Nama : Ardi Budi Setiawan
NIM : 22305141017
Kelas: Matematika-B

# Menggambar Plot 3D dengan EMT

Ini adalah pengenalan plot 3D di Euler. Kita membutuhkan plot 3D untuk memvisualisasikan fungsi dari dua variabel.

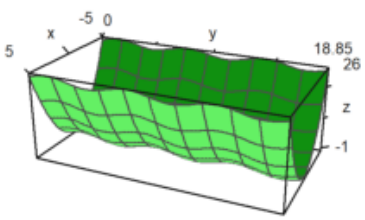
Euler menggambar fungsi tersebut menggunakan algoritma pengurutan untuk menyembunyikan bagian latar belakang. Secara umum, Euler menggunakan proyeksi pusat. Defaultnya adalah dari kuadran-xy positif menuju titik asal x=y=z=0, tetapi angle=0° terlihat dari arah sumbu y. Sudut pandang dan ketinggian dapat diubah.

Euler dapat memplot

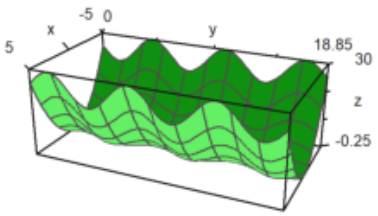
- permukaan dengan bayangan dan garis level atau rentang level,
- titik awan,
- kurva parametrik,
- permukaan implisit.

Plot 3D dari suatu fungsi menggunakan plot3d. Cara termudah adalah dengan memplot ekspresi dalam x dan y. Parameter r mengatur kisaran plot di sekitar (0,0).

```
>aspect(1.5); plot3d("x^2+sin(y)",-5,5,0,6*pi):
```

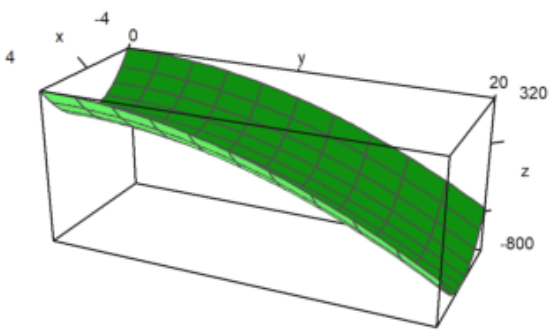



```
>plot3d("x^2+x*sin(y)",-5,5,0,6*pi):
```

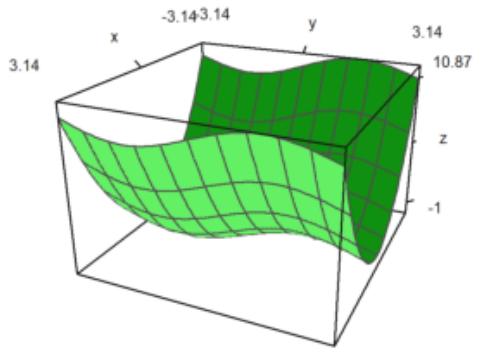



Silakan lakukan modifikasi agar gambar "talang bergelombang" tersebut tidak lurus melainkan melengkung/melingkar, baik melingkar secara mendatar maupun melingkar turun/naik (seperti papan peluncur pada kolam renang. Temukan rumusnya.

```
>aspect(1); plot3d("20*x^2-2*y^2",-4,4,0,20):
```




```
>aspect(1.5); plot3d("x^2+sin(y)",r=pi):
```




## Fungsi Dua Variabel

Untuk grafik suatu fungsi, gunakan

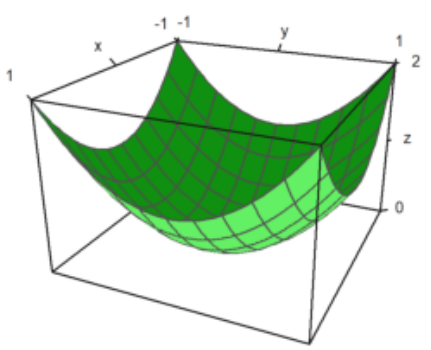
- ekspresi sederhana dalam x dan y,
- nama fungsi dari dua variabel,
- atau data matriks.

Defaultnya adalah kotak kawat yang diisi dengan warna berbeda di kedua sisi. Perhatikan bahwa jumlah default interval kisi adalah 10, tetapi plot menggunakan jumlah default 40x40 persegi panjang untuk membangun permukaan. Ini bisa diubah.

- n=40, n=[40,40]: jumlah garis kisi di setiap arah
- grid=10, grid=[10,10]: jumlah garis kisi di setiap arah

Kita gunakan default n=40 dan grid=10

```
>plot3d("x^2+y^2"):
```

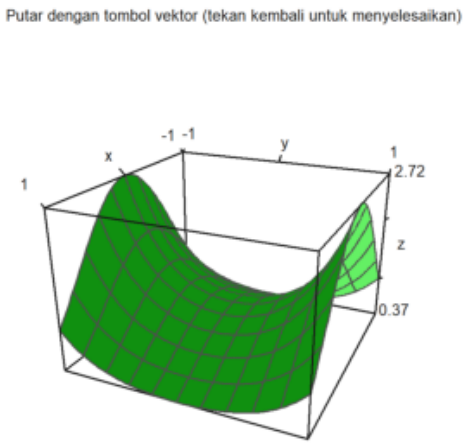



Interaksi pengguna dimungkinkan dengan parameter >user. Pengguna dapat menekan tombol berikut.

- kiri, kanan, atas, bawah: memutar sudut pandang
- +,-: memperbesar atau memperkecil
- a: menghasilkan anaglyph (lihat bawah)
- l: beralih memutar sumber cahaya (lihat bawah)

- spasi: mengatur ulang ke default
- kembali: mengakhiri interaksi

```
>plot3d("exp(-x^2+y^2)",>user, ...
>  title="Putar dengan tombol vektor (tekan kembali untuk menyelesaikan)"):
```



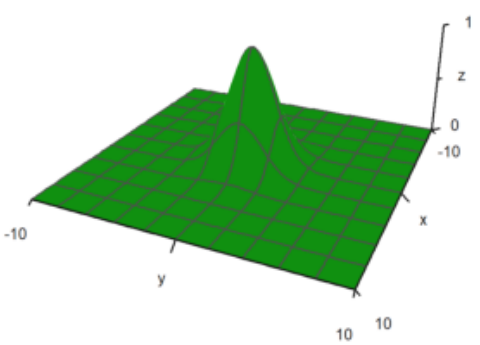

Rentang plot untuk fungsi dapat ditentukan dengan

- a,b: rentang-x
- c,d: rentang-y
- r: persegi simetris di sekitar (0,0).
- n: jumlah subinterval untuk plot.

Ada beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.

fscale: skala ke nilai fungsi (defaultnya adalah <fscale).
scale: angka atau vektor 1x2 untuk skala ke arah x dan y.
frame: jenis bingkai (default 1).

```
>plot3d("exp(-(x^2+y^2)/5)",r=10,n=80,fscale=4,scale=1.2,frame=3):
```



Tampilan dapat diubah dengan berbagai cara.

- distance: jarak pandang ke plot.
- zoom: nilai perbesaran.
- angle: sudut terhadap sumbu-y negatif dalam radian.
- height: ketinggian tampilan dalam radian.

Nilai default dapat diperiksa atau diubah dengan fungsi view(). Ini mengembalikan parameter dalam urutan di atas.

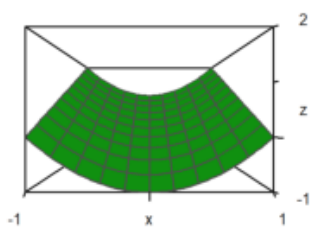
```
>view
```

```
  [5,  2.6,  2,  0.4]
```

Jarak yang lebih dekat membutuhkan lebih sedikit perbesaran. Efeknya lebih seperti lensa sudut lebar.

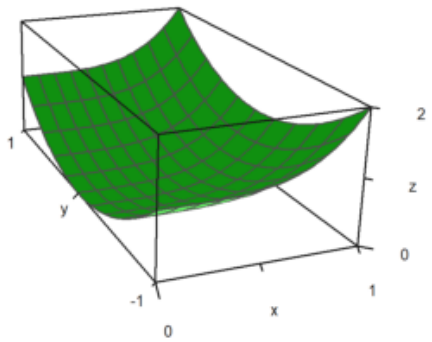
Dalam contoh berikut, angle=0 dan height=0 terlihat dari sumbu-y negatif. Label sumbu untuk y disembunyikan dalam kasus ini.

```
>plot3d("x^2+y",distance=3,zoom=1,angle=0,height=0):
```




Plot terlihat selalu ke pusat plot kubus. Anda dapat memindahkan pusat dengan parameter center.

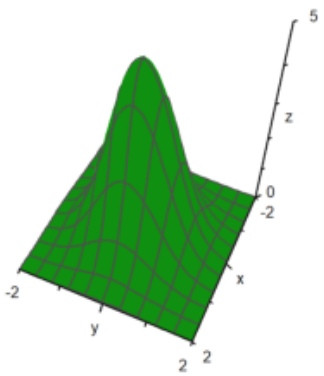
```
>plot3d("x^4+y^2",a=0,b=1,c=-1,d=1,angle=-20°,height=20°, ...
>  center=[0.4,0,0],zoom=4):
```




Plot diskalakan agar sesuai dengan kubus satuan untuk dilihat. Jadi tidak perlu mengubah jarak atau perbesaran tergantung pada ukuran plot. Namun, label mengacu pada ukuran sebenarnya.

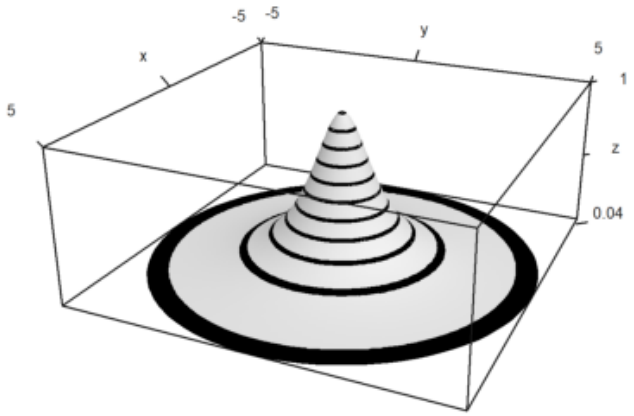
Jika Anda mematikannya dengan scale=false, Anda perlu berhati-hati, bahwa plot masih cocok dengan jendela plot, dengan mengubah jarak pandang atau perbesaran, dan memindahkan pusat.

```
>plot3d("5*exp(-x^2-y^2)",r=2,<fscale,<scale,distance=13,height=50°, ...
>  center=[0,0,-2],frame=3):
```

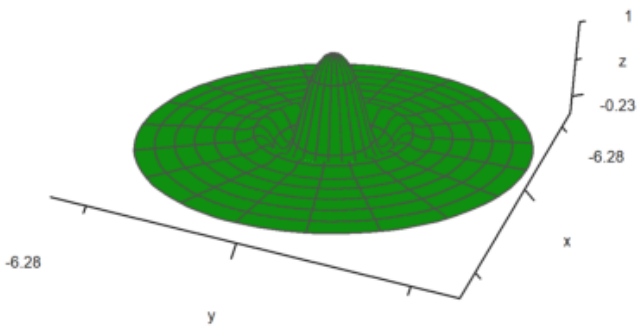



Sebuah plot polar juga tersedia. Parameter polar=true menggambar plot polar. Fungsi tersebut harus tetap merupakan fungsi dari x dan y. Parameter "fscale" menskalakan fungsi dengan skala sendiri. Jika tidak, fungsi diskalakan agar sesuai dengan kubus.

```
>plot3d("1/(x^2+y^2+1)",r=5,>polar, ...
>fscale=2,>hue,n=100,zoom=4,>contour,color=gray):
```



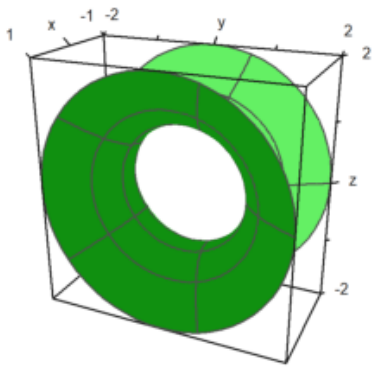

```
>function f(r) := exp(-r/2)*cos(r); ...
>plot3d("f(x^2+y^2)",>polar,scale=[1,1,0.4],r=2pi,frame=3,zoom=4):
```



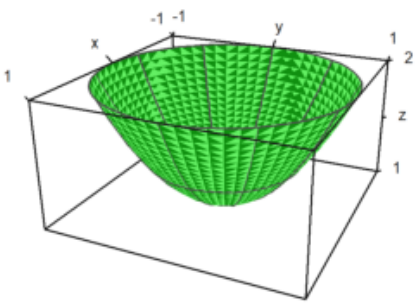

Parameter rotate memutar fungsi dalam x di sekitar sumbu-x.

- rotate=1: Menggunakan sumbu-x
- rotate=2: Menggunakan sumbu-z

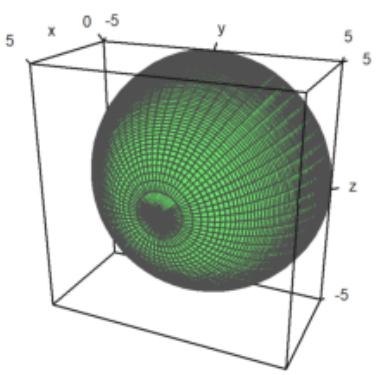
```
>plot3d("x^2+1",a=-1,b=1,rotate=true,grid=5):
```



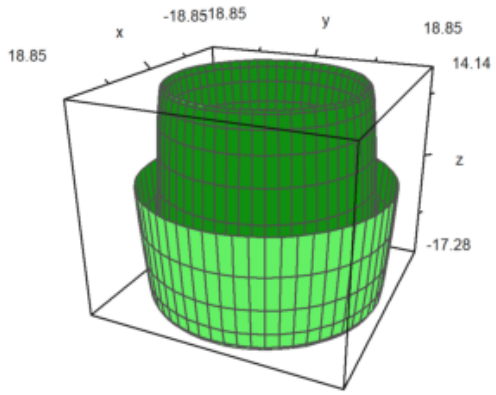

```
>plot3d("x^2+1",a=-1,b=1,rotate=2,grid=5):
```




```
>plot3d("sqrt(25-x^2)",a=0,b=5,rotate=1):
```

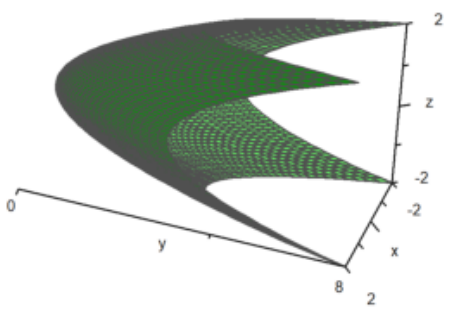


```
>plot3d("x*sin(x)",a=0,b=6pi,rotate=2):
```




Berikut adalah plot dengan tiga fungsi.

```
>plot3d("x","x^2+y^2","y",r=2,zoom=3.5,frame=3):
```




## Plot Kontur

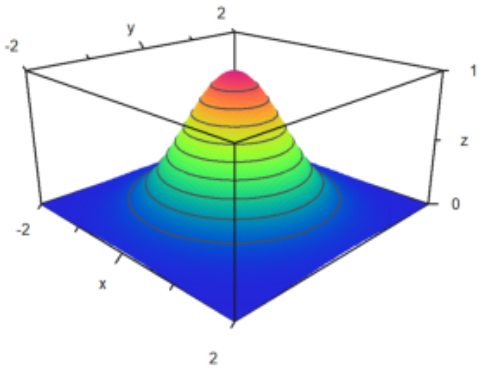
Untuk plot, Euler menambahkan garis kisi. Sebagai gantinya dimungkinkan untuk menggunakan garis level dan rona satu warna atau rona berwarna spektral. Euler dapat menggambar fungsi tinggi pada plot dengan bayangan. Di semua plot 3D, Euler dapat menghasilkan anaglyph merah/cyan.

- >hue: Menyalakan bayangan cahaya alih-alih kabel.
- >contour: Memplot garis kontur otomatis pada plot.
- level=... (atau levels): Sebuah vektor nilai untuk garis kontur.

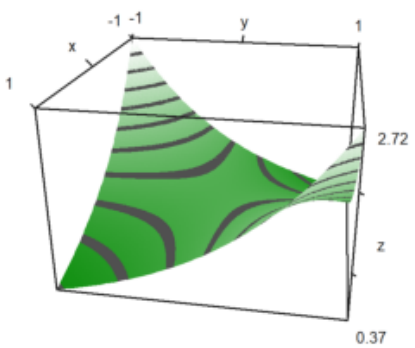
Defaultnya adalah level="auto", yang menghitung beberapa garis level secara otomatis. Seperti yang Anda lihat di plot, level sebenarnya adalah rentang level.

Gaya default dapat diubah. Untuk plot kontur berikut, kita gunakan kisi yang lebih halus untuk 100x100 poin, skala fungsi dan plot, dan menggunakan sudut pandang yang berbeda.

```
>plot3d("exp(-x^2-y^2)",r=2,n=100,level="thin", ...
> >contour,>spectral,fscale=1,scale=1.1,angle=45°,height=20°):
```




```
>plot3d("exp(x*y)",angle=100°,>contour,color=green):
```

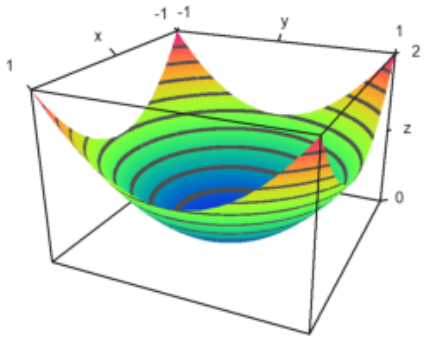



Bayangan default menggunakan warna abu-abu. Tetapi, rentang warna spektral juga tersedia.

- >spectral: Menggunakan skema spektral default
- color=...: Menggunakan warna khusus atau skema spektral

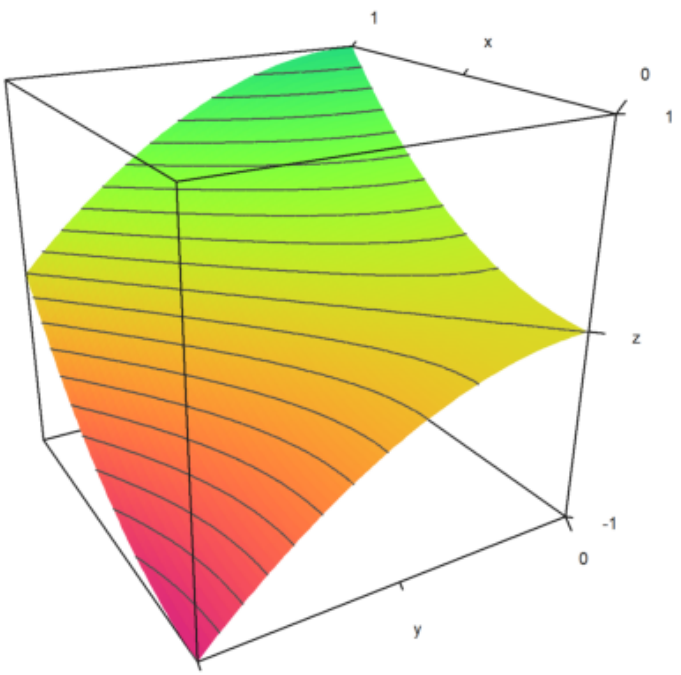
Untuk plot berikut, kita gunakan skema spektral default dan menambah jumlah titik untuk mendapatkan tampilan yang sangat halus.

```
>plot3d("x^2+y^2",>spectral,>contour,n=100):
```




Alih-alih garis level otomatis, kita juga dapat mengatur nilai garis level. Ini akan menghasilkan garis level tipis alih-alih rentang level.

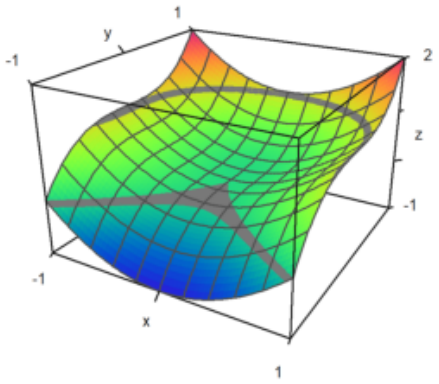
```
>plot3d("x^2-y^2",0,1,0,1,angle=240°,level=-1:0.1:1,color=redgreen):
```




Dalam plot berikut, kita gunakan dua pita level yang sangat luas dari -0.1 hingga 1, dan dari 0.9 hingga 1. Ini dimasukkan sebagai matriks dengan batas level sebagai kolom.

Selain itu, kita lapisi kisi dengan 10 interval di setiap arah.

```
>plot3d("x^2+y^3",level=[-0.1,0.9;0,1], ...
>  >spectral,angle=30°,grid=10,contourcolor=gray):
```

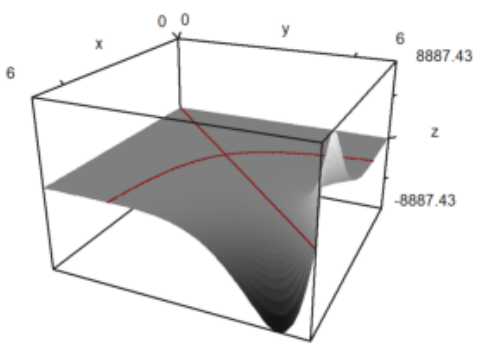



Dalam contoh berikut, kita memplot himpunan, di mana

$$f(x,y) = x^y - y^x = 0$$

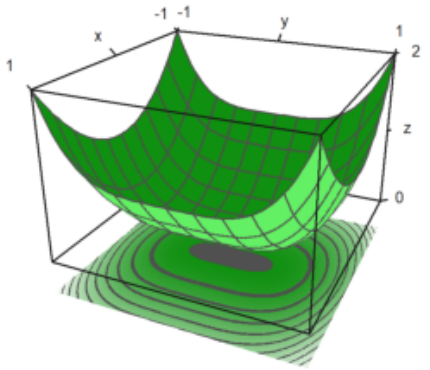
Kita gunakan garis tipis tunggal untuk garis level.

```
>plot3d("x^y-y^x",level=0,a=0,b=6,c=0,d=6,contourcolor=red,n=100):
```



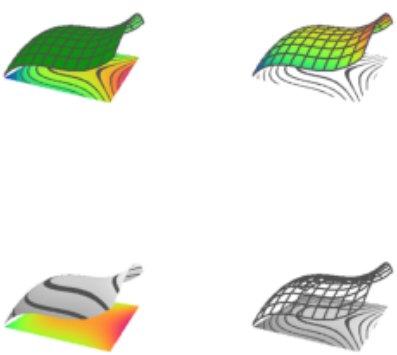

Dimungkinkan untuk menunjukkan bidang kontur di bawah plot. Warna dan jarak ke plot dapat ditentukan.

```
>plot3d("x^2+y^4",>cp,cpcolor=green,cpdelta=0.2):
```




Berikut adalah beberapa gaya lagi. Kita selalu mematikan frame, dan menggunakan berbagai skema warna untuk plot dan kisi.

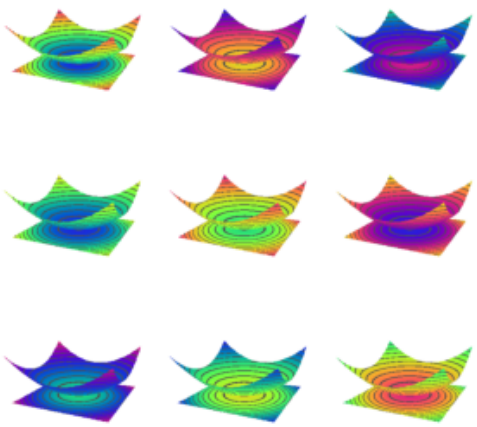
```
>figure(2,2); ...
>expr="y^3-x^2"; ...
>figure(1);  ...
>  plot3d(expr,<frame,>cp,cpcolor=spectral); ...
>figure(2);  ...
>  plot3d(expr,<frame,>spectral,grid=10,cp=2); ...
>figure(3);  ...
>  plot3d(expr,<frame,>contour,color=gray,nc=5,cp=3,cpcolor=greenred); ...
>figure(4);  ...
>  plot3d(expr,<frame,>hue,grid=10,>transparent,>cp,cpcolor=gray); ...
>figure(0):
```



Ada beberapa skema spektral lainnya, diberi nomor 1 hingga 9. Tetapi Anda juga dapat mengggunakan color=value, di mana nilai

- spectral: untuk rentang dari biru ke merah
- white: untuk rentang yang lebih redup
- yellowblue,purplegreen,blueyellow,greenred
- blueyellow, greenpurple,yellowblue,redgreen

```
>figure(3,3); ...
>for i=1:9;  ...
>  figure(i); plot3d("x^2+y^2",spectral=i,>contour,>cp,<frame,zoom=4);  ...
>end; ...
>figure(0):
```

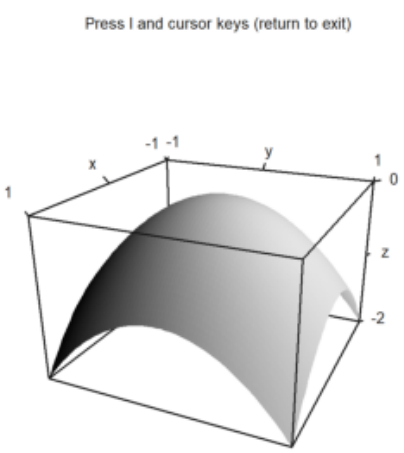


Sumber cahaya dapat diubah dengan 1 dan tombol kursor selama interaksi pengguna. Itu juga dapat diatur dengan parameter.

- light: arah cahaya
- amb: cahaya sekitar antara 0 dan 1

Perhatikan bahwa program tidak membuat perbedaan antara sisi plot. Tidak ada bayangan. Untuk ini, Anda perlu Povray.

```
>plot3d("-x^2-y^2", ...
>  hue=true,light=[0,1,1],amb=0,user=true, ...
>  title="Press l and cursor keys (return to exit)"):
```

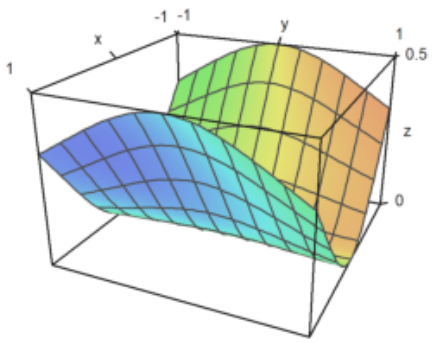


Parameter warna mengubah warna permukaan. Warna garis level juga dapat diubah.

```
>plot3d("-x^2-y^2",color=rgb(0.2,0.2,0),hue=true,frame=false, ...
>  zoom=3,contourcolor=red,level=-2:0.1:1,dl=0.01):
```

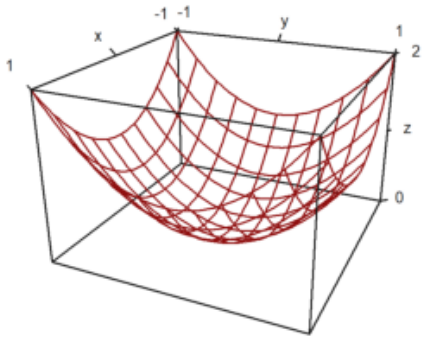
Warna 0 memberikan efek pelangi khusus.

```
>plot3d("x^2/(x^2+y^2+1)",color=0,hue=true,grid=10):
```




Permukaannya juga bisa transparan.

```
>plot3d("x^2+y^2",>transparent,grid=10,wirecolor=red):
```




## Plot Implisit

Ada juga plot implisit dalam tiga dimensi. Euler menghasilkan pemotongan melalui objek. Fitur plot3d termasuk plot implisit. Plot-plot ini menunjukkan himpunan nol dari suatu fungsi dalam tiga variabel.

Solusi dari

$$f(x,y,z)=0$$

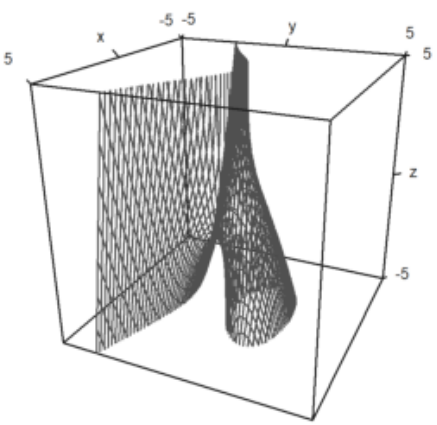
Dapat divisualisasikan dalam potongan sejajar dengan bidang x-y-, x-z-, dan y-z-.

- implicit=1: memotong sejajar dengan bidang-yz
- implicit=2: memotong sejajar dengan bidang-xz
- implicit=4: memotong sejajar dengan bidang-xy

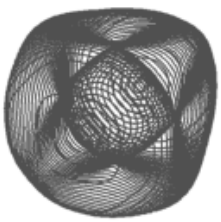
Tambahkan nilai ini, jika Anda mau. Dalam contoh kita memplot

$$M=\{(x,y,z):x^2+y^3+zy=1\}$$

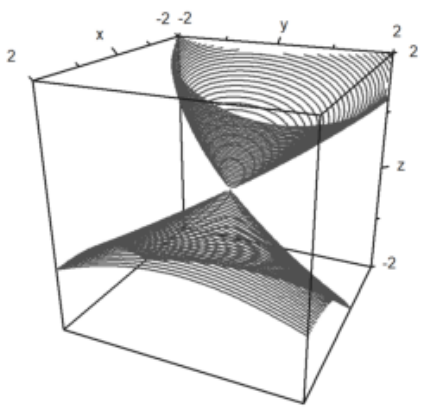
```
>plot3d("x^2+y^3+z*y-1",r=5,implicit=3):
```



```
>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):
```




## Memplot Data 3D

Sama seperti plot2d, plot3d menerima data. Untuk objek 3D, Anda perlu menyediakan matriks nilai x,y, dan z atau tiga fungsi atau ekspresi fx(x,y), fy(x,y), fz(x,y).

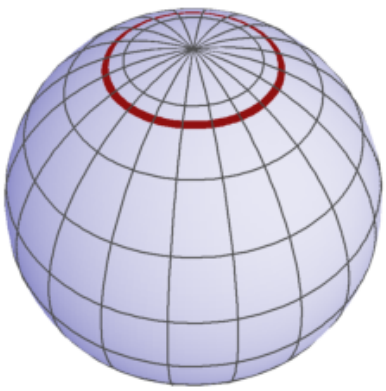
$$\gamma(t,s) = (x(t,s), y(t,s), z(t,s))$$

Karena x,y,z adalah matriks, kita asumsikan bahwa (t,s) berlari melalui kotak perseg. Hasilnya, Anda dapat memplot gambar persehi panjang di ruang angkasa.

Anda dapat menggunakan bahasa matriks Euler untuk menghasilkan koordinat secara efektif.

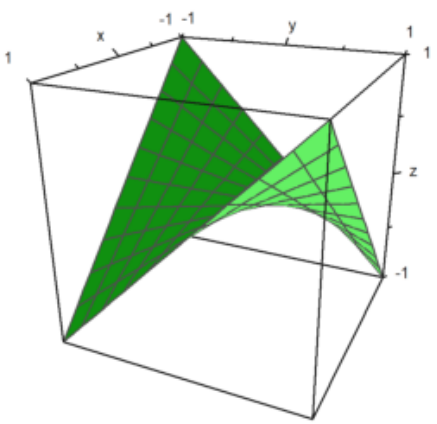
Dalam contoh berikut, kita gunakan vektor nilai t baris dan vektor nilai kolom s untuk membuat parameter permukaan bola. Dalam gambar kita dapat menandai daerah, dalam kasus kita daerah kutub.

```
>t=linspace(0,2pi,180); s=linspace(-pi/2,pi/2,90)'; ...
>x=cos(s)*cos(t); y=cos(s)*sin(t); z=sin(s); ...
>plot3d(x,y,z,>hue, ...
>color=blue,<frame,grid=[10,20], ...
>values=s,contourcolor=red,level=[90°-24°;90°-22°], ...
>scale=1.4,height=50°):
```



Berikut adalah contoh, yang merupakan grafik fungsi.

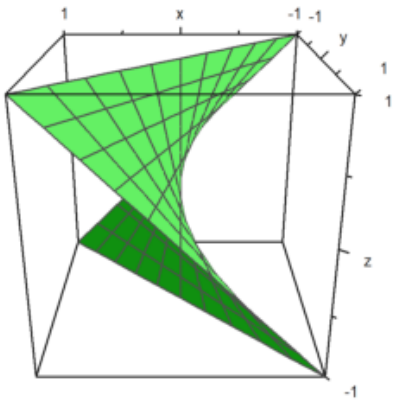
```
>t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,t*s,grid=10):
```




Namun, kita dapat membuat segala macam permukaan. Berikut adalah permukaan yang sama dengan fungsi

$$x = y\,z$$

```
>plot3d(t*s,t,s,angle=180°,grid=10):
```




Dengan lebih banyak usaha, kita 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

$$0 \le t \le 2\pi, \quad \frac{-\pi}{2} \le s \le \frac{\pi}{2}.$$

Kita distorsi ini dengan faktor

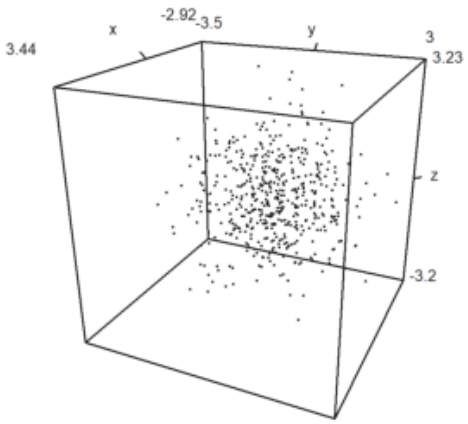
$$d(t,s) = \frac{\cos(4t)+\cos(8s)}{4}.$$

```
>t=linspace(0,2pi,320); s=linspace(-pi/2,pi/2,160)'; ...
>d=1+0.2*(cos(4*t)+cos(8*s)); ...
>plot3d(cos(t)*cos(s)*d,sin(t)*cos(s)*d,sin(s)*d,hue=1, ...
>  light=[1,0,1],frame=0,zoom=5):
```

Tentu saja, titik awan juga dimungkinkan. Untuk memplot data titik dalam ruang, kita membutuhkan tiga vektor untuk koordinat titik-titik tersebut.

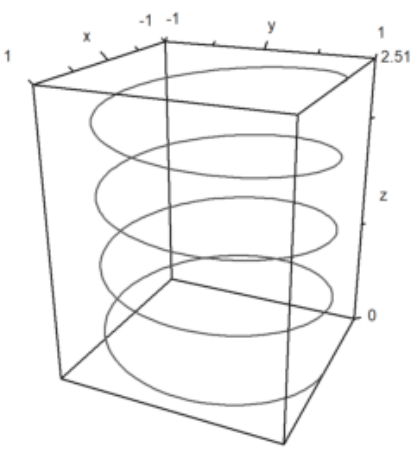
Gayanya sama seperti di plot2d dengan points=true;

```
>n=500;  ...
>  plot3d(normal(1,n),normal(1,n),normal(1,n),points=true,style="."):
```

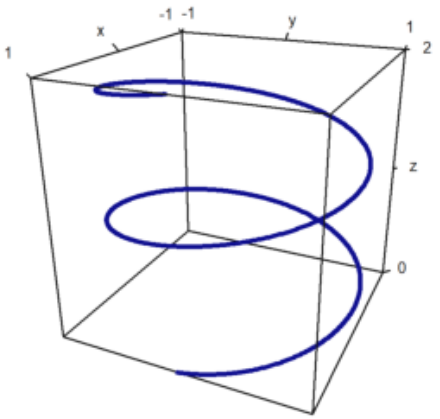



Dimungkinkan juga untuk memplot kurva dalam 3D. Dalam hal ini, lebih mudah untuk menghitung titik-titik kurva. Untuk kurva di bidang kita gunakan urutan koordinat dan parameter wire=true.

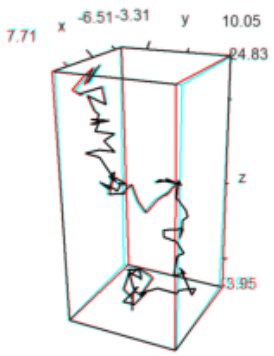
```
>t=linspace(0,8pi,500); ...
>plot3d(sin(t),cos(t),t/10,>wire,zoom=3):
```




```
>t=linspace(0,4pi,1000); plot3d(cos(t),sin(t),t/2pi,>wire, ...
>linewidth=3,wirecolor=blue):
```

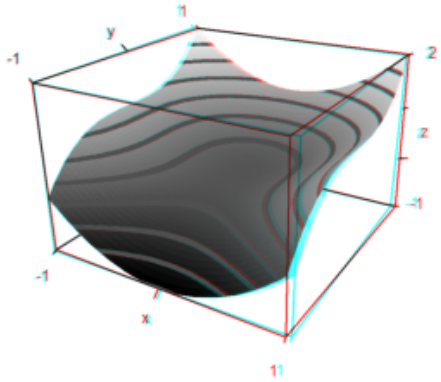



```
>X=cumsum(normal(3,100)); ...
> plot3d(X[1],X[2],X[3],>anaglyph,>wire):
```



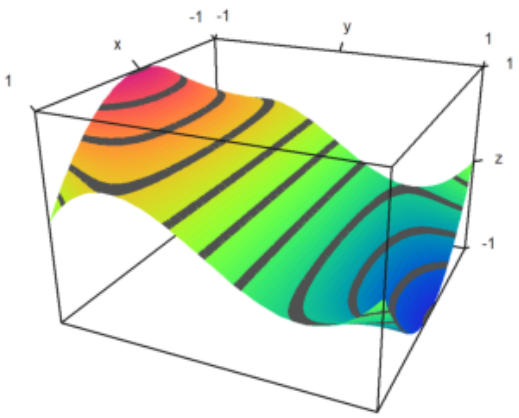

EMT juga dapat memplot dalam mode anaglyph. Untuk melihat plot seperti itu, Anda memerlukan kacamata merah/cyan

```
> plot3d("x^2+y^3",>anaglyph,>contour,angle=30°):
```




Seringkali, skema warna spektral digunakan untuk plot. Ini menekankan fungsi tinggi.

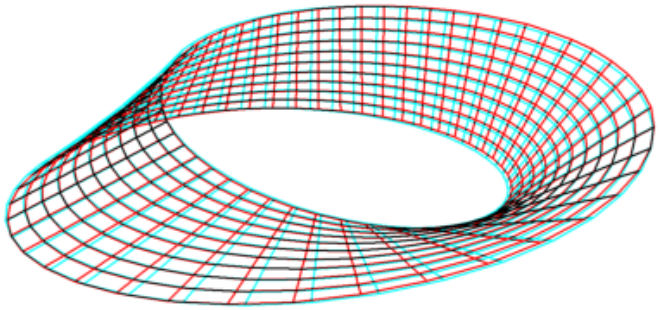
```
>plot3d("x^2*y^3-y",>spectral,>contour,zoom=3.2):
```




Euler juga dapat memplot permukaan berparameter, ketika parameternya adalah nilai x, y, dan z dari gambar kotak persegi panjang dalam ruang.

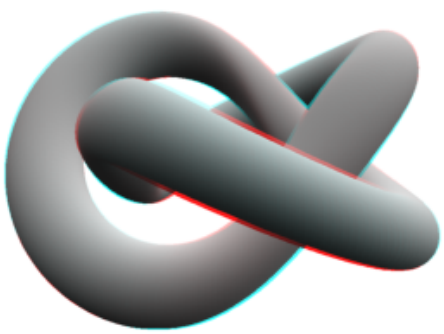
Untuk demo berikut, kita atur parameter u dan v, dan menghasilkan koordinat ruang dari ini.

```
>u=linspace(-1,1,10); v=linspace(0,2*pi,50)'; ...
>X=(3+u*cos(v/2))*cos(v); Y=(3+u*cos(v/2))*sin(v); Z=u*sin(v/2); ...
>plot3d(X,Y,Z,>anaglyph,<frame,>wire,scale=2.3):
```



Berikut adalah contoh yang lebih rumit, yang megah dengan kacamata merah/cyan.

```
>u:=linspace(-pi,pi,160); v:=linspace(-pi,pi,400)';  ...
>x:=(4*(1+.25*sin(3*v))+cos(u))*cos(2*v); ...
>y:=(4*(1+.25*sin(3*v))+cos(u))*sin(2*v); ...
> z=sin(u)+2*cos(3*v); ...
>plot3d(x,y,z,frame=0,scale=1.5,hue=1,light=[1,0,-1],zoom=2.8,>anaglyph):
```



## Plot Statistik

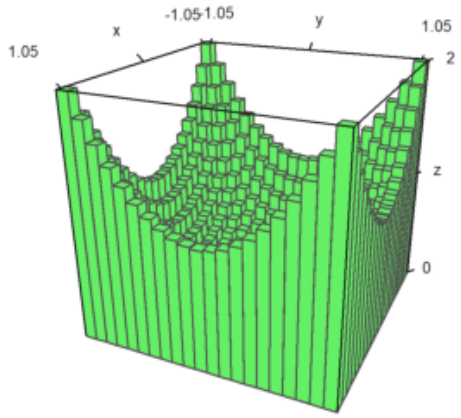
Plot batang juga dimungkinkan. Untuk ini, kita harus menyediakan

- x: vektor baris dengan n+1 elemen
- y: vektor kolom dengan n+1 elemen
- z: nxn nilai matriks

z bisa lebih besar, tetapi hanya nilai nxn yang akan digunakan.

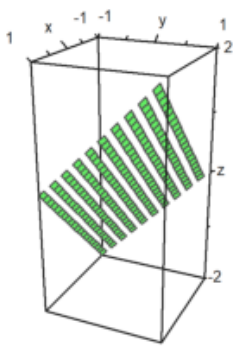
Dalam contoh, pertama-tama kita menghitung nilainya. Kemudian kita sesuaikan x dan y, sehingga vektor berpusat pada nilai yang digunakan.

```
>x=-1:0.1:1; y=x'; z=x^2+y^2; ...
>xa=(x|1.1)-0.05; ya=(y_1.1)-0.05; ...
>plot3d(xa,ya,z,bar=true):
```



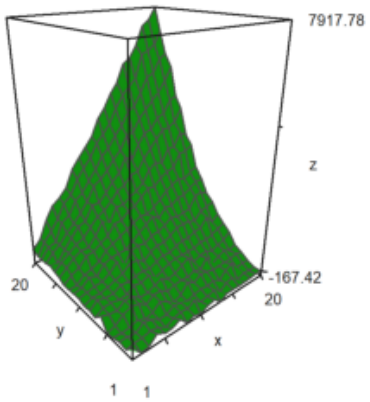

Dimungkinkan untuk membagi plot permukaan menjadi dua atau lebih bagian.

```
>x=-1:0.1:1; y=x'; z=x+y; d=zeros(size(x)); ...
>plot3d(x,y,z,disconnect=2:2:20):
```

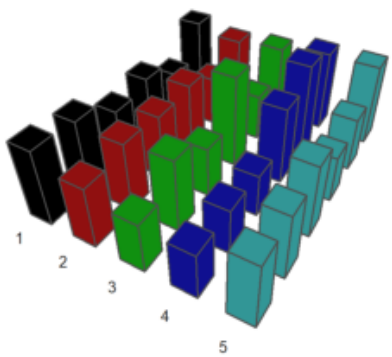



Jika memuat atau menghasilkan matriks data M dari file dan perlu memplotnya dalam 3D, Anda dapat menskalakan matriks ke [-1,1] dengan scale(M), atau menskalakan matriks dengan >zscale. Ini dapat dikombinasikan dengan faktor penskalaan individu yang diterapkan sebagai tambahan.

```
>i=1:20; j=i'; ...
>plot3d(i*j^2+100*normal(20,20),>zscale,scale=[1,1,1.5],angle=-40°,zoom=1.8):
```

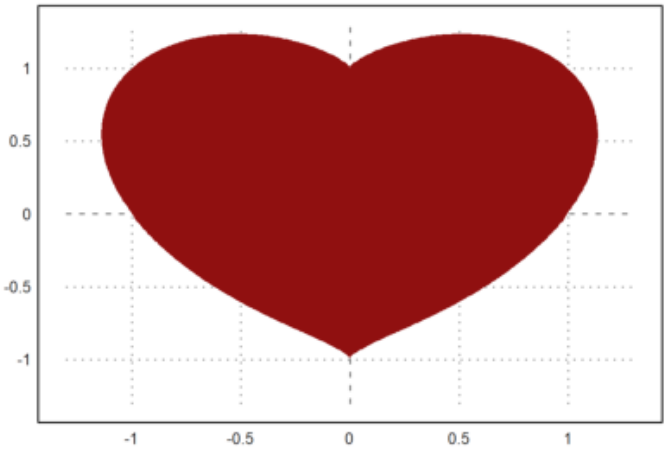



```
>Z=intrandom(5,100,6); v=zeros(5,6); ...
>loop 1 to 5; v[#]=getmultiplicities(1:6,Z[#]); end; ...
>columnsplot3d(v',scols=1:5,ccols=[1:5]):
```




## Permukaan Benda Putar

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




```
>ekspresi &= (x^2+y^2-1)^3-x^2*y^3; $ekspresi
```

$$\left(y^2+x^2-1\right)^3-x^2\,y^3$$

Kita ingin memutar lekukan hati di sekitar sumbu-y. Berikut adalah ekspresi, yang mendefinisikan hati:

$$f(x,y)=(x^2+y^2-1)^3-x^2.y^3.$$

Selanjutnya kita atur

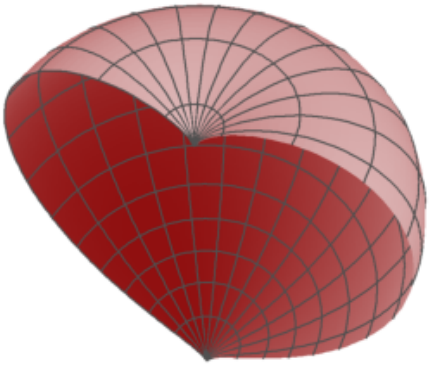
$$x=r.cos(a),\quad y=r.sin(a).$$

```
>function fr(r,a) &= ekspresi with [x=r*cos(a),y=r*sin(a)] | trigreduce; $fr(r,a)
```

$$\left(r^2-1\right)^3+\frac{\left(\sin\left(5\,a\right)-\sin\left(3\,a\right)-2\,\sin a\right)\,r^5}{16}$$

Hal ini memungkinkan untuk mendefinisikan fungsi numerik, yang memecahkan r, jika a diberikan. Dengan fungsi itu kita dapat memplot jantung yang diputar sebagai permukaan parametrik.

```
>function map f(a) := bisect("fr",0,2;a); ...
>t=linspace(-pi/2,pi/2,100); r=f(t);  ...
>s=linspace(pi,2pi,100)'; ...
>plot3d(r*cos(t)*sin(s),r*cos(t)*cos(s),r*sin(t), ...
>>hue,<frame,color=red,zoom=4,amb=0,max=0.7,grid=12,height=50°):
```

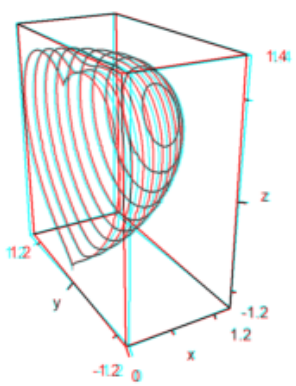


Berikut ini adalah plot3D dari gambar di atas yang diputar di sekitar sumbu-z. Kita definisikan fungsi, yang menggambarkan objek.

```
>function f(x,y,z) ...
```

```
  r=x^2+y^2;
  return (r+z^2-1)^3-r*z^3;
   endfunction
```

```
>plot3d("f(x,y,z)", ...
>xmin=0,xmax=1.2,ymin=-1.2,ymax=1.2,zmin=-1.2,zmax=1.4, ...
>implicit=1,angle=-30°,zoom=2.5,n=[10,60,60],>anaglyph):
```




## Plot 3D Khusus

Fungsi plot3d bagus untuk dimiliki, tetapi tidak memenuhi semua kebutuhan. Selain rutinitas yang lebih mendasar, dimungkinkan untuk mendapatkan plot berbingkai dari objek apapun yang Anda suka.

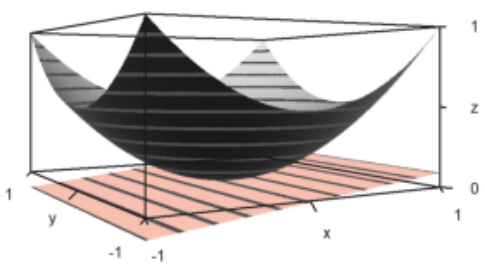
Meskipun Euler bukan program 3D, ia dapat menggabungkan beberapa objek dasar. Kita coba memvisualisasikan paraboloida dan garis singgungnya.

```
>function myplot ...

    y=-1:0.01:1; x=(-1:0.01:1)';
    plot3d(x,y,0.2*(x-0.1)/2,<scale,<frame,>hue, ..
      hues=0.5,>contour,color=orange);
    h=holding(1);
    plot3d(x,y,(x^2+y^2)/2,<scale,<frame,>contour,>hue);
    holding(h);
  endfunction
```

Sekarang framedplot() menyediakan bingkai, dan mengatur tampilan.

```
>framedplot("myplot",[-1,1,-1,1,0,1],height=0,angle=-30°, ...
>  center=[0,0,-0.7],zoom=3):
```

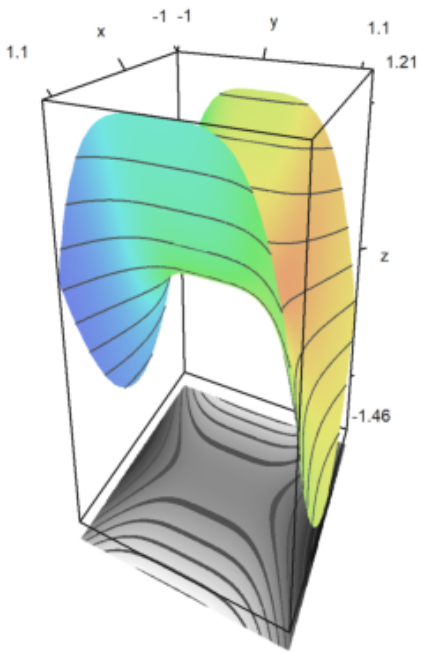



Dengan cara yang sama, Anda dapat memplot bidang kontur secara manual. Perhatikan bahwa plot3d() mengatur jendela ke fullwindow() secara default, tetapi plotcontourplane() mengasumsikan itu.

```
>x=-1:0.02:1.1; y=x'; z=x^2-y^4;
>function myplot (x,y,z) ...

    zoom(2);
    wi=fullwindow();
    plotcontourplane(x,y,z,level="auto",<scale);
    plot3d(x,y,z,>hue,<scale,>add,color=white,level="thin");
    window(wi);
    reset();
  endfunction

>myplot(x,y,z):
```



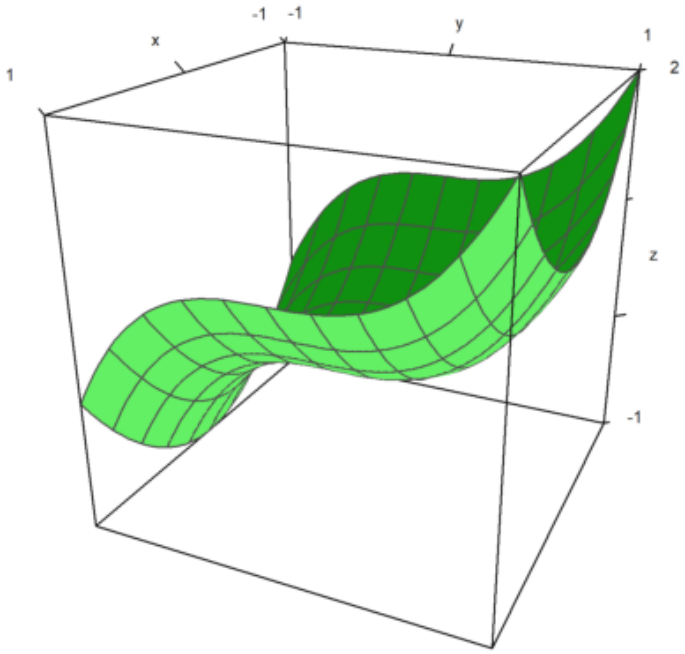

## Animasi

Euler dapat menggunakan frame untuk menghitung animasi terlebih dahulu.

Salah satu fungsi yang memanfaatkan teknik ini adalah rotate. Ini dapat mengubah sudut pandang dan menggambar ulang plot 3D. Fungsi memanggil addpage() untuk setiap plot baru. Akhirnya itu menganimasikan plot.

Silakan pelajari sumber rotate untuk melihat lebih detail.

```
>function testplot () := plot3d("x^2+y^3"); ...
>rotate("testplot"); testplot():
```




## Menggambar Povray

Dengan bantuan file Euler povray.e, Euler dapat menghasilkan file Povray. Hasilnya sangat bagus untuk dilihat.

Anda perlu menginstal Povray (32bit atau 64bit) dari http://www.povray.org/
dan meletakkan sub-direktori "bin" dari Povray ke path lingkungan, atau mengatur variabel "defaultpovray" dengan path lengkap yang menunjuk ke :pvengine.exe".

Antarmuka Povray dari Euler menghasilkan file Povray di direktori home pengguna, dan memanggil Povray untuk mengurai file-file ini. Nama file default adalah current.pov, dan direktori default adalah eulerhome(), biasanya c:Users\Username\Euler. Povray menghasilkan file PNG, yang dapat dimuat oleh Euler ke dalam notebook. Untuk mmebersihkan file-file ini, gunakan povclear().

Fungsi pov3d memiliki semangat yang sama dengan plot3d. Ini bisa menghasilkan grafik fungsi f(x,y), atau permukaan dengan koordinat X,Y,Z dalam matriks, termasuk garis level opsional. Fungsi ini memulai ray-tracer secara otomatis, dan memuat adegan ke dalam notebook Euler.

Selain pov3d(), ada banyak fungsi yang menghasilkan objek Povray. Fungsi-fungsi ini mengembalikan string, yang berisi kode Povray untuk objek. Untuk menggunakan fungsi ini, mulai file Povray dengan povstart(). Kemudian gunakan write1n(...) untuk menulis objek ke file adegan. Terakhir, akhiri file dengan povend(). Secara default, raytracer akan dimulai, dan PNG akan dimasukkan ke dalam notebook Euler.

Fungsi objek memiliki parameter yang disebut "look", yang membutuhkan string dengan kode Povray untuk tekstur dan hasil akhir objek. Fungsi povlook() dapat digunakan untuk menghasilkan string ini. Ini memiliki parameter untuk warna, transparansi, Shading Pong dll.

Perhatikan bahwa dunia Povray memiliki sistem koordinat lain. Antarmuka ini menerjemahkan semua koordinat ke sistem Povray. Jadi Anda dapat terus berpikir dalam sistem koordinat Euler dengan z menunjuk vertikal ke atas, dan sumbu x,y,z dalam arti tangan kanan.
Anda perlu memuat file povray.

```
>load povray;
```

Pastikan direktori Povray bin di path. Jika tidak, edit variabel berikut sehingga berisi path ke povray yang dapat dieksekusi.

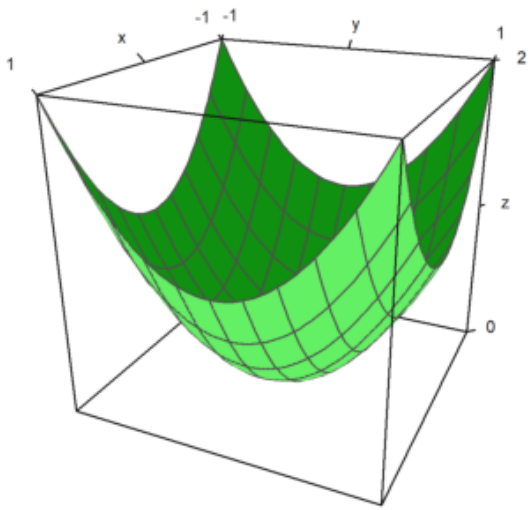
```
>defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

```
  C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe
```

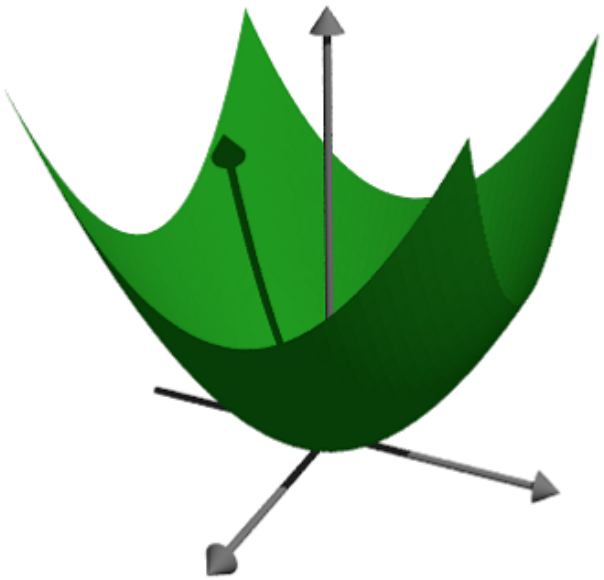
Untuk kesan pertama, kita memplot fungsi sederhana. Perintah berikut menghasilkan file povray di direktori pengguna Anda, dan menjalankan Povray untuk ray tracing file ini.

Jika Anda memulai perintah berikut, GUI Povray akan terbuka, menjalankan file, dan menutup secara otomatis. Karena alasan keamanan, Anda akan ditanya, apakah Anda ingin mengizinkan file exe untuk dijalankan. Anda dapat menekan cancel untuk menghentikan pertanyaan lebih lanjut. Anda mungkin harus menekan OK di jendela Povray untuk mengakui dialog awal Povray.

```
>plot3d("x^2+y^2",zoom=2):
```

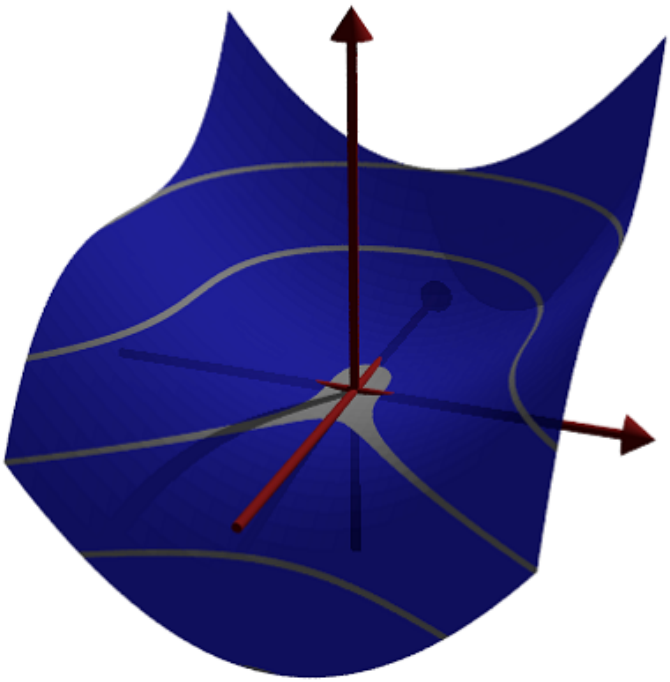



```
>pov3d("x^2+y^2",zoom=3);
```



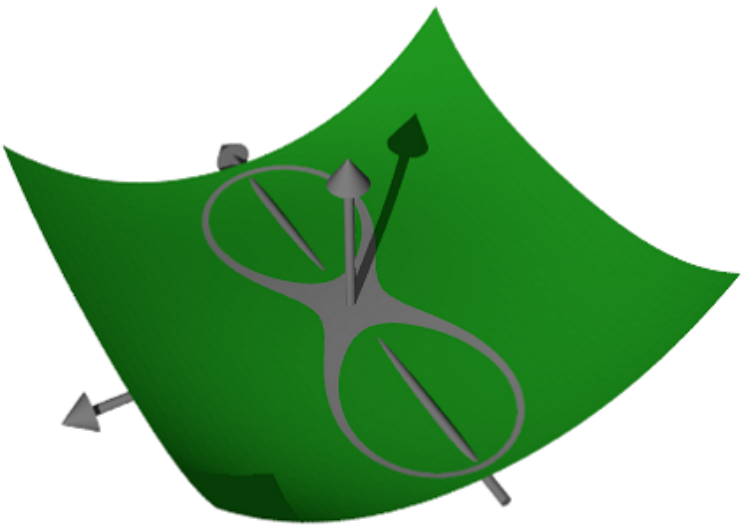
Kita dapat membuat fungsi transparan dan menambahkan hasil akhir lainnya. Kita juga dapat menambahkan garis level ke plot fungsi.

```
>pov3d("x^2+y^3",axiscolor=red,angle=20°, ...
>  look=povlook(blue,0.2),level=-1:0.5:1,zoom=3.8);
```



Terkadang perlu untuk mencegah penskalaan fungsi, dan menskalakan fungsi dengan tangan.
Kita plot himpunan titik di bidang kompleks, di mana produk dari jarak ke 1 dan -1 sama dengan 1.

```
>pov3d("((x-1)^2+y^2)*((x+1)^2+y^2)/40",r=1.5, ...
>  angle=-120°,level=1/40,dlevel=0.005,light=[-1,1,1],height=45°,n=50, ...
>  <fscale,zoom=3.8);
```

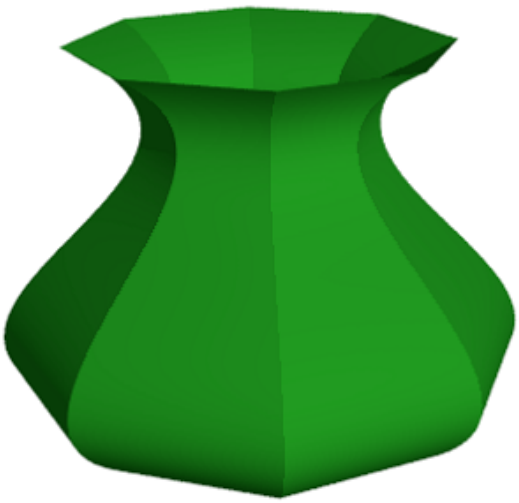


## Memplot dengan Koordinat

Alih-alih fungsi, kita dapat memplot dengan koordinat. Seperti pada plot3d, kita membutuhkan tiga matriks untuk mendefinisikan objek.

Dalam contoh kita memutar fungsi di sekitar sumbu-z.

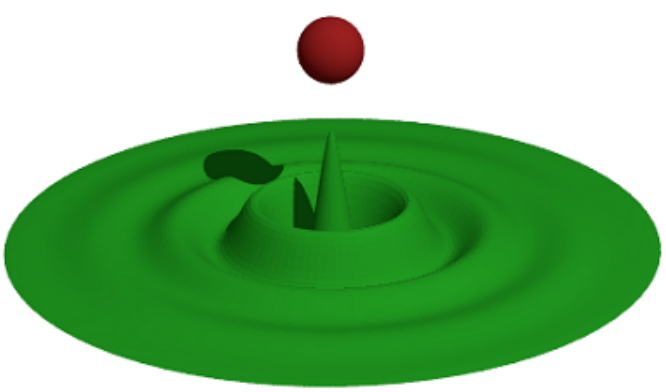
```
>function f(x) := x^3-x+1; ...
>x=-1:0.01:1; t=linspace(0,2pi,8)'; ...
>Z=x; X=cos(t)*f(x); Y=sin(t)*f(x); ...
>pov3d(X,Y,Z,angle=40°,height=20°,axis=0,zoom=4,light=[10,-5,5]);
```



Dalam contoh berikut, kita plot gelombang teredam. Kita hasilkan gelombang dengan bahasa matriks Euler.

Kita juga tunjukkan, bagaimana objek tambahan dapat ditambahkan ke adegan pov3d. Untuk pembuatan objek, lihat contoh berikut. Perhatikan bahwa plot3d meskalakan plot, sehingga cocok dengan kubus satuan.

```
>r=linspace(0,1,80); phi=linspace(0,2pi,80)'; ...
>x=r*cos(phi); y=r*sin(phi); z=exp(-5*r)*cos(8*pi*r)/3;  ...
>pov3d(x,y,z,zoom=5,axis=0,add=povsphere([0,0,0.5],0.1,povlook(red)), ...
>  w=500,h=300);
```



Dengan metode bayangan yang canggih dari Povray, sangat sedikit titik yang dapat menghasilkan permukaan yang sangat halus. Hanya di perbatasan dan dalam bayang-bayang triknya mungkin menjadi jelas.

Untuk ini, kita perlu menambahkan vektor normal di setiap titik matriks.

```
>Z &= x^2*y^3
```

$$x^2 y^3$$

Persamaan permukaannya adalah [x,y,z]. Kita hitung dua turunan terhadap x dan y ini dan mengambil produk silang sebagai normal.

```
>dx &= diff([x,y,Z],x); dy &= diff([x,y,Z],y);
```

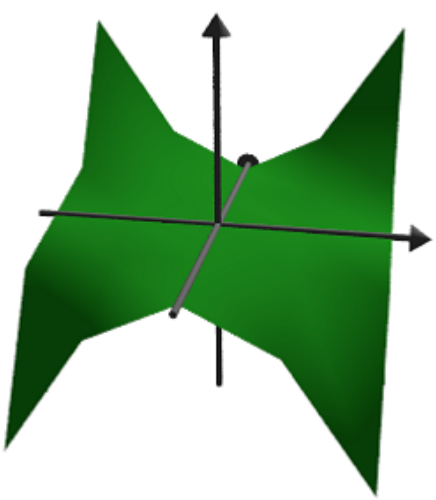
Kita definisikan normal sebagai produk silang dari turunan ini, dan mendefinisikan fungsi koordinat.

```
>N &= crossproduct(dx,dy); NX &= N[1]; NY &= N[2]; NZ &= N[3]; N,
```

$$\left[-2 x y^3, -3 x^2 y^2, 1\right]$$

Kita hanya menggunakan 25 titik.

```
>x=-1:0.5:1; y=x';
>pov3d(x,y,Z(x,y),angle=10°, ...
>  xv=NX(x,y),yv=NY(x,y),zv=NZ(x,y),<shadow);
```



Berikut ini adalah simpul Trefoil yang dilakukan oleh A. Busser di Povray. Ada versi yang ditingkatkan dari ini dalam contoh.

See: Examples\Trefoil Knot | Trefoil Knot

Untuk tampilan yang bagus dengan tidak terlalu banyak titik, kita tambahkan vektor normal di sini. Kita gunakan Maxima untuk menghitung normal untuk kita. Pertama, ketiga fungsi koordinat sebagai ekspresi simbolik.

```
>X &= ((4+sin(3*y))+cos(x))*cos(2*y); ...
>Y &= ((4+sin(3*y))+cos(x))*sin(2*y); ...
>Z &= sin(x)+2*cos(3*y);
```

Kemudian kedua vektor turunan terhadap x dan terhadap y.

```
>dx &= diff([X,Y,Z],x); dy &= diff([X,Y,Z],y);
```

Sekarang normal, yang merupakan produk silang dari dua turunan.

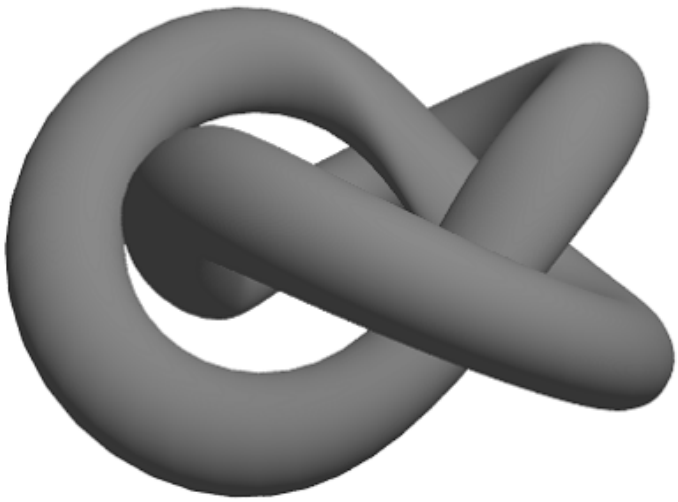
```
>dn &= crossproduct(dx,dy);
```

Kita sekarang mengevaluasi semua ini secara numerik.

```
>x:=linspace(-%pi,%pi,40); y:=linspace(-%pi,%pi,100)';
```

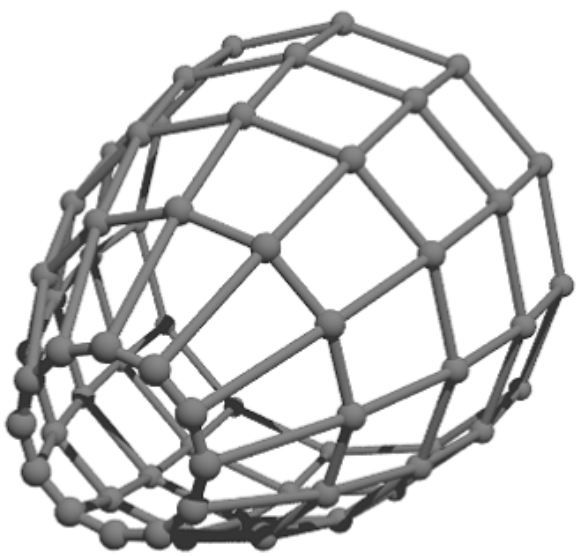
Vektor normal adalah evaluasi dari ekspresi simbolik dn[i] untuk i=1,2,3. Sintaks untuk ini adalah &"expression"(paramters). Ini adalah alternatif dari metode pada contoh sebelumnya, di mana kita mendefinisikan ekspresi simbolik NX, NY, NZ terlebih dahulu.

```
>pov3d(X(x,y),Y(x,y),Z(x,y),axis=0,zoom=5,w=450,h=350, ...
>  <shadow,look=povlook(gray), ...
>  xv=&"dn[1]"(x,y), yv=&"dn[2]"(x,y), zv=&"dn[3]"(x,y));
```



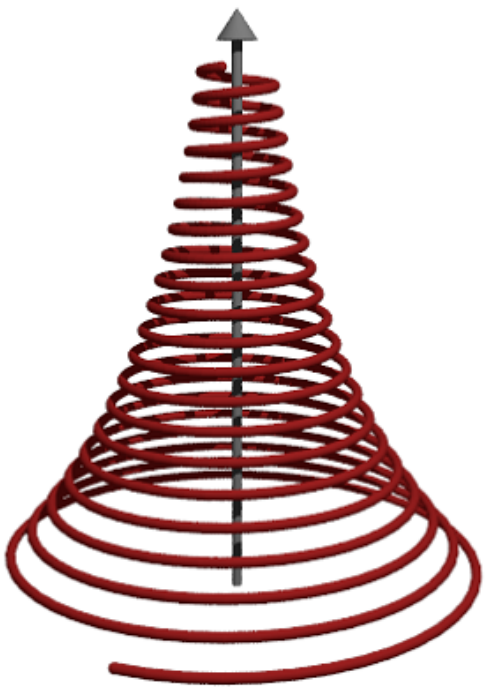
Kita juga dapat menghasilkan kisi dalam 3D.

```
>povstart(zoom=4); ...
>x=-1:0.5:1; r=1-(x+1)^2/6; ...
>t=(0°:30°:360°)'; y=r*cos(t); z=r*sin(t); ...
>writeln(povgrid(x,y,z,d=0.02,dballs=0.05)); ...
>povend();
```



Dengan povgrid(), kurva dimungkinkan.

```
>povstart(center=[0,0,1],zoom=3.6); ...
>t=linspace(0,2,1000); r=exp(-t); ...
>x=cos(2*pi*10*t)*r; y=sin(2*pi*10*t)*r; z=t; ...
>writeln(povgrid(x,y,z,povlook(red))); ...
>writeAxis(0,2,axis=3); ...
>povend();
```



## Objek Povray

Di atas, kita menggunakan pov3d untuk memplot permukaan. Antarmuka povray di Euler juga dapat menghasilkan objek Povray. Objek-objek ini disimpan sebagai string di Euler, dan perlu ditulis ke file Povray.
Kita memulai output dengan povstart().

```
>povstart(zoom=4);
```

Pertama kita mendefinisikan tiga silinder, dan menyimpannya dalam string di Euler.
Fungsi povx() dll. hanya mengembalikan vektor [1,0,0], yang dapat digunakan sebagai gantinya.

```
>c1=povcylinder(-povx,povx,1,povlook(red)); ...
>c2=povcylinder(-povy,povy,1,povlook(green)); ...
>c3=povcylinder(-povz,povz,1,povlook(blue)); ...
```

String berisi kode Povray, yang tidak perlu kita pahami pada saat itu.

```
>c1
```

```
  cylinder { <-1,0,0>, <1,0,0>, 1
   texture { pigment { color rgb <0.564706,0.0627451,0.0627451> }  }
   finish { ambient 0.2 }
   }
```

Seperti yang Anda lihat, kita menambahkan tekstur ke objek dalam tiga warna berbeda.

Ini dilakukan oleh povlook(), yang mengembalikan string dengan kode Povray yang relevan. Kita dapat menggunakan warna Euler default, atau menentukan warna kita sendiri. Kita juga dapat menambahkan transparansi, atau mengubah cahaya sekitar.

```
>povlook(rgb(0.1,0.2,0.3),0.1,0.5)
```

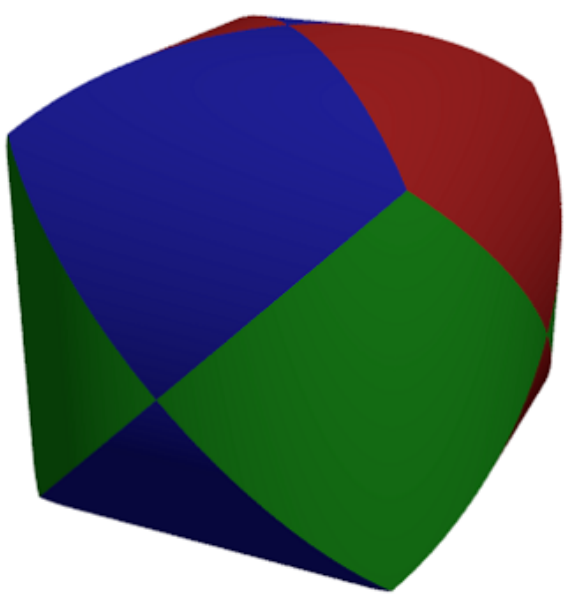
```
   texture { pigment { color rgbf <0.101961,0.2,0.301961,0.1> }  }
   finish { ambient 0.5 }
```

Sekarang kita mendefinisikan objek persimpangan, dan menulis hasilnya ke file.

```
>writeln(povintersection([c1,c2,c3]));
```

Persimpangan tiga silinder sulit untuk divisualisasikan, jika Anda belum pernah melihatnya sebeleumnya.

```
>povend;
```



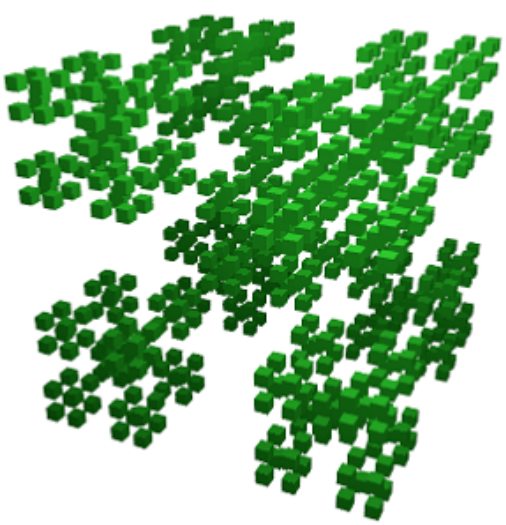
Fungsi berikut menghasilkan pecahan secara rekursif.

Fungsi pertama menunjukkan, bagaimana Euler menangani objek Povray sederhana. Fungsi povbox() mengembalikan string, yang berisi kooordinat kotak, tekstur, dan hasil akhir.

```
>function onebox(x,y,z,d) := povbox([x,y,z],[x+d,y+d,z+d],povlook());
>function fractal (x,y,z,h,n) ...
```

```
  if n==1 then writeln(onebox(x,y,z,h));
  else
    h=h/3;
    fractal(x,y,z,h,n-1);
    fractal(x+2*h,y,z,h,n-1);
    fractal(x,y+2*h,z,h,n-1);
    fractal(x,y,z+2*h,h,n-1);
    fractal(x+2*h,y+2*h,z,h,n-1);
    fractal(x+2*h,y,z+2*h,h,n-1);
    fractal(x,y+2*h,z+2*h,h,n-1);
    fractal(x+2*h,y+2*h,z+2*h,h,n-1);
    fractal(x+h,y+h,z+h,h,n-1);
  endif;
endfunction
```

```
>povstart(fade=10,<shadow);
>fractal(-1,-1,-1,2,4);
>povend();
```



Perbedaan memungkinkan memotong satu objek dari yang lain. Seperti persimpangan, ada bagian dari objek CSG Povray.

```
>povstart(light=[5,-5,5],fade=10);
```

Untuk demonstrasi ini, kita definisikan objek di Povray, alih-alih menggunakan string di Euler. Definisi ditulis ke file segera.
Koordinat kotak -1 berarti [-1,-1,-1].

```
>povdefine("mycube",povbox(-1,1));
```

Kita dapat menggunakan objek di povobject(), yang mengembalikan string seperti biasa.

```
>c1=povobject("mycube",povlook(red));
```

Kita hasilkan kubus kedua, dan memutar dan menskalakannya sedikit.

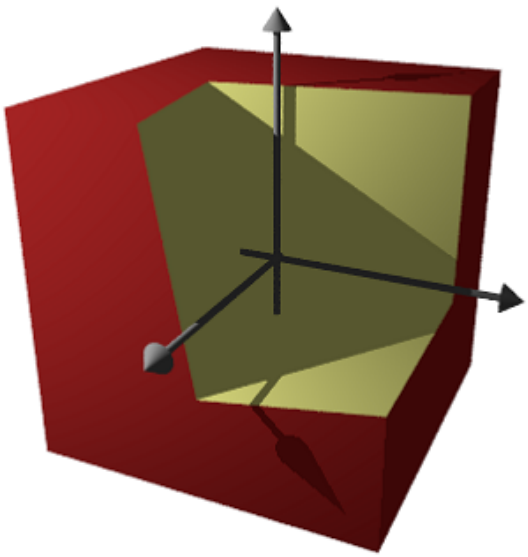
```
>c2=povobject("mycube",povlook(yellow),translate=[1,1,1], ...
>  rotate=xrotate(10°)+yrotate(10°), scale=1.2);
```

Kemudian kita ambil selisih kedua benda tersebut.

```
>writeln(povdifference(c1,c2));
```

Sekarang tambahkan tiga sumbu.

```
>writeAxis(-1.2,1.2,axis=1); ...
>writeAxis(-1.2,1.2,axis=2); ...
>writeAxis(-1.2,1.2,axis=4); ...
>povend();
```



## Fungsi Implisit

Povray dapat memplot himpunan di mana f(x,y,z)=0, seperti parameter implisit di plot3d. Namun, hasilnya terlihat jauh lebih baik.

Sintaks untuk fungsinya sedikit berbeda. Anda tidak dapat menggunakan output dari ekspresi Maxima atau Euler.

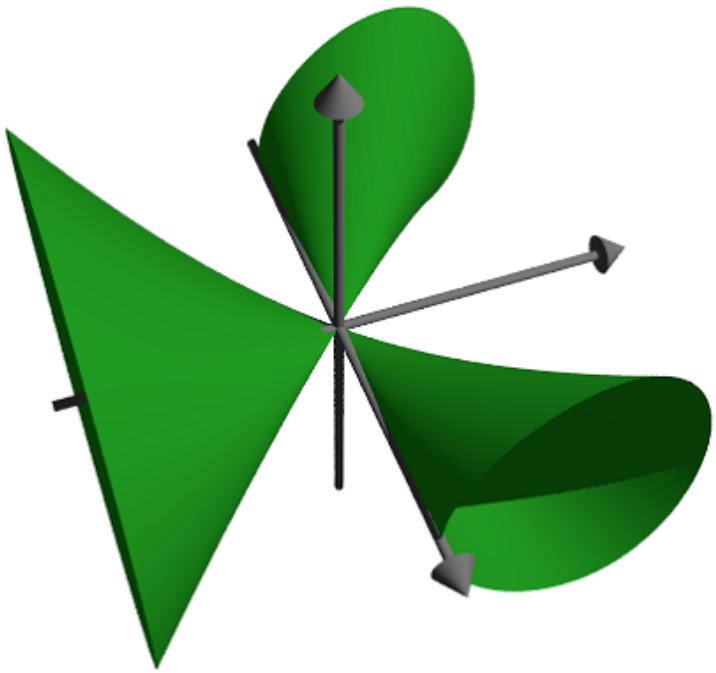
```
>povstart(angle=70°,height=50°,zoom=4);
>c=0.1; d=0.1; ...
>writeln(povsurface("(pow(pow(x,2)+pow(y,2)-pow(c,2),2)+pow(pow(z,2)-1,2))*(pow(pow(y,2)+p
>povend();
```

```
>povstart(angle=25°,height=10°);
>writeln(povsurface("pow(x,2)+pow(y,2)*pow(z,2)-1",povlook(blue),povbox(-2,2,"")));
>povend();
```

```
>povstart(angle=70°,height=50°,zoom=4);
```

Buat permukaan implisit. Perhatikan sintaks yang berbeda dalam ekspresi.

```
>writeln(povsurface("pow(x,2)*y-pow(y,3)-pow(z,2)",povlook(green))); ...
>writeAxes(); ...
>povend();
```



## Objek Mesh

Dalam contoh ini, kita tunjukkan cara membuat objek mesh, dan menggambarnya dengan informasi tambahan.

Kita ingin memaksimalkan xy di bawah kondisi x+y=1 dan menunjukkan sentuhan tangensial dari garis level.

```
>povstart(angle=-10°,center=[0.5,0.5,0.5],zoom=7);
```

Kita tidak dapat menyimpan objek dalam string seperti sebelumnya, karena terlalu besar. Jadi kita mendefinisikan objek dalam file Povray menggunakan declare. Fungsi povtriangle() melakukan ini secara otomatis. Itu dapat menerima vektor normal seperti pov3d().

Berikut ini mendefinisikan objek mesh, dan langsung menulisnya ke dalam file.

```
>x=0:0.02:1; y=x'; z=x*y; vx=-y; vy=-x; vz=1;
>mesh=povtriangles(x,y,z,"",vx,vy,vz);
```

Sekarang kita mendefinisikan dua cakram, yang akan berpotongan dengan permukaan.

```
>cl=povdisc([0.5,0.5,0],[1,1,0],2); ...
>ll=povdisc([0,0,1/4],[0,0,1],2);
```

Tulis permukaan yang dikurangi dua cakram.

```
>writeln(povdifference(mesh,povunion([cl,ll]),povlook(green)));
```

Tulis dua persimpangan.

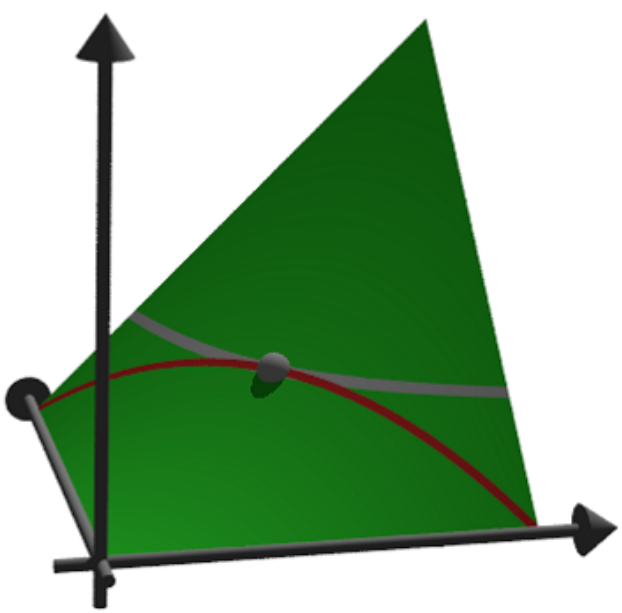
```
>writeln(povintersection([mesh,cl],povlook(red)); ...
>writeln(povintersection([mesh,ll],povlook(gray)));
```

Tulis titik maksimum.

```
>writeln(povpoint([1/2,1/2,1/4],povlook(gray),size=2*defaultpointsize));
```

Tambahkan sumbu dan selesaikan.

```
>writeAxes(0,1,0,1,0,1,d=0.015); ...
>povend();
```



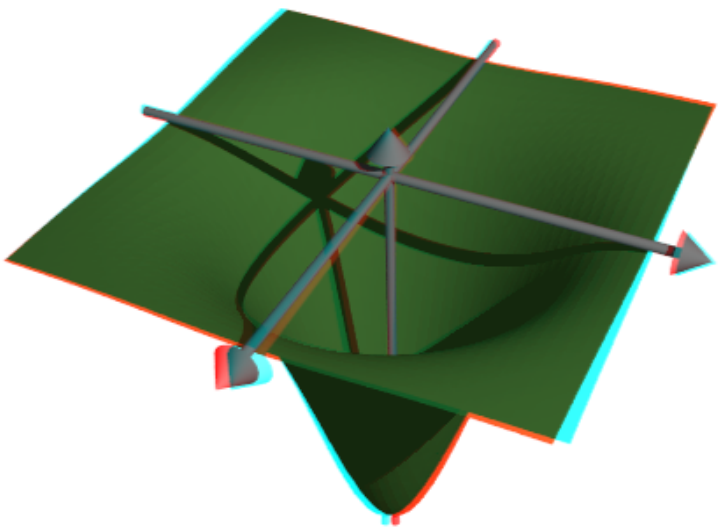
## Anaglyphs di Povray

Untuk menghasilkan anglyph untuk kacamata merah/cyan, Povray harus dijalankan dua kali dari posisi kamera yang berbeda. Ini menghasilkan dua file Povray dan dua file PNG, yang dimuat dengan fungsi loadanaglyph().

Tentu saja, Anda memerlukan kacamata merah/cyan untuk melihat contoh berikut dengan benar.

Fungsi pov3d() memiliki pengalihan sederhana untuk menghasilkan anaglyhph.

```
>pov3d("-exp(-x^2-y^2)/2",r=2,height=45°,>anaglyph, ...
>  center=[0,0,0.5],zoom=3.5);
```



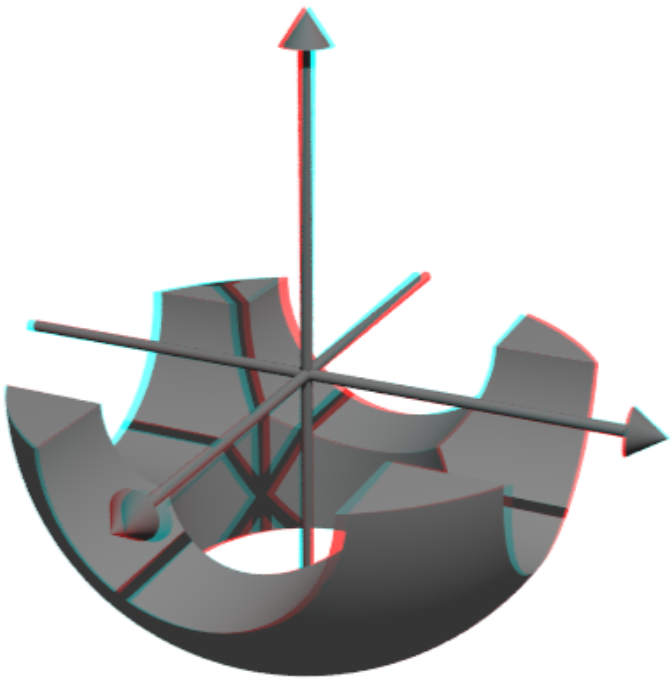
Jika Anda membuat adegan dengan objek, Anda perlu menempatkan generasi adegan ke dalam fungsi, dan menjalankannya dua kali dengan nilai yang berbeda untuk parameter anaglyph.

```
>function myscene ...
```

```
    s=povsphere(povc,1);
    cl=povcylinder(-povz,povz,0.5);
    clx=povobject(cl,rotate=xrotate(90°));
    cly=povobject(cl,rotate=yrotate(90°));
    c=povbox([-1,-1,0],1);
    un=povunion([cl,clx,cly,c]);
    obj=povdifference(s,un,povlook(red));
    writeln(obj);
    writeAxes();
  endfunction
```

Fungsi povanaglyph() melakukan semua ini. Parameternya seperti di povstart() dan povend() digabungkan.

```
>povanaglyph("myscene",zoom=4.5);
```



## Mendefinisikan Objek

Antarmuka Povray Euler sendiri berisi banyak objek. Tapi Anda tidak terbatas pada ini. Anda dapat membuat objek sendiri, yang menggabungkan objek lain, atau objek yang sama sekali baru.

Kita mendemonstrasikan sebuah torus. Perintah Povray untuk ini adalah "torus". Jadi kita kembalikan string dengan perintah ini dan parameternya. Perhatikan bahwa torus selalu berpusat di titik asal.

```
>function povdonat (r1,r2,look="") ...
```

```
    return "torus {"+r1+","+r2+look+"}";
endfunction
```

Inilah torus pertama kita.

```
>t1=povdonat(0.8,0.2)
```

```
torus {0.8,0.2}
```

Mari kita gunakan objek ini untuk membuat torus kedua, diterjemahkan dan diputar.

```
>t2=povobject(t1,rotate=xrotate(90°),translate=[0.8,0,0])
```

```
object { torus {0.8,0.2}
 rotate 90 *x
 translate <0.8,0,0>
 }
```

Sekarang kita menempatkan objek-objek ini ke dalam sebuah adegan. Untuk tampilan, kita menggunakan Shading Pong.

```
>povstart(center=[0.4,0,0],angle=0°,zoom=3.8,aspect=1.5); ...
>writeln(povobject(t1,povlook(green,phong=1))); ...
>writeln(povobject(t2,povlook(green,phong=1))); ...
```

```
>povend();
```

memanggil program Povray. Namun, jika terjadi kesalahan, itu tidak menampilkan kesalahan. Karena itu Anda harus menggunakan

```
>povend(<exit);
```

Jika ada yang tidak berhasil. Ini akan membiarkan jendela Povray terbuka.

```
>povend(h=320,w=480);
```

Berikut adalah contoh yang lebih rumit. Kita pecahkan

$$Ax \le b, \quad x \ge 0, \quad c.x \to \text{Max}.$$

dan menunjukkan titik-titik yang layak dan optimal dalam plot 3D.

```
>A=[10,8,4;5,6,8;6,3,2;9,5,6];
>b=[10,10,10,10]';
>c=[1,1,1];
```

Pertama, mari kita periksa, apakah contoh ini memiliki solusi sama sekali.

```
>x=simplex(A,b,c,>max,>check)'
```

```
  [0,  1,  0.5]
```

Ya, sudah.
Selanjutnya kita definisikan dua objek. Yang pertama adalah

$$a \cdot x \le b$$

```
>function oneplane (a,b,look="") ...
```

```
    return povplane(a,b,look)
  endfunction
```

Kemudian kita definisikan perpotongan semua setengah ruang dan sebuah kubus.

```
>function adm (A, b, r, look="") ...
```

```
    ol=[];
    loop 1 to rows(A); ol=ol|oneplane(A[#],b[#]); end;
    ol=ol|povbox([0,0,0],[r,r,r]);
    return povintersection(ol,look);
  endfunction
```

Kita sekarang dapat merencanakan adegannya.

```
>povstart(angle=120°,center=[0.5,0.5,0.5],zoom=3.5); ...
>writeln(adm(A,b,2,povlook(green,0.4))); ...
>writeAxes(0,1.3,0,1.6,0,1.5); ...
```

Berikut ini adalah lingkaran di sekitar optimal.

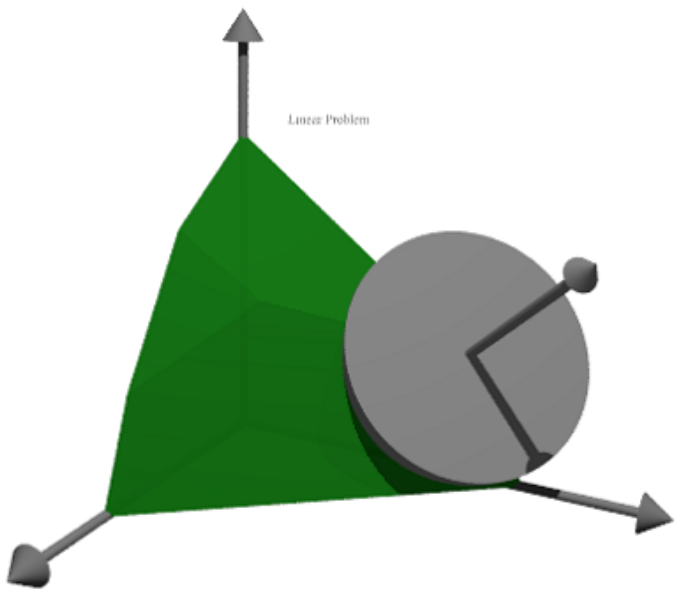
```
>writeln(povintersection([povsphere(x,0.5),povplane(c,c.x')], ...
>  povlook(red,0.9)));
```

Dan kesalahan ke arah yang optimum.

```
>writeln(povarrow(x,c*0.5,povlook(red)));
```

Kita tambahkan teks ke layar. Teks hanyalah objek 3D. Kita perlu menempatkan dan memutarnya menurut pandangan kita.

```
>writeln(povtext("Linear Problem",[0,0.2,1.3],size=0.05,rotate=125°)); ...
>povend();
```




## Contoh Lainnya

Anda dapat menemukan beberapa contoh lagi untuk Povray di Euler dalam file berikut.

See: Examples/Dandelin Spheres
See: Examples/Donat Math
See: Examples/Trefoil Knot
See: Examples/Optimization by Affine Scaling

## Percobaan beberapa contoh lainnya

### Bola Dandelin

Bola Dandelin adalah satu atau dua bola yang bersinggungan baik dengan pesawat dan kerucut yang memotong bidang.

Pertama kita hitung jari-jari bola.
Kita membutuhkan dua lingkaran yang menyentuh dua garis yang membentuk kerucut dan satu garis yang membentuk bidang yang memotong kerucut.

```
>load geometry
```

```
  Numerical and symbolic geometry.
```

```
>g1 &= lineThrough([0,0],[1,a])
```

$$[-a, 1, 0]$$

```
>g2 &= lineThrough([0,0],[-1,a])
```

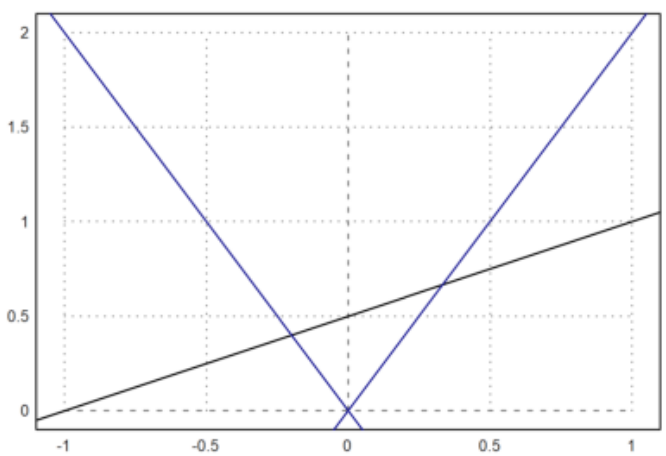
$$[-a, -1, 0]$$

```
>g &= lineThrough([-1,0],[1,1])
```

$$[-1, 2, 1]$$

Buat plot untuk mengecek perpotongan garis.

```
>setPlotRange(-1,1,0,2);
>color(black); plotLine(g(),"")
>a:=2; color(blue); plotLine(g1(),""), plotLine(g2(),""):
```




Sekarang kita ambil titik umum sumbu-y.

```
>P &= [0,u]
```

$$[0,\ u]$$

Hitung jarak ke g1.

```
>d1 &= distance(P,projectToLine(P,g1))
```

$$\sqrt{\left(\frac{a^2\ u}{a^2+1}-u\right)^2+\frac{a^2\ u^2}{(a^2+1)^2}}$$

Hitung jarak ke g.

```
>d &= distance(P,projectToLine(P,g))
```

$$\sqrt{\left(\frac{u+2}{5}-u\right)^2+\frac{(2\ u-1)^2}{25}}$$

Dan temukan pusat kedua lingkaran yang jaraknya sama.

```
>sol &= solve(d1^2=d^2,u)
```

$$\left[u=\frac{-\sqrt{5}\ \sqrt{a^2+1}+2\ a^2+2}{4\ a^2-1},\ u=\frac{\sqrt{5}\ \sqrt{a^2+1}+2\ a^2+2}{4\ a^2-1}\right]$$

Ada dua solusi.
Kita evaluasi solusi simbolik, dan menemukan kedua pusat, dan kedua jarak.

```
>u := sol()
```

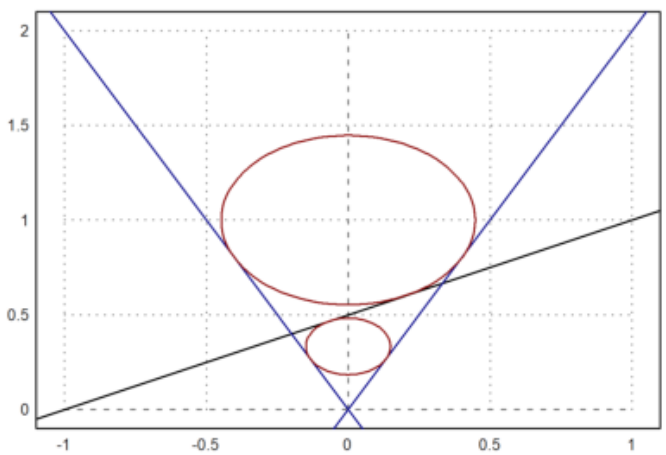
```
  [0.33333,  1]
```

```
>dd := d()
```

```
  [0.14907,  0.44721]
```

Plot lingkaran ke dalam gambar.

```
>color(red);
>plotCircle(circleWithCenter([0,u[1]],dd[1]),"");
>plotCircle(circleWithCenter([0,u[2]],dd[2]),"");
>insimg;
```




Selanjutnya kita buat dengan Povray.

```
>load povray;
>defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

```
  C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe
```

Atur adegan dengan tepat.

```
>povstart(zoom=11,center=[0,0,0.5],height=10°,angle=140°);
```

Selanjutnya kita buat dua bola.

```
>writeln(povsphere([0,0,u[1]],dd[1],povlook(red)));
>writeln(povsphere([0,0,u[2]],dd[2],povlook(red)));
```

Buat kerucut transparan.

```
>writeln(povcone([0,0,0],0,[0,0,a],1,povlook(lightgray,1)));
```

Kita hasilkan batas bidang kerucut.

```
>gp=g();
>pc=povcone([0,0,0],0,[0,0,a],1,"");
>vp=[gp[1],0,gp[2]]; dp=gp[3];
>writeln(povplane(vp,dp,povlook(blue,0.5),pc));
```

Sekarang kita menghasilkan dua titik pada lingkaran, di mana bola menyentuh kerucut.

```
>function turnz(v) := return [-v[2],v[1],v[3]]
>P1=projectToLine([0,u[1]],g1()); P1=turnz([P1[1],0,P1[2]]);
>writeln(povpoint(P1,povlook(yellow)));
>P2=projectToLine([0,u[2]],g1()); P2=turnz([P2[1],0,P2[2]]);
>writeln(povpoint(P2,povlook(yellow)));
```

Kemudian kita hasilkan dua titik di mana bola menyentuh bidang. Ini adalah fokus dari 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 hitung P1P2 bidang.

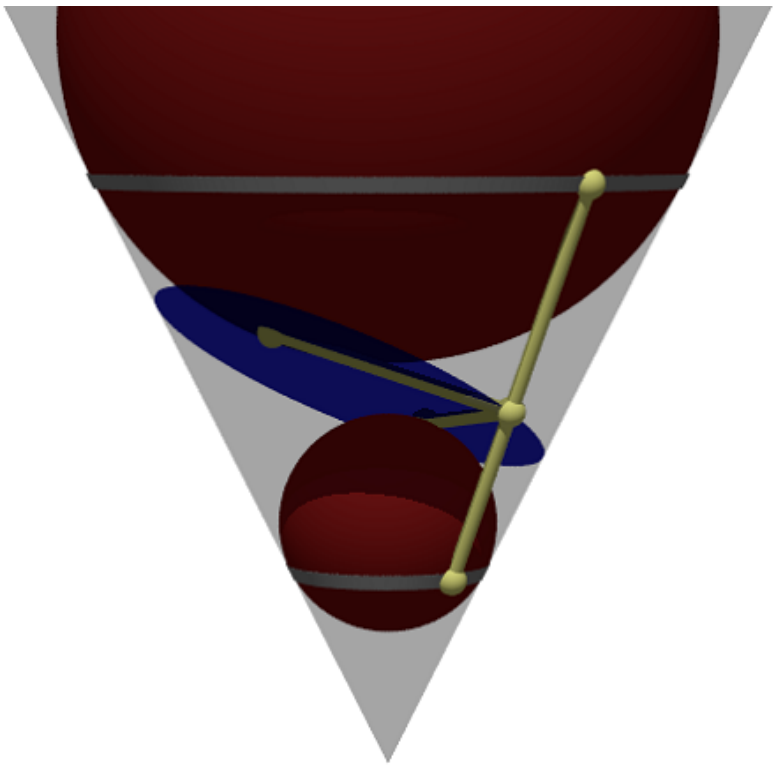
```
>t1=scalp(vp,P1)-dp; t2=scalp(vp,P2)-dp; P5=P1+t1/(t1-t2)*(P2-P1);
>writeln(povpoint(P5,povlook(yellow)));
```

Kita hubungkan titik-titik dengan segmen garis.

```
>writeln(povsegment(P1,P2,povlook(yellow)));
>writeln(povsegment(P5,P3,povlook(yellow)));
>writeln(povsegment(P5,P4,povlook(yellow)));
```

Sekarang kita menghasilkan pita abu-abu, di mana bola menyentuh kerucut.

```
>pcw=povcone([0,0,0],0,[0,0,a],1.01);
>pc1=povcylinder([0,0,P1[3]-defaultpointsize/2],[0,0,P1[3]+defaultpointsize/2],1);
>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)));
>povend();
```



## Donat Matematika

Pertama kita buat donat yang sangat bagus. Kita gunakan Povray untuk itu dengan Shading Phong.

```
>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=0,height=40°);
```

Fungsi berikut untuk membuat donat.

```
>function povdonat (r1,r2,look="") := "torus {"+r1+","+r2+look+"}";
>writeln(povobject(povdonat(1,0.5),povlook(red,>phong),xrotate(90°)));
>povend();
```

Torus adalah gambar silinder di bawah pemetaan berikut.

```
>function phi(x,t,z) &= [x*cos(t),x*sin(t),z]
```

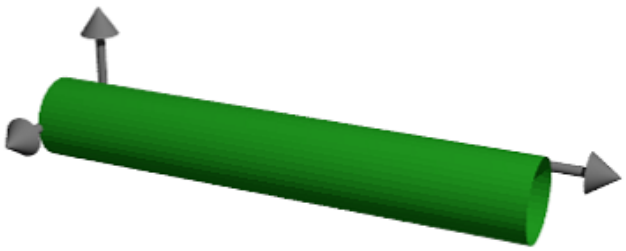
```
                        [cos(t) x, sin(t) x, z]
```

Untuk Euler, kita membutuhkan koordinat sumbu-x, sumbu-y, dan sumbu-z dari pemetaan ini.

```
>function phix(x,t,z) &= phi(x,t,z)[1];
>function phiy(x,t,z) &= phi(x,t,z)[2];
>function phiz(x,t,z) &= phi(x,t,z)[3];
```

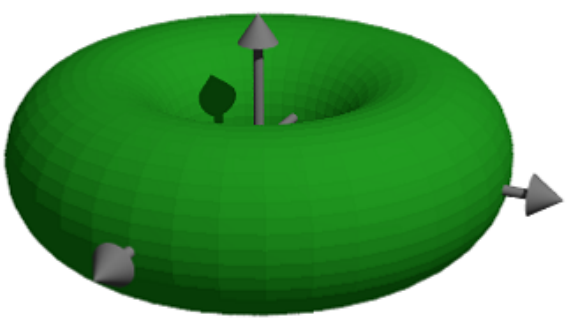
Sekarang kita buat sebuah silinder dengan lingkaran yang berpusat di x=R, y=0, z=0 dengan tinggi 2phi.

```
>s=linspace(0,2pi,40); t=s';
>R=1; r=0.5;
>x=R+r*cos(s); y=t; z=r*sin(s);
>pov3d(x,y,z,zoom=4);
```



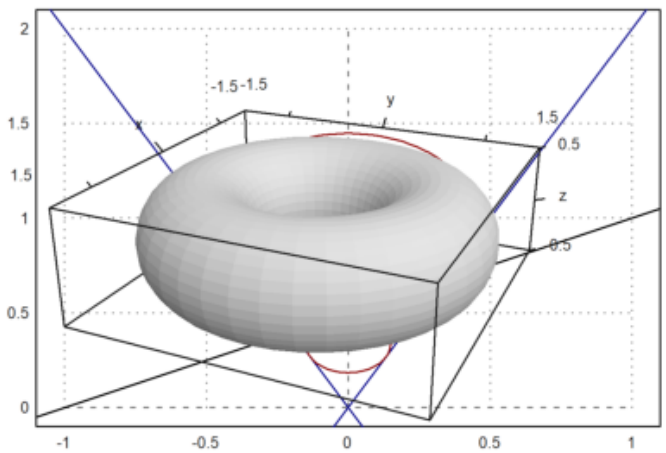
Transformasi phi memetakan silinder ini ke donat. Setiap titik dipetakan dengan phi.

```
>X=phix(x,y,z); Y=phiy(x,y,z); Z=phiz(x,y,z);
>pov3d(X,Y,Z,zoom=3.8);
```



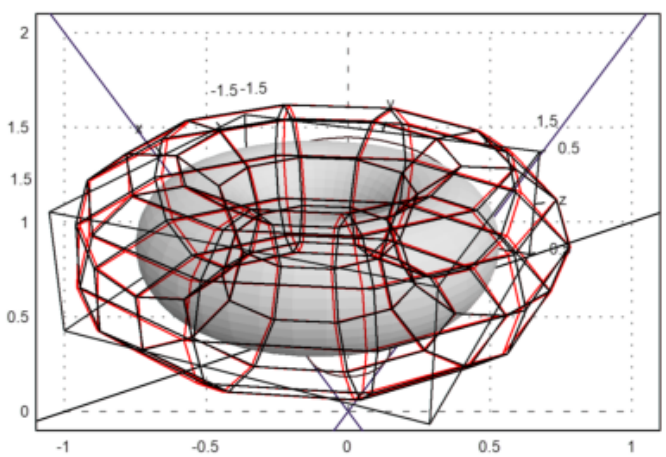
Berikut adalah gambar yang sama di Euler. Kita ambil sumbu-Z, karena bayangan Euler default didasarkan pada sisi permukaan yang benar.

```
>plot3d(X,Y,-Z,>hue,zoom=3.6):
```




Gambar berikut adalah Anagylph untuk kacamata merah/cyan yang menunjukkan kerangka kawat torus dengan detail yang lebih sedikit:

```
>s=linspace(0,2pi,10); t=s'; ...
>R=1; r=0.5; ...
>X=phix(R+r*cos(s),t,r*sin(s)); ...
>Y=phiy(R+r*cos(s),t,r*sin(s)); ...
>Z=phiz(R+r*cos(s),t,r*sin(s)); ...
>plot3d(X,Y,Z,>wire,zoom=5,<frame,>anaglyph):
```




Selanjutnya kita akan memotong donat dengan bidang y=c. Perintah berikut memanggil Povray untuk menghasilkan versi irisan dari donat kita.

```
>povstart(angle=0°,height=40°,light=[0,-0.1,1],zoom=3.6);
```

Menggambar Donat terlebih dahulu.

```
>donat=povobject(povdonat(1,0.5),xrotate(90°));
```

Sekarang kita mendefinisikan sebuah fungsi, yang menghasilkan silinder untuk pemotongan.

```
>function cut(y1,y2) := povcylinder([0,y1,0],[0,y2,0],2);
```

Sekarang definisikan vektor untuk pemotongan.

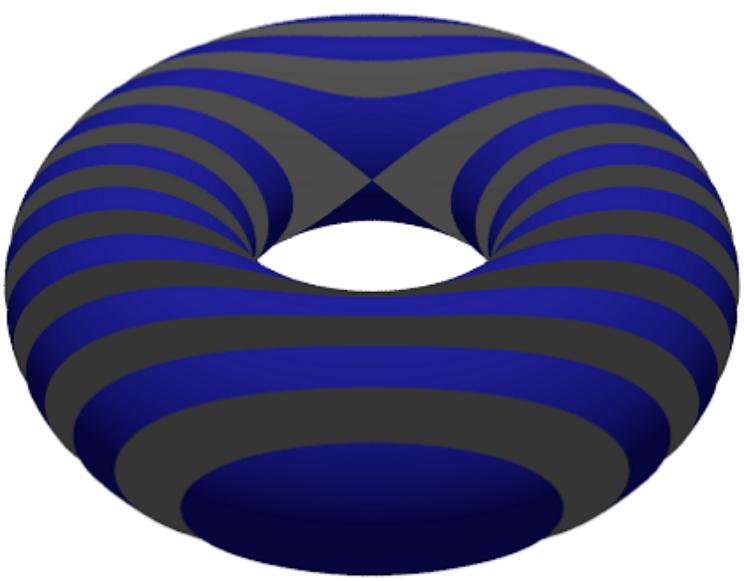
```
>a=linspace(-1.5,1.5,21);
```

Silinder pemotong disimpan dalam vektor string.

```
>s=[]; for i=1 to (length(a)-1)/2; s=s|cut(a[2*i],a[2*i+1]); end;
```

Bentuk penyatuan potongan.

```
>cuts=povunion(s);
>writeln(povdifference(donat,cuts,povlook(blue)));
>povend();
```



## Beberapa contoh penyelesaian masalah menggambar kurva/plot 3D

1. Gambarlah sebuah grafik dari fungsi-fungsi berikut:

a.

$$f(x,y) = 9x^2 + 4y^2 + 9z^2 = 36$$

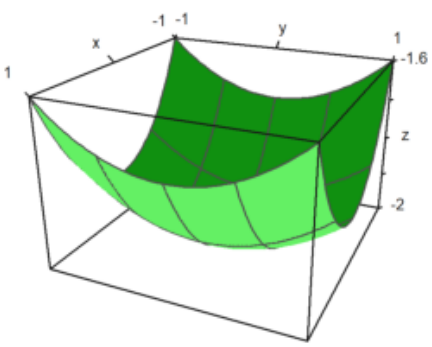
b.

$$g(x,y) = cos(y) * sin(x * y)$$

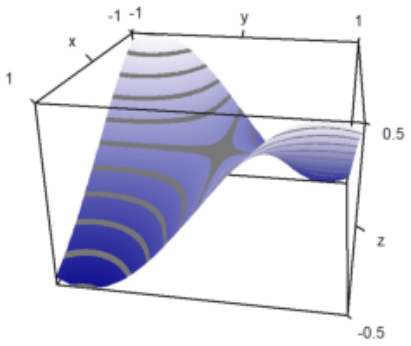
c.

$$h(x,y) = \sqrt{10x^2 + 2y^2}$$

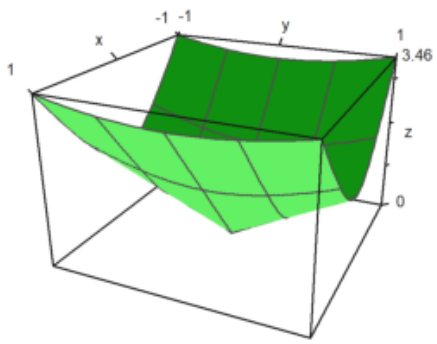
```
>function f(x,y) &= -sqrt(4-x^2-(4*y^2)/9);
>plot3d("f",grid=4):
```




```
>function g(x,y) &= cos(y)*sin(x*y);
>plot3d("g",angle=100°,>contour,color=blue):
```




```
>function h(x,y) &= sqrt(10*x^2+2*y^2);
>plot3d("h",grid=4):
```




---

2. Sketsakan 4 fungsi tersebut pada R3!

a.

$$\frac{x^2}{y}$$

b.

$$e^{-x^2-y^2}$$

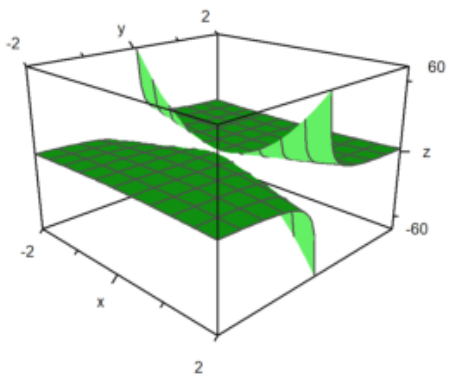
c.

$$3 - x^2 * y^2$$

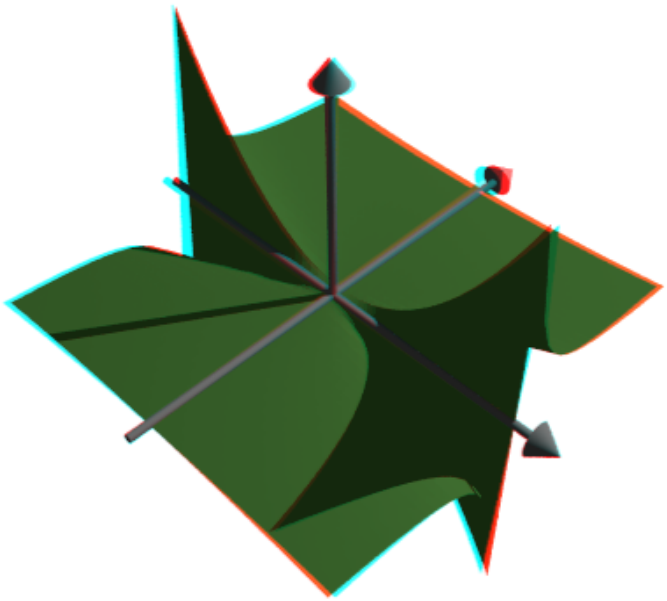
d.

$$-|xy|$$

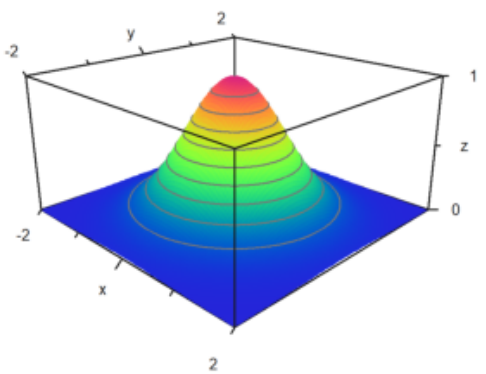
```
>function f(x,y) &= (x^2/y);
>plot3d("f",-2,2,-2,2):
```



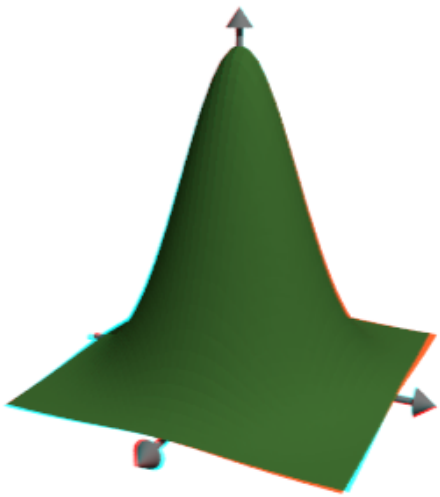

```
>pov3d("x^2*y^-1",r=2,height=45°,>anaglyph, ...
>  center=[0,0,0.5],zoom=3.5);
```



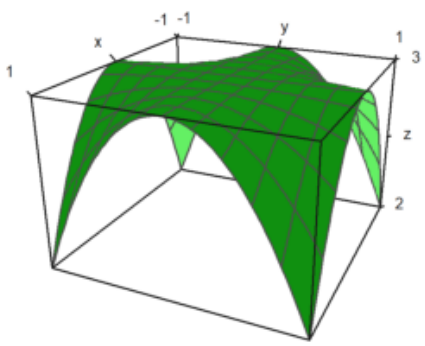
```
>function g(x,y) &= E^(-x^2-y^2);
>plot3d("g",r=2,n=100,level="thin", ...
>>contour,>spectral,fscale=1,scale=1.1,angle=45°,height=20°):
```



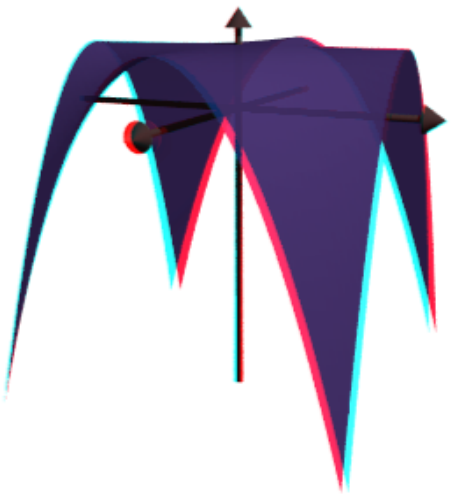

```
>pov3d("g",r=2,height=15°,>anaglyph,center=[0,0,0.5],zoom=2.5);
```



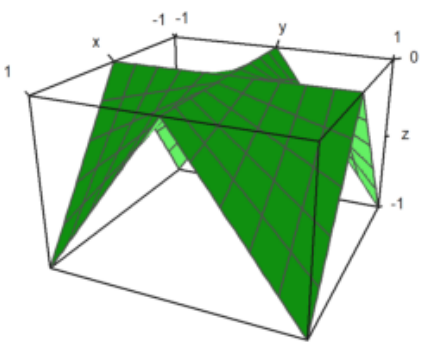
```
>function k(x,y) &= 3-x^2*y^2;
>plot3d("k"):
```



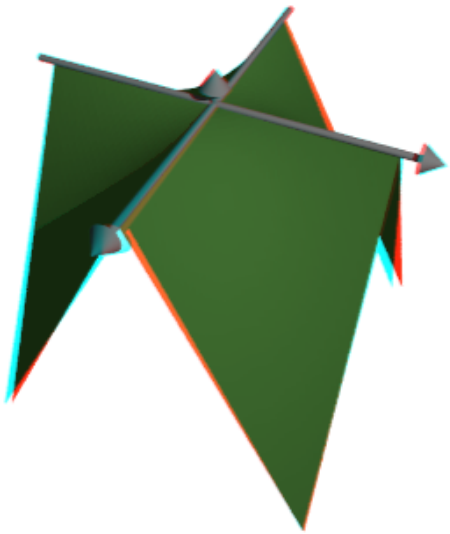

```
>pov3d("k",axiscolor=red,look=povlook(blue,0.2),r=2,height=15°,>anaglyph,center=[0,0,0.5],
```



```
>function h(x,y) &= -abs(x*y);
>plot3d("h"):
```




```
>pov3d("h",r=2,height=45°,>anaglyph,center=[0,0,0.5],zoom=2.5)
```



---

3. Sketsa kurva ketinggian z=k, untuk k=-2,-1,0,1,2 pada permukaan z=y-sin(x)!

jawab:
k=-2
ketika x=0, maka y=-2
ketika y=0, maka sin(x)=2

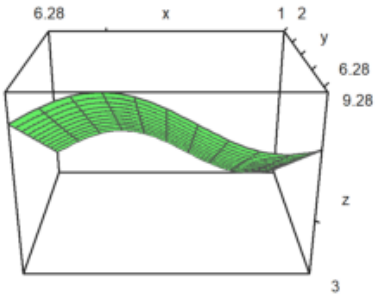
k=-1
ketika x=0, maka y=-1
ketika y=0, maka sin(x)=1

k=0
ketika x=0, maka y=0
ketika y=0, maka sin(x)=0

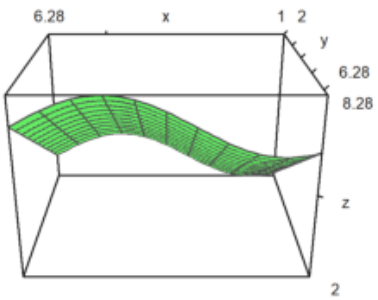
k=1
ketika x=0, maka y=1
ketika y=0, maka sin(x)=-1

k=2
ketika x=0, maka y=2
ketika y=0, maka sin(x)=-2

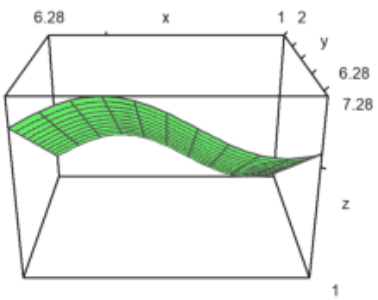
```
>plot3d("y-sin(x)+2",1,2*pi,2,2*pi, angle=180°):
```



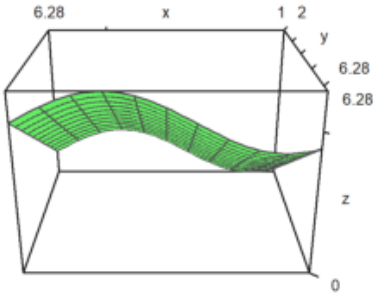

```
>plot3d("y-sin(x)+1",1,2*pi,2,2*pi, angle=180°):
```



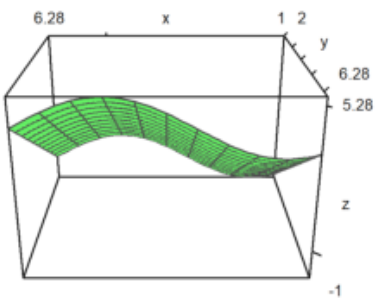

```
>plot3d("y-sin(x)",1,2*pi,2,2*pi, angle=180°):
```



```
>plot3d("y-sin(x)-1",1,2*pi,2,2*pi, angle=180°):
```




```
>plot3d("y-sin(x)-2",1,2*pi,2,2*pi, angle=180°):
```




---

4.Sketsakan grafik fungsi berikut:

$$f(x,y) = x^3y + 3xy^2$$

Lalu, Tentukan vektor gradien fungsinya , kemudian
tentukan persamaan bidang singgung di titik (2,-2)

jawab:
Vektor gradien fungsi f(x,y)=

$$(fx(x,y))i + (fy(x,y))j$$

$$(3x^2y + 3y^2)i + (x^3 + 6xy)j$$

Vektor gradien fungsi f(2,-2)=

$$-12i + (-16)j$$

Persamaan bidang singgung di (2,-2)

$$Z = f(2,-2) + (-12i - 16j).((x-2)i + (y+2)j)$$

$$= 8 + 12x + 24 - 16y - 32$$

$$Z = -12x - 16y$$

```
>plot3d("x^3*y+3*x*y^2", angle=250°):
```