan dari setiap perintah dalam sintaks tersebut:
- mf:=["m","f"]: Mendefinisikan variabel mf sebagai array yang berisi dua elemen, yaitu "m" dan "f". Kemungkinan ini digunakan untuk merepresentasikan jenis kelamin, di mana "m" mewakili laki-laki dan "f" mewakili perempuan.
- yn:=["y","n"]: Mendefinisikan variabel yn sebagai array yang berisi dua elemen, yaitu "y" dan "n". Kemungkinan ini digunakan untuk merepresentasikan jawaban ya ("y") dan tidak ("n") dari suatu pertanyaan.
- ev:=strtokens("g vg m b vb"): Mendefinisikan variabel ev sebagai hasil dari pemisahan string "g vg m b vb" berdasarkan spasi. Kemungkinan ini digunakan untuk merepresentasikan kategori tertentu yang terkait dengan data atau survei.

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;
```

Sintaks ini digunakan untuk membaca data dari file "table.dat" ke dalam sebuah tabel. Pada saat membaca file, argumen opsional tok2, tok4, tok5, dan tok7 digunakan untuk menentukan bagaimana data dalam file tersebut akan diinterpretasikan.

```
>load over statistics;
```

Sintaks ini digunakan untuk memuat data atau variabel yang telah disimpan sebelumnya.

Untuk mencetak, kita perlu menentukan set token yang sama. Kami mencetak empat baris pertama saja.

```
>writetable(MT[1:4],labc=hd,wc=6,tok2:=mf,tok4:=yn,tok5:=ev,tok7:=yn);
```

```
MT is not a variable!
Error in:
writetable(MT[1:4],labc=hd,wc=6,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.
Berikut adalah penjelasan dari setiap perintah dalam sintaks tersebut:
- ctok=[2,4,5,7]: digunakan untuk mendefinisikan variabel ctok sebagai array yang berisi angka 2, 4, 5, dan 7. Kemungkinan ini digunakan untuk menentukan kolom-kolom tertentu yang akan dibaca dari file "table.dat".
- {MT,hd,tok}=readtable("table.dat",ctok=ctok): sintaks ini menggunakan fungsi readtable untuk membaca data dari file "table.dat" ke dalam tiga variabel yang berbeda, yaitu MT, hd, dan tok, dengan menggunakan kolom-kolom yang telah ditentukan sebelumnya dalam variabel ctok.

```
>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] ...
     ^
```

Sintaks tersebut digunakan untuk mengakses elemen pertama dari variabel MT yang telah diimpor sebelumnya.
Berikut isi tabel dengan nomor yang belum diterjemahkan.

```
>writetable(MT,wc=5)
```

```
Variable or function MT not found.
Error in:
writetable(MT,wc=5) ...
             ^
```

Untuk kenyamanan, Anda dapat memasukkan keluaran readtable() ke dalam list.

```
>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;
```

Dengan menggunakan kolom token yang sama dan token yang dibaca dari file, kita dapat mencetak tabel. Kita dapat menentukan ctok, tok, dll atau menggunakan tabel daftar

```
>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, melewatkan baris apa pun 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]); ...
                 ^
```

Sintaks ini digunakan untuk mengekstrak kolom ke-5 dan ke-6 dari matriks atau tabel MT. Hasilnya akan disimpan dalam variabel c dan i secara berturut-turut. Dengan kata lain, c akan berisi kolom ke-5 dari MT, sedangkan i akan berisi kolom ke-6 dari MT.
Kita bisa 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)
```

```
MT is not a variable!
Error in:
j=[1,5,6]; writetable(MT[i,j],labc=hd[j],ctok=[2],tok=tok) ...
                             ^
```

Berikut adalah maksud dari sintaks tersebut:
- MT[i,j]: Mengambil subset dari tabel MT yang terdiri dari baris-baris yang ditentukan oleh i dan kolom-kolom yang ditentukan oleh j. Ini mungkin digunakan untuk membuat subset dari data yang akan ditulis ke dalam file.
- labc=hd[j]: Menentukan label kolom untuk subset kolom yang dipilih dari MT berdasarkan label kolom dari hd. Ini mungkin digunakan untuk menetapkan label yang sesuai untuk kolom-kolom yang akan ditulis ke dalam file.
- ctok=[2]: Menentukan token tertentu yang akan digunakan saat menulis tabel ke dalam file. Ini mungkin digunakan untuk menetapkan cara khusus untuk menginterpretasikan data saat ditulis ke dalam file.
- tok=tok: Menggunakan token yang telah ditentukan sebelumnya saat menulis tabel ke dalam file. Ini mungkin digunakan untuk memastikan bahwa token yang sama digunakan saat menulis dan membaca tabel.
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 menggunakannya untuk menentukan nilai rata-rata suatu 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.

```
>{xu,count}=getstatistics(tablecol(MT,5)); xu, count,
```

```
Variable or function MT not found.
Error in:
{xu,count}=getstatistics(tablecol(MT,5)); xu, count, ...
                                    ^
```

Variabel xu adalah elemen unik (indeks) dari kolom ke-5 tabel MT, sedangkan variabel count adalah jumlah data dari variabel xu.
Kita bisa mencetak hasilnya di 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 v pada 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 diekstraksi dan diurutkan di kolom ke-5.

```
>writetable(MT1[i],labc=hd,ctok=ctok,tok=tok,wc=7);
```

```
MT1 is not a variable!
Error in:
writetable(MT1[i],labc=hd,ctok=ctok,tok=tok,wc=7); ...
                 ^
```

Untuk statistik selanjutnya, kami ingin menghubungkan dua kolom tabel. Jadi kita ekstrak kolom 2 dan 4 dan urutkan tabelnya.

```
>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])',ctok ...
               ^
```

Dengan getstatistics(), kita juga bisa menghubungkan jumlah 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] ...
                ^
```

Sebuah 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);
```

Sub Topik 8 : Melakukan Perhitungan Untuk Uji Statistika

Perhitungan untuk Uji Statistika Dalam Euler, banyak uji yang diterapkan. Semua uji dalam Euler mengembalikan kesalahan yang diterima jika hipotesis nol ditolak.
Setiap jenis uji statistik memiliki asumsi dan syarat tertentu yang harus dipenuhi sebelum dapat diterapkan pada data. Pemilihan jenis uji statistik yang tepat sangat penting untuk memastikan hasil analisis data yang akurat dan dapat diandalkan.

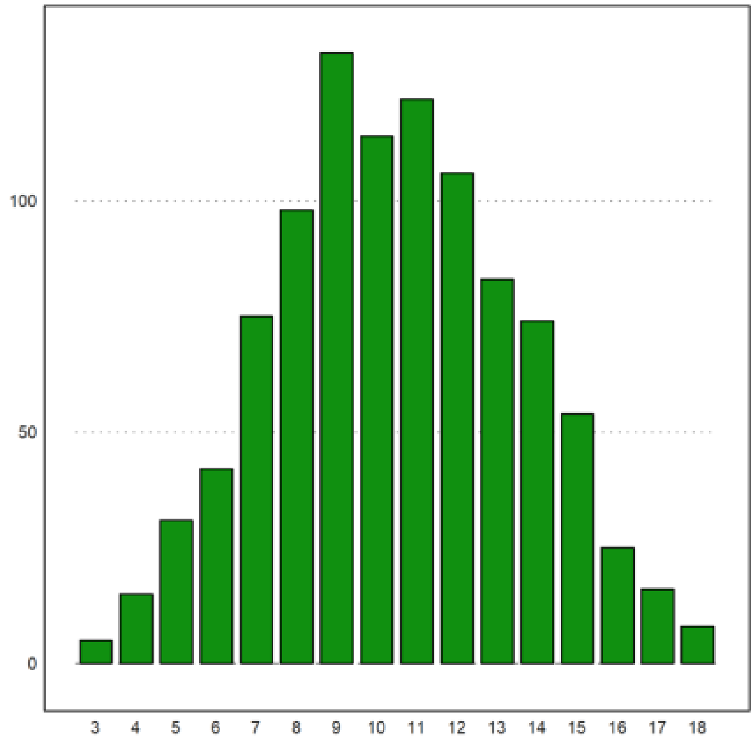
Simulasi Monte Carlo

Simulasi Monte Carlo yang digunakan untuk memecahkan masalah dengan menggunakan sampel acak berulang untuk memperkirakan distribusi dari suatu fenomena. Dalam konteks uji statistika, Monte Carlo dapat digunakan untuk menghitung nilai-nilai yang sulit atau tidak mungkin dihitung secara analitik, seperti p-value dalam uji chi-square.
Euler dapat digunakan untuk mensimulasikan kejadian acak. Berikut ini contoh yang mensimulasikan 1000 kali 3 lemparan dadu, dan ditampilkan distribusi jumlahnya.
Fungsi ini akan menghasilkan 1000 bilangan bulat acak antara 3 dan 6, menjumlahkannya, dan menyimpan hasilnya dalam ds.
Fungsi getmultiplicities(v,x) mengembalikan kelipatan dari nilai-nilai dalam v dalam vektor x.
Dalam hal ini, fs akan berisi kelipatan dari nilai 3 hingga 18 dalam vektor ds.

```
>ds:=sum(intrandom(1000,3,6))'; fs=getmultiplicities(3:18,ds)
```

```
[5,  15,  31,  42,  75,  98,  132,  114,  122,  106,  83,  74,  54,
25,  16,  8]
```

```
>columnsplot(fs,lab=3:18):
```



Untuk menentukan distribusi yang diharapkan tidak begitu mudah. Diperlukan rekursi lanjutan untuk hal ini.

Fungsi berikut menghitung banyaknya cara bilangan k dapat dinyatakan sebagai jumlah dari n bilangan dalam rentang 1 sampai m. Hal berikut 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
```

```
>cw=countways(3:18,3,6)
```

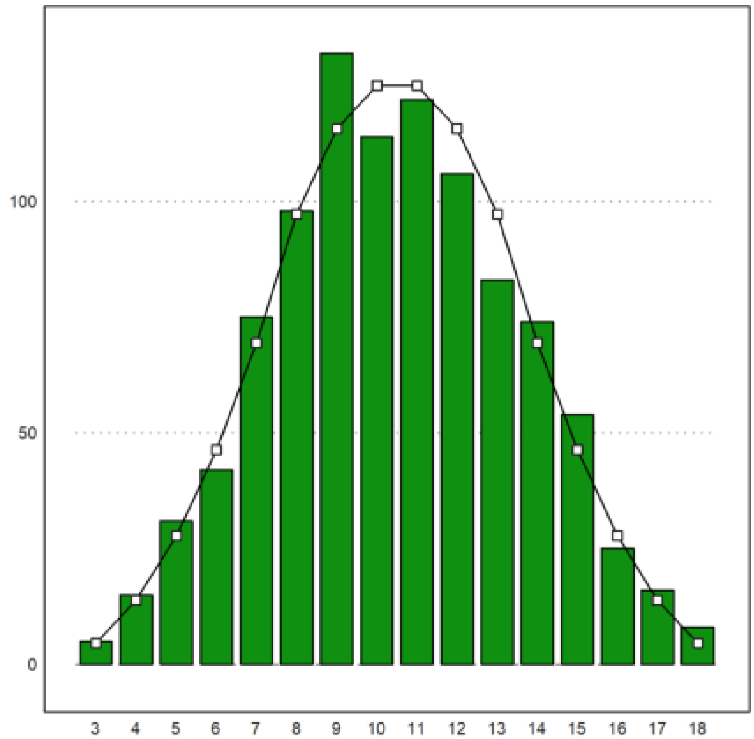
```
 [1,   3,   6,   10,   15,   21,   25,   27,   27,   25,   21,   15,   10,   6,   3,
 1]
```

Nilai yang diharapkan dapat direpresentasikan ke dalam plot.

Fungsi countways menghitung jumlah cara di mana bilangan k dapat direpresentasikan sebagai jumlah dari n bilangan dalam rentang 1 hingga m.

Fungsi ini bekerja secara rekursif dengan cara yang jelas. Jika n=1, maka fungsi mengembalikan k>=1 && k<=m, jika tidak, fungsi akan menghitung jumlah cara dengan melakukan iterasi dari 1 hingga m. Hasilnya adalah jumlah cara di mana bilangan k dapat direpresentasikan sebagai jumlah dari n bilangan dalam rentang 1 hingga m.

```
>plot2d(cw/6^3*1000,>add); plot2d(cw/6^3*1000,>points,>add):
```



Uji Chi Square

Untuk melakukan uji chi^2 (chi-squared) kita menggunakan fungsi chitest(). Fungsi ini digunakan untuk membandingkan frekuensi observasi dengan frekuensi yang diharapkan. Dalam konteks ini, vektor x merepresentasikan frekuensi observasi, sedangkan vektor y merepresentasikan frekuensi yang diharapkan. Misalnya, jika dari sampel 100 orang ditemukan 40 pria, maka vektor observasi x adalah [40,60], dan vektor harapan y mungkin [50,50]. Uji chi^2 digunakan untuk menilai sejauh mana sampel tersebut sesuai dengan frekuensi yang diharapkan.
Sebagai contoh, kami menguji lemparan dadu untuk distribusi baju. Pada 600 lemparan, kami mendapatkan nilai berikut, yang kami masukkan ke dalam uji chi-kuadrat.

```
>chitest([90,103,114,101,103,89],dup(100,6)')
```

```
0.498830517952
```

Fungsi ini digunakan untuk menguji asosiasi antara dua set data. Set pertama [90,103,114,101,103,89] mewakili jumlah yang diamati, dan set kedua dup(100,6) mewakili jumlah yang diharapkan. Hasil dari fungsi chitest adalah probabilitas yang terkait dengan statistik uji chi-kuadrat, yang menunjukkan kemungkinan bahwa data kategoris yang diamati diambil dari distribusi yang diharapkan
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)
```

```
0.497
```

Kesalahan ini terlalu besar. Jadi kita tidak bisa menolak distribusi baju. Ini tidak membuktikan bahwa dadu kami adil. Tapi kita tidak bisa menolak hipotesis kita.
Selanjutnya kita menghasilkan 1000 lemparan dadu menggunakan generator angka acak, dan melakukan tes yang sama.

```
>n=1000; t=random([1,n*6]); chitest(count(t*6,6),dup(n,6)')
```

```
0.594554930686
```

Fungsi random([1,n*6]) digunakan untuk menghasilkan serangkaian bilangan acak antara 1 dan 6 sebanyak n kali. Kemudian, fungsi count(t*6,6) digunakan untuk menghitung berapa kali angka 6 muncul dalam serangkaian bilangan acak tersebut. Fungsi dup(n,6) digunakan untuk menghasilkan serangkaian bilangan 6 sebanyak n kali. Hasil dari fungsi chitest(count(t*6,6),dup(n,6)) akan mengembalikan nilai p, yang dapat digunakan untuk menentukan apakah serangkaian bilangan acak tersebut terdistribusi secara merata atau tidak.

Uji T

Uji-t adalah sebuah metode statistik yang digunakan untuk menguji perbedaan signifikan antara rata-rata dua kelompok yang berbeda.
Misalnya, akan dilakukan pengujian nilai rata-rata dari 100 elemen data menggunakan uji-t.
Mari kita uji nilai rata-rata 100 dengan uji-t.

```
>s=200+normal([1,100])*10; ...
>ttest(mean(s),dev(s),100,200)
```

```
0.161169176307
```

Fungsi ttest() membutuhkan nilai rata-rata, simpangan, jumlah data, dan nilai rata-rata yang akan diuji. Dalam syntax tersebut, normal([1,100]) menghasilkan 100 angka acak yang diambil dari distribusi normal dengan nilai rata-rata 0 dan simpangan baku 1. Kemudian, nilai-nilai tersebut dikalikan dengan 10 dan ditambahkan dengan 200 untuk menghasilkan sampel data dengan nilai rata-rata 200 dan simpangan baku 10.
Sekarang mari kita periksa dua pengukuran untuk mean yang sama. Kami menolak hipotesis bahwa mereka memiliki rata-rata yang sama, jika hasilnya <0,05.

```
>tcomparedata(normal(1,10),normal(1,10))
```

```
0.23564235612
```

Jika kita menambahkan bias ke satu distribusi, kita mendapatkan lebih banyak penolakan. Ulangi simulasi ini beberapa kali untuk melihat efeknya.

```
>tcomparedata(normal(1,10),normal(1,10)+2)
```

```
0.000114481122387
```

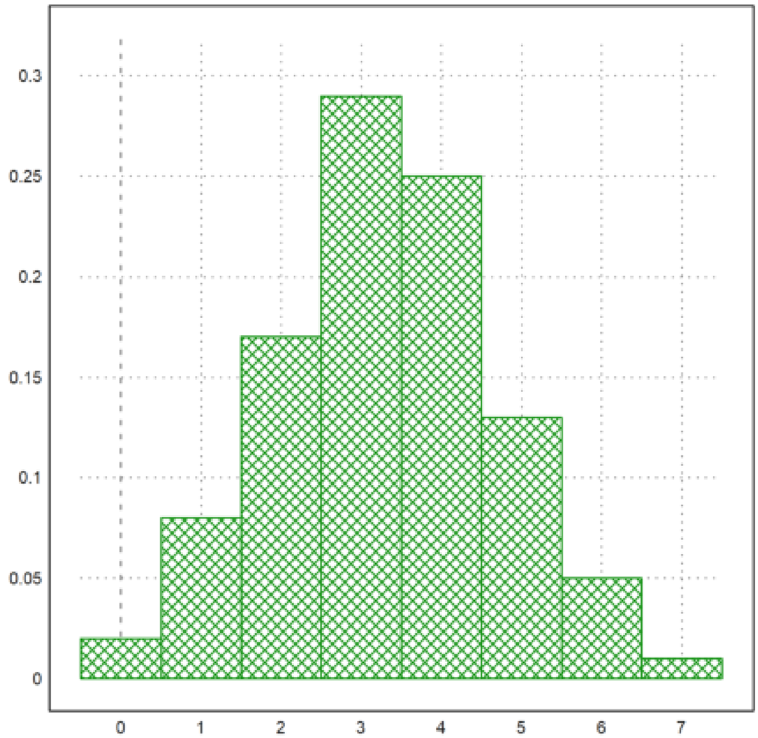
Pada contoh berikutnya, kita menghasilkan 20 lemparan dadu acak sebanyak 100 kali dan menghitung yang ada di dalamnya. Harus ada 20/6=3,3 yang rata-rata.

```
>R=random(100,20); R=sum(R*6<=1)'; mean(R)
```

```
3.31
```

Bandingkan jumlah satu dengan distribusi binomial. Pertama lakukan plot distribusi satuan.

```
>plot2d(R,distribution=max(R)+1,even=1,style="\/"):
```




```
>t=count(R,21);
```

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-kuadrat menolak hipotesis bahwa distribusi kami adalah distribusi binomial, jika hasilnya <0,05.

```
>chitest(t1,b1)
```

```
0.45755878393
```

Uji Independensi

Uji independensi adalah salah satu uji statistik yang digunakan untuk mengetahui apakah ada hubungan antara dua variabel kategorik, dengan kata lain untuk mengetahui independensi antara variabel baris dan kolom. Uji ini berguna untuk mengukur perbedaan pengamatan dan menaksir frekuensi suatu pengamatan dalam kategori tertentu.
Contoh berikut berisi hasil dua kelompok orang (laki-laki dan perempuan, katakanlah) memberikan suara untuk 6 jenis minuman kaleng

```
>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
```

Tes tabel chi^2 dapat melakukan uji independensi suara dari jenis kelamin

```
>tabletest(A)
```

```
 0.990701632326
```

Berikut ini 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 dekat dengan 0, kami menyimpulkan bahwa pemilihan jenis minuman kaleng tidak bergantung pada jenis kelamin.

```
>contingency(A)
```

```
 0.0427225484717
```

Metode Anova

Selanjutnya kami menggunakan analisis varians (Uji-F) untuk menguji tiga sampel data yang terdistribusi normal untuk nilai rata-rata yang sama. Metode tersebut disebut ANOVA (analisis varians). 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
```

Dalam pengujian hipotesis statistik di atas, hipotesis nol mengasumsikan bahwa nilai rata-rata ke-3 set data (x1,x2,x3) sama. Namun, berdasarkan analisis varians telah kita lakukan, hipotesis nol dapat ditolak dengan probabilitas kesalahan 1,3%. Ini berarti bahwa ada perbedaan yang signifikan secara statistik antara rata-rata dari ketiga set angka.

Sub Topik 9 : Menyimpan Data Hasil Analisis

Data-data yang kita gunakan dalam melakukan analisis statistika dapat kita simpan ke dalam suatu file sehingga ketika kelak ingin digunakan lagi, data tersebut masih ada di file penyimpanan kita. Tak hanya itu, hasil dari analisis statistika yang sudah kita lakukan pun dapat kita simpan ke dalam suatu file.
Berikut adalah cara menyimpan/menulis data ke suatu file.

```
>a=random(1,100); mean(a); dev(a);
>filename="Simpan";
```

Seletah memberi nama untuk file, kita akan menulis vektor a ke dalam file dengan menggunakan fungsi writematrix() dan menggunakan fungsi readmatrix() untuk membaca data.

```
>writematrix(a',filename);
>a=readmatrix(filename)';
```

Kita juga bisa menghapus file yang sudah tersimpan dengan menggunakan fileremove

```
>fileremove(filename);
```

Kemudian kita akan mencoba untuk menggantikan data baru ke file lama dengan menghapus semua data lama, dan menulis lagi data baru yang akan disimpan.

```
>file="Simpan"; open(file,"w");
>writeln("A,B,C"); writematrix(random(3,3));
>close();
>printfile(file)
```

```
A,B,C
0.8351051327697636,0.08458248162153686,0.5192250558799737
0.8548977070796793,0.7679427770316303,0.4018472121828296
0.02466356619093713,0.4253015574769268,0.1649367711598173
```

Selain itu kita juga bisa menyimpan dalam bentuk excel

```
>file="test.csv";
>M=random(3,3); writematrix(M,file);
```

Berikut adalah isi dari file ini.

```
>printfile(file)
```

```
0.2641349455581374,0.6713124187949838,0.135806558906826
0.3437161954193733,0.7730785630232085,0.05730568239045469
0.3519819858104311,0.1107098010416762,0.4220892269525604
```

CVS 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)
```

```
      0.264135      0.671312      0.135807
      0.343716      0.773079     0.0573057
      0.351982       0.11071      0.422089
```