

23030130028_Hanifatunnisa Amatulloh

Nama = Hanifatunnisa Amatulloh
NIM = 23030130028
Prodi/Kelas = Pendidikan Matematika/A
Mata Kuliah = Aplikasi Komputer

Menggambar Plot 3D dengan EMT

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

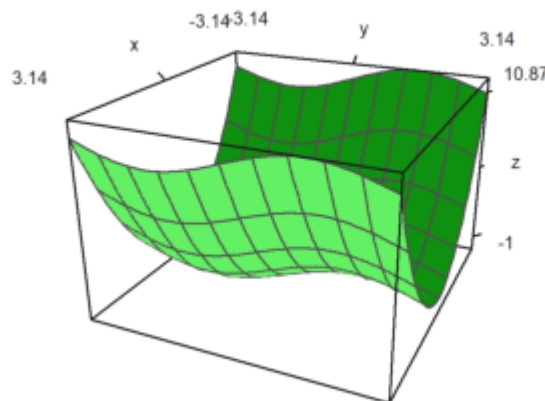
Euler menggambar fungsi tersebut menggunakan algoritma pengurutan untuk menyembunyikan bagian di latar belakang. Secara umum, Euler menggunakan proyeksi pusat. Standarnya adalah dari kuadran x-y positif menuju titik asal $x=y=z=0$, tetapi sudut $=0^\circ$ terlihat dari arah sumbu y. Sudut pandang dan ketinggian dapat diubah.

Euler dapat merencanakan

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

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

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



Fungsi dua Variabel

Untuk grafik fungsi, gunakan

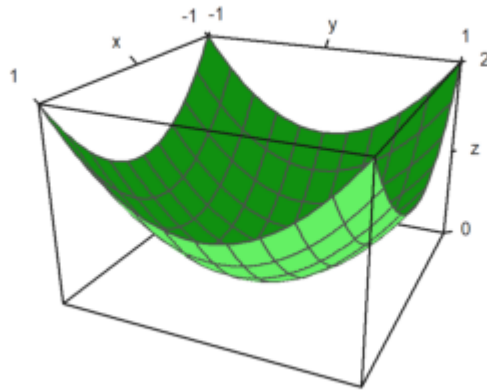
- ekspresi sederhana dalam x dan y,
- nama fungsi dari dua variabel
- atau matriks data.

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

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

Kami menggunakan default $n=40$ dan $grid=10$.

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

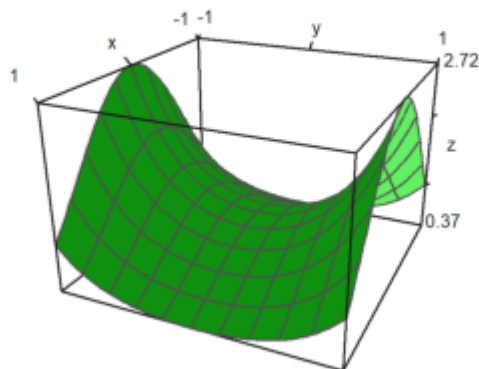


Interaksi pengguna dimungkinkan dengan >parameter pengguna. Pengguna dapat menekan tombol berikut.

- kiri, kanan, atas, bawah: putar sudut pandang
- +,-: memperbesar atau memperkecil
- a: menghasilkan anaglyph (lihat di bawah)
- l: beralih memutar sumber cahaya (lihat di bawah)
- spasi: reset ke default
- kembali: akhiri interaksi

```
>plot3d("exp(-x^2+y^2)",>user, ...  
  title="Turn with the vector keys (press return to finish)");
```

Turn with the vector keys (press return to finish)



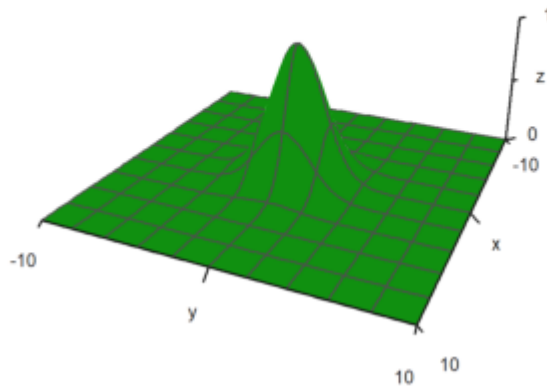
Rentang plot untuk fungsi dapat ditentukan dengan

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

Ada beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.

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

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



Tampilan dapat diubah dengan berbagai cara.

- jarak: jarak pandang ke plot.
- zoom: nilai zoom.
- sudut: sudut terhadap sumbu y negatif dalam radian.
- tinggi: ketinggian tampilan dalam radian.

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

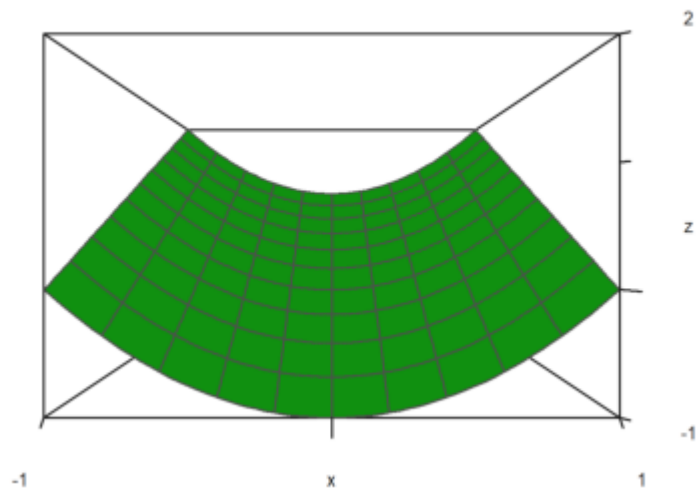
```
>view
```

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

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

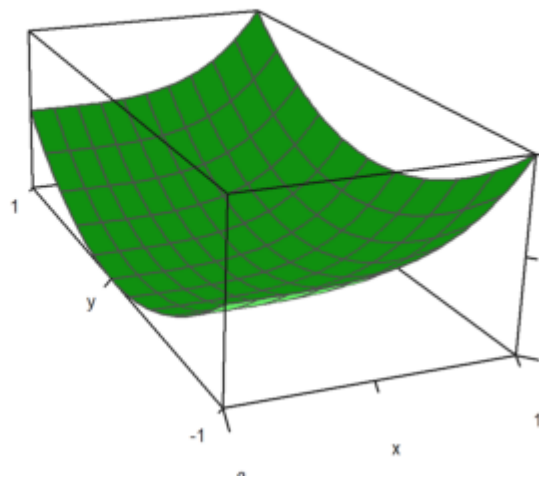
Dalam contoh berikut, sudut=0 dan tinggi=0 terlihat dari sumbu y negatif. Label sumbu untuk y disembunyikan dalam kasus ini.

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



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

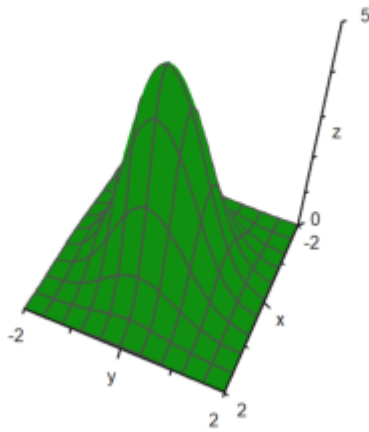
```
>plot3d("x^4+y^2",a=0,b=1,c=-1,d=1,angle=-20°,height=20°, ...
  center=[0.4,0,0],zoom=5):
```



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

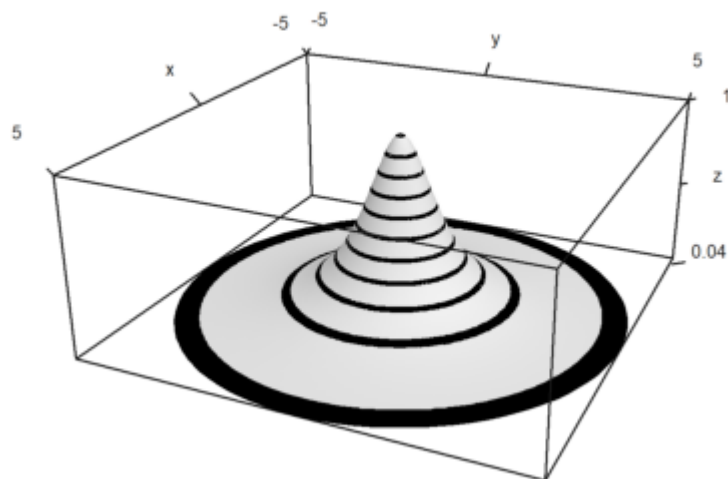
Jika Anda mematikannya dengan `scale=false`, Anda perlu berhati-hati, bahwa plot masih cocok dengan jendela plot, dengan mengubah jarak pandang atau zoom, dan memindahkan pusat.

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

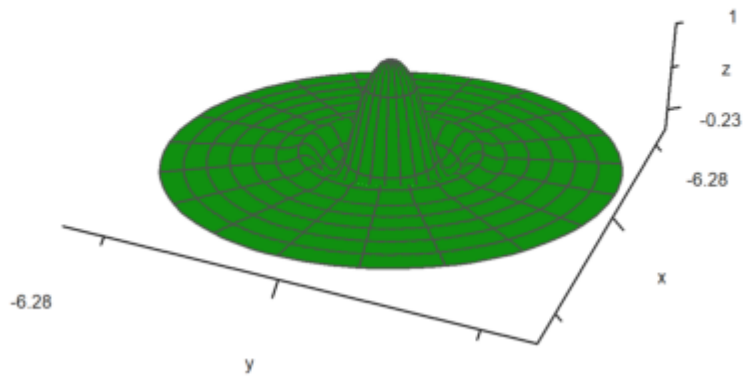


Sebuah plot kutub juga tersedia. Parameter `polar=true` menggambar plot polar. Fungsi tersebut harus tetap merupakan fungsi dari x dan y . Parameter `fscale` menskalakan fungsi dengan skala sendiri. Jika tidak, fungsi diskalakan agar sesuai dengan kubus.

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



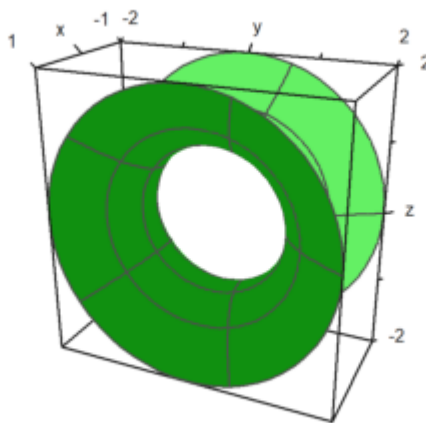
```
>function f(r) := exp(-r/2)*cos(r); ...
  plot3d("f(x^2+y^2)",>polar,scale=[1,1,0.4],r=2pi,frame=3,zoom=4):
```



Rotasi parameter memutar fungsi dalam x di sekitar sumbu x.

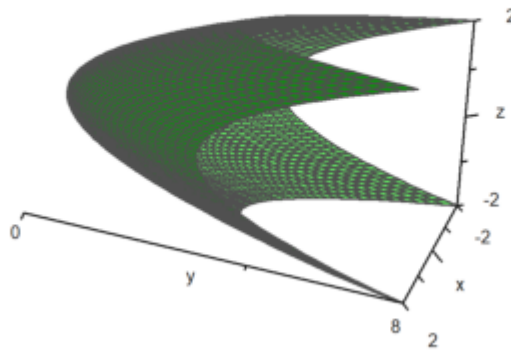
- rotate=1: Menggunakan sumbu x
- rotate=2: Menggunakan sumbu z

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



Berikut adalah plot dengan tiga fungsi.

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

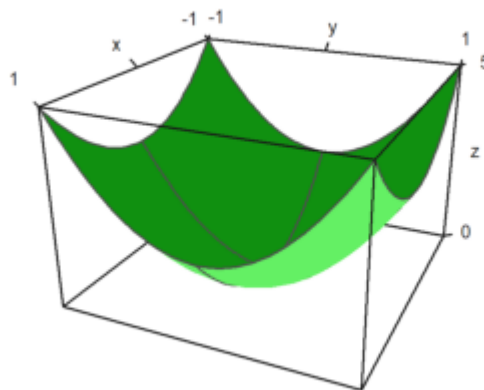


Contoh soal dan pembahasan fungsi dua variabel

Gambarlah grafik dari fungsi berikut

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

```
>plot3d("2*x^2+3*y^2",n=40,grid=2):
```



Gambar di atas menampilkan grafik fungsi dengan $n=40$ dan $grid=2$.

- n = jumlah interval kisi-kisi, jumlah n default=40
- $grid$ = jumlah garis kisi di setiap arah, jumlah $grid$ default=10

Penjelasan:
misalkan

$$z = 2x^2 + 3y^2$$

$$z = \frac{x^2}{\frac{1}{2}} + \frac{y^2}{\frac{1}{3}}$$

(yang dikenal sebagai persamaan sebuah paraboloida eliptik)
dan perhatikan bahwa

$$z \geq 0$$

cari jejak pada bidang koordinat
-bidang XOY(z=0):

$$\frac{x^2}{\frac{1}{2}} + \frac{y^2}{\frac{1}{3}} = 0$$

jika z=0 maka x^2 dan y^2 juga harus 0, maka diperoleh titik (0,0,0)

-bidang YOZ(x=0)

$$\frac{y^2}{\frac{1}{3}} = z$$

$$y^2 = \frac{1}{3}z$$

(berupa parabola searah sumbu z dan titik puncaknya (0,0))

-bidang XOZ(y=0)

$$\frac{x^2}{\frac{1}{2}} = z$$

$$x^2 = \frac{1}{2}z$$

(berupa parabola searah sumbu z dan titik puncaknya (0,0))

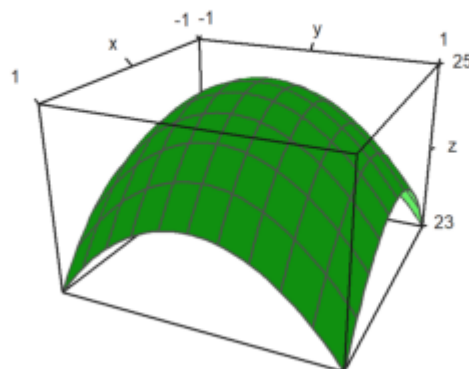
Soal 2

Gambarlah grafik dari fungsi berikut

$$f(x, y) = 25 - x^2 - y^2$$

```
>plot3d("25-x^2-y^2",>user, ...  
title= "Turn with the vector keys (press return to finish)");
```

Turn with the vector keys (press return to finish)



Plot Kontur

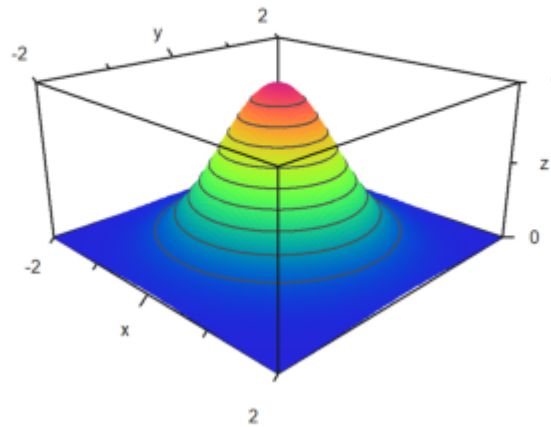
Untuk plot, Euler menambahkan garis grid. Sebagai gantinya dimungkinkan untuk menggunakan garis level dan rona satu warna atau rona berwarna spektral. Euler dapat menggambar tinggi fungsi pada plot dengan bayangan. Di semua plot 3D, Euler dapat menghasilkan anaglyph merah/sian.

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

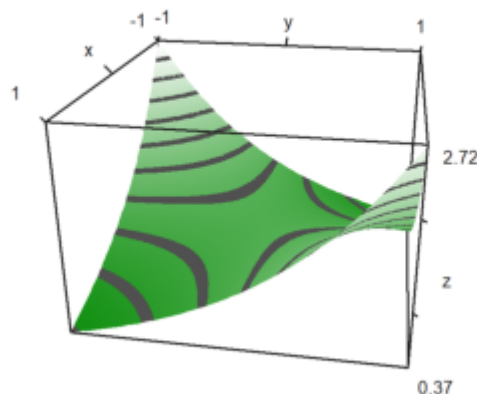
Standarnya adalah level="auto", yang menghitung beberapa garis level secara otomatis. Seperti yang Anda lihat di plot, level sebenarnya adalah rentang level.

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

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



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

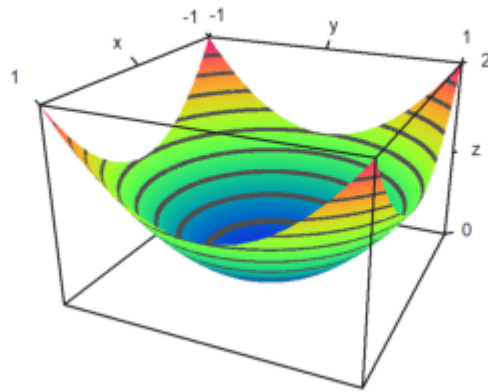


Bayangan default menggunakan warna abu-abu. Tetapi rentang warna spektral juga tersedia.

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

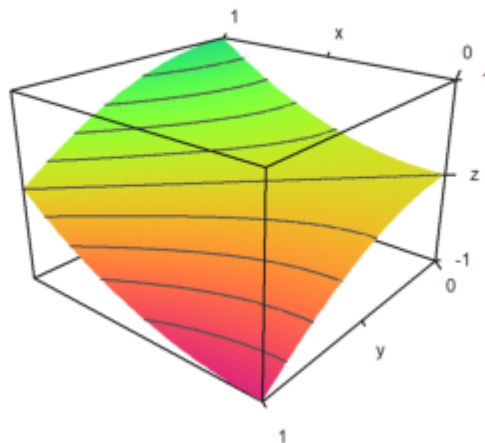
Untuk plot berikut, kami menggunakan skema spektral default dan menambah jumlah titik untuk mendapatkan tampilan yang sangat halus.

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



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

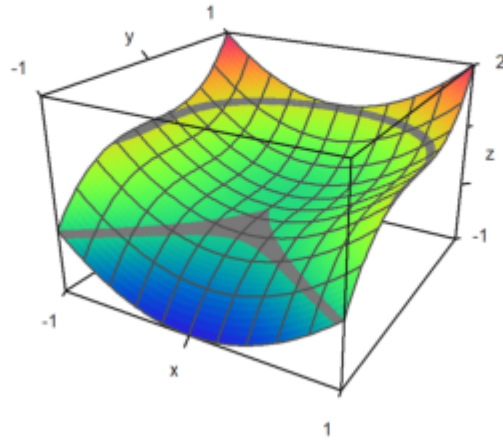
```
>plot3d("x^2-y^2",0,1,0,1,angle=220°,level=-1:0.2:1,color=redgreen):
```



Dalam plot berikut, kami menggunakan dua pita level yang sangat luas dari -0,1 hingga 1, dan dari 0,9 hingga 1. Ini dimasukkan sebagai matriks dengan batas level sebagai kolom.

Selain itu, kami melapisi kisi dengan 10 interval di setiap arah.

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

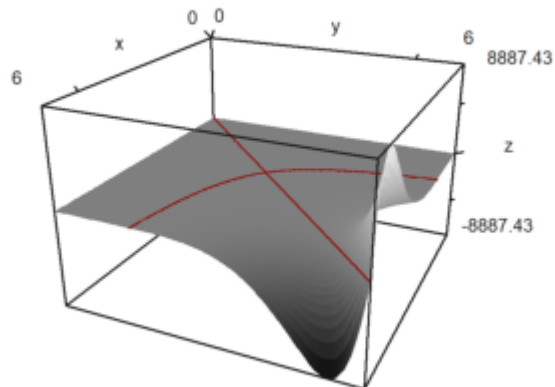


Dalam contoh berikut, kami memplot himpunan, di mana

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

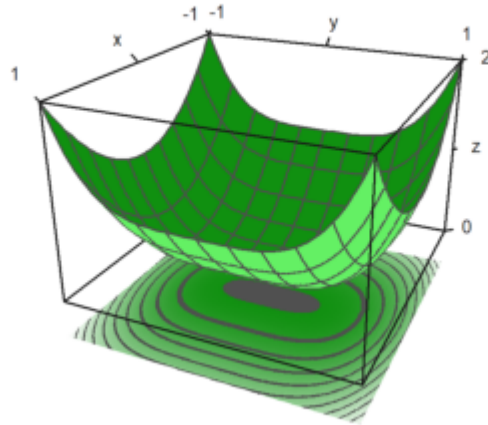
Kami menggunakan satu garis tipis untuk garis level.

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



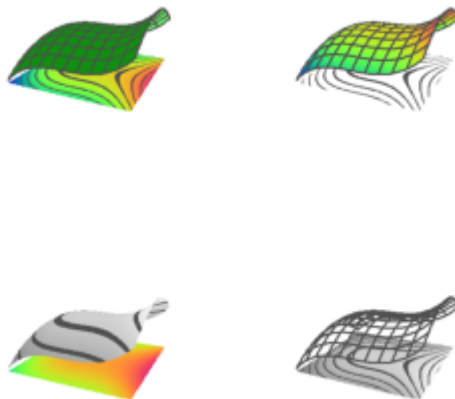
Dimungkinkan untuk menunjukkan bidang kontur di bawah plot. Warna dan jarak ke plot dapat ditentukan.

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



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

```
>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):
```

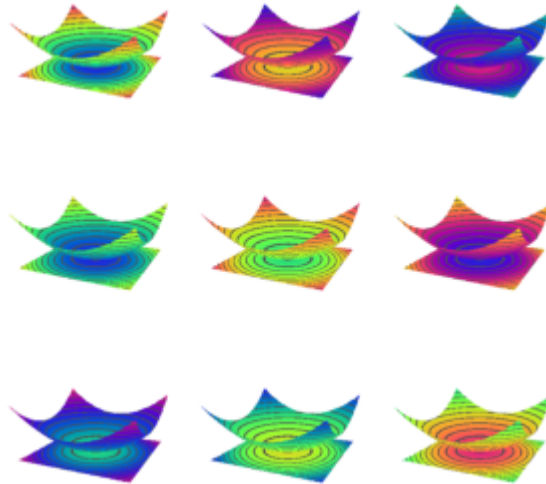


Ada beberapa skema spektral lainnya, bernomor dari 1 hingga 9. Tetapi Anda juga dapat menggunakan warna=nilai, di mana nilai

- 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):
```



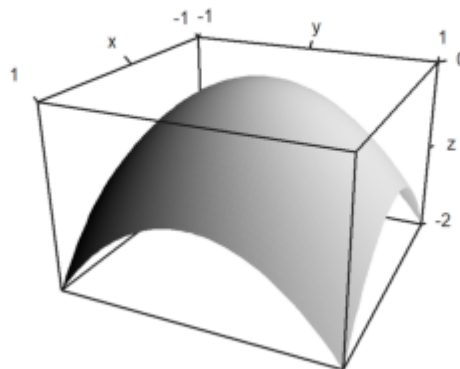
Sumber cahaya dapat diubah dengan `l` dan tombol cursor selama interaksi pengguna. Itu juga dapat diatur dengan parameter.

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

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

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

Press l and cursor keys (return to exit)



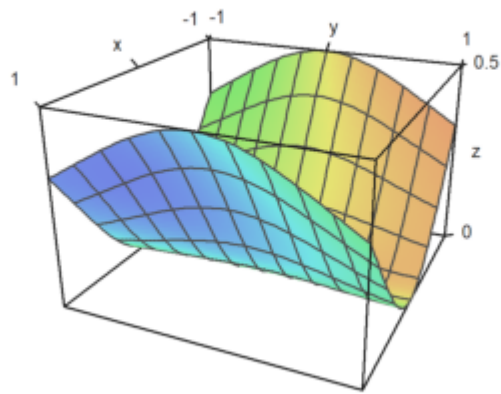
Parameter warna mengubah warna permukaan. Warna garis level juga dapat diubah.

```
>plot3d("-x^2-y^2",color=rgb(0.2,0.2,0),hue=true,frame=false, ...
zoom=3,contourcolor=red,level=-2:0.1:1,d1=0.01):
```



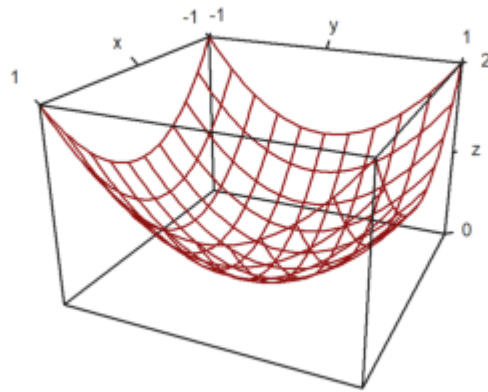
Warna 0 memberikan efek pelangi khusus.

```
>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):
```



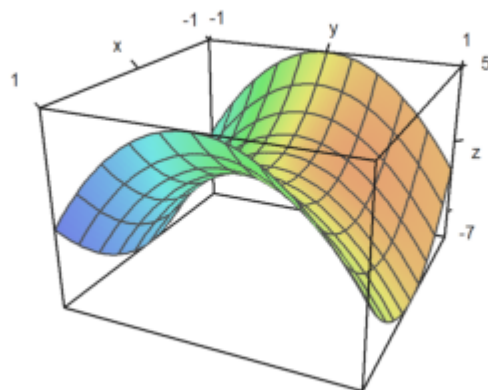
Contoh soal plot kontur

Gambarlah grafik fungsi berikut

$$f(x, y) = 5x^2 - 7y^2$$

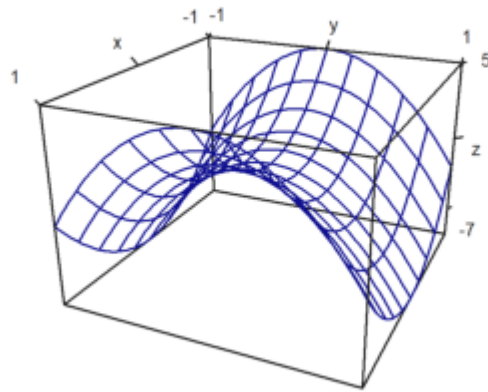
dengan warna pelangi

```
>plot3d("5x^2-7y^2",color=0,hue=true,grid=10):
```



dengan permukaan transparan

```
>plot3d("5x^2-7y^2",>transparent,grid=10,wirecolor=blue):
```



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

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

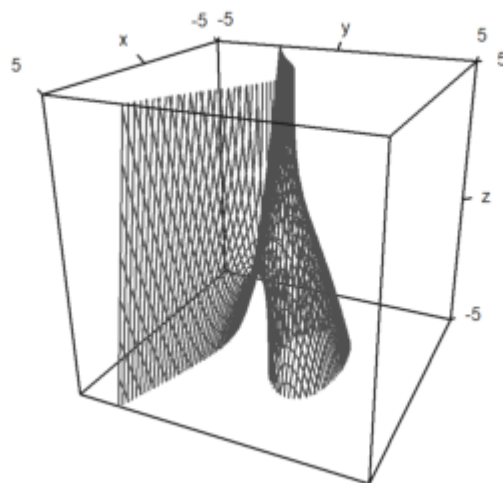
dapat divisualisasikan dalam potongan sejajar dengan bidang x-y-, x-z- dan y-z.

- implicit=1: memotong sejajar bidang y-z
- implicit=2: memotong sejajar bidang x-z
- implicit=4: memotong sejajar bidang x-y

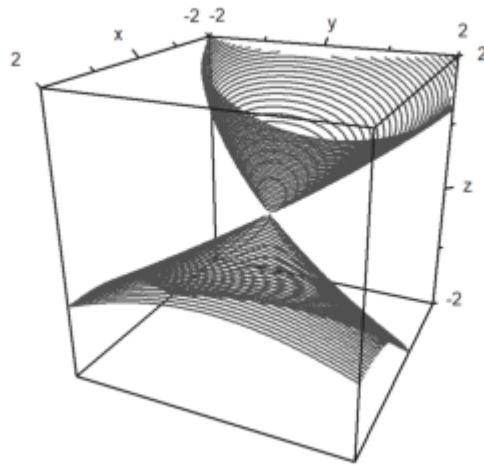
Tambahkan nilai-nilai ini, jika Anda suka. Dalam contoh kita plot

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

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



```
>plot3d("x^2+y^2+4*x*z+z^3",>implicit,r=2,zoom=2.5):
```

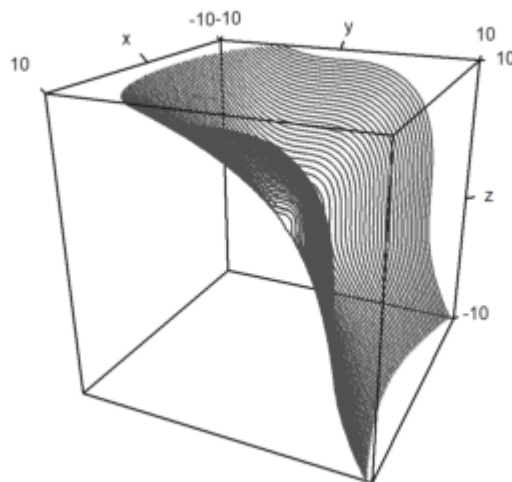
Contoh soal plot implisit

Gambarlah grafik fungsi berikut

$$f(x, y, z) = -8x^2 + 2y^3 + z^3$$

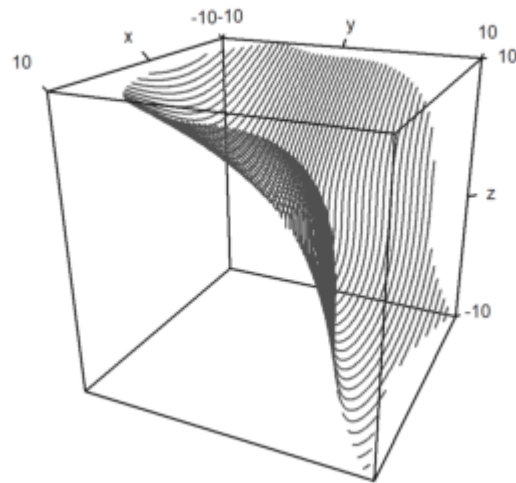
memotong sejajar bidang y-z

```
>plot3d("-8x^2+2y^3+z^3",r=10,implicit=1):
```



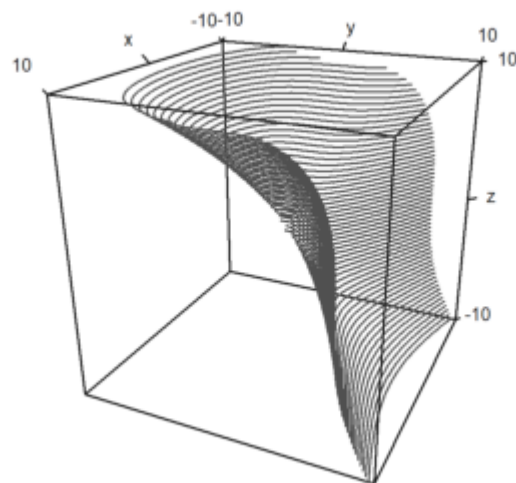
memotong sejajar bidang x-z

```
>plot3d("-8x^2+2y^3+z^3",r=10,implicit=2):
```



memotong sejajar bidang x-y

```
>plot3d("-8x^2+2y^3+z^3",r=10,implicit=4):
```



Merencanakan Data 3D

Sama seperti plot2d, plot3d menerima data. Untuk objek 3D, Anda perlu menyediakan matriks nilai x-, y- dan z, atau tiga fungsi atau ekspresi $f_x(x,y)$, $f_y(x,y)$, $f_z(x,y)$.

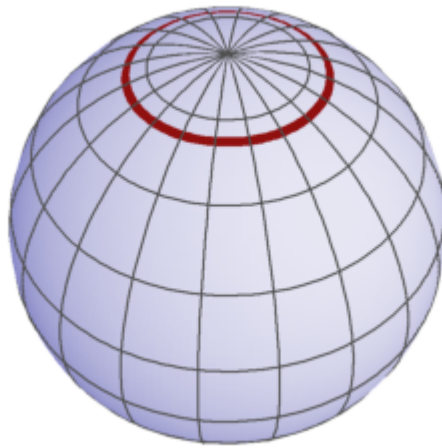
$$\gamma(t, s) = (x(t, s), y(t, s), z(t, s))$$

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

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

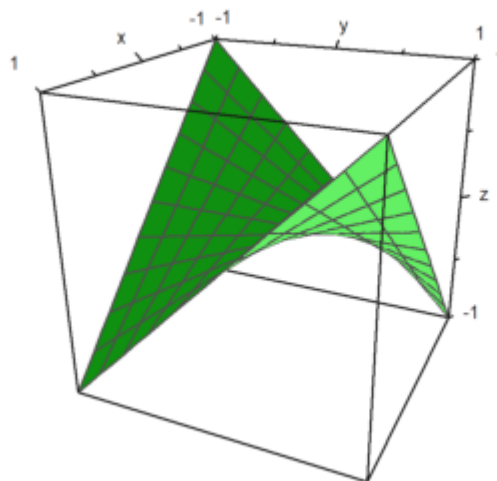
Dalam contoh berikut, kami menggunakan vektor nilai t dan vektor kolom nilai s untuk membuat parameter permukaan bola. Dalam gambar kita dapat menandai daerah, dalam kasus kita daerah kutub.

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



Berikut adalah contoh, yang merupakan grafik fungsi.

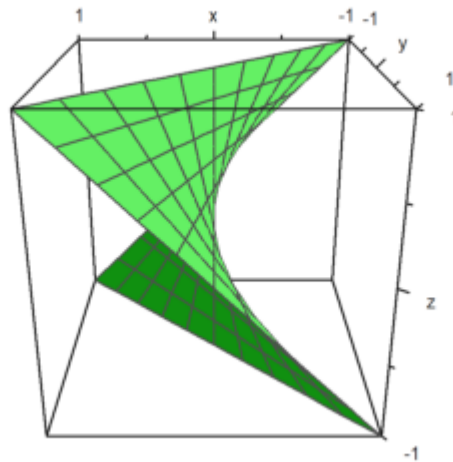
```
>t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,t*s,grid=10):
```



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

$$x = yz$$

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



Dengan lebih banyak usaha, kami dapat menghasilkan banyak permukaan.

Dalam contoh berikut, kita membuat tampilan bayangan dari bola yang terdistorsi. Koordinat biasa untuk bola adalah

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

dengan

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

Kami mendistorsi ini dengan sebuah faktor

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

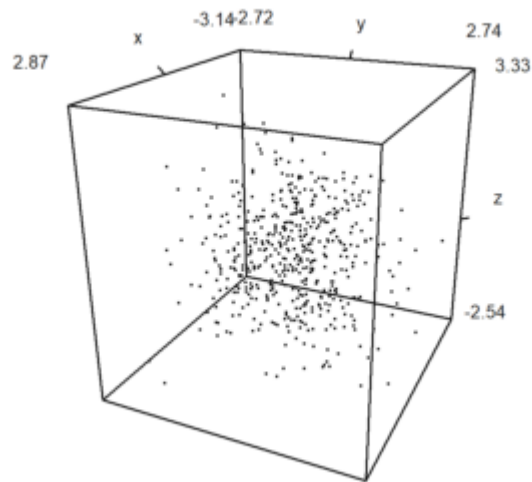
```
>t=linspace(0,2pi,320); s=linspace(-pi/2,pi/2,160)'; ...
d=1+0.2*(cos(4*t)+cos(8*s)); ...
plot3d(cos(t)*cos(s)*d,sin(t)*cos(s)*d,sin(s)*d,hue=1, ...
light=[1,0,1],frame=0,zoom=5):
```



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

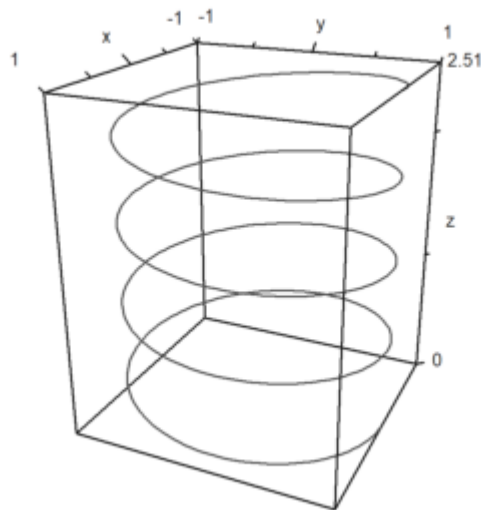
Gayanya sama seperti di plot2d dengan points=true;

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

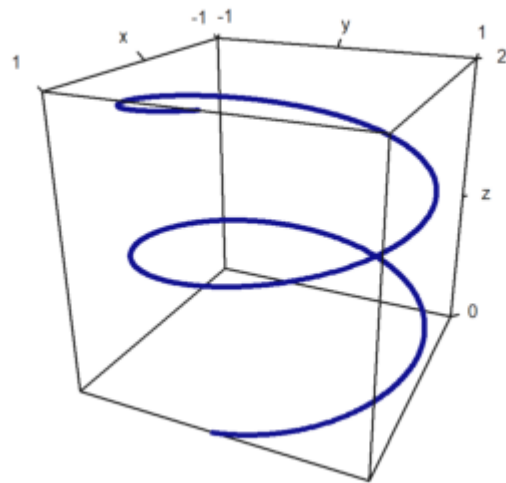


Dimungkinkan juga untuk memplot kurva dalam 3D. Dalam hal ini, lebih mudah untuk menghitung titik-titik kurva. Untuk kurva di pesawat kami menggunakan urutan koordinat dan parameter `wire=true`.

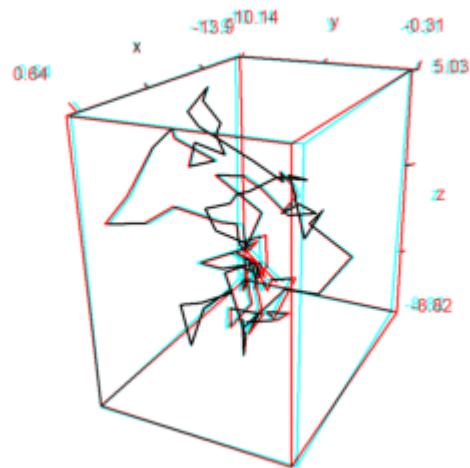
```
>t=linspace(0,8pi,500); ...  
plot3d(sin(t),cos(t),t/10,>wire, zoom=3):
```



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

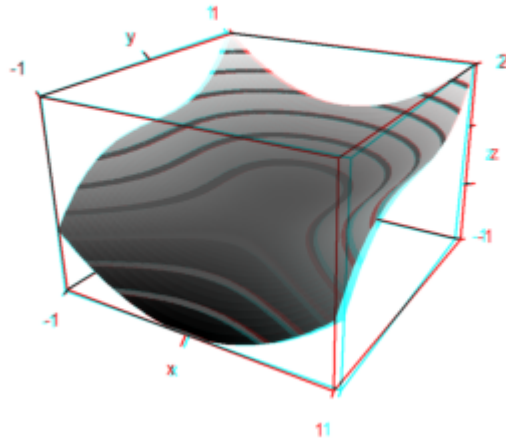


```
>X=cumsum(normal(3,100)); ...
  plot3d(X[1],X[2],X[3],>anaglyph,>wire):
```



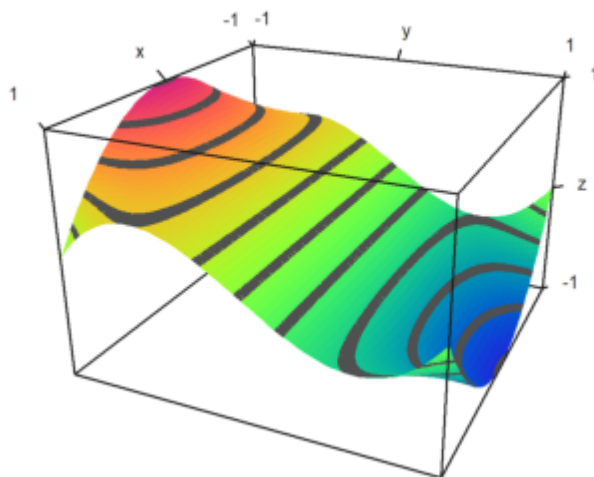
EMT juga dapat memplot dalam mode anaglyph. Untuk melihat plot seperti itu, Anda memerlukan kacamata merah/sian.

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



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

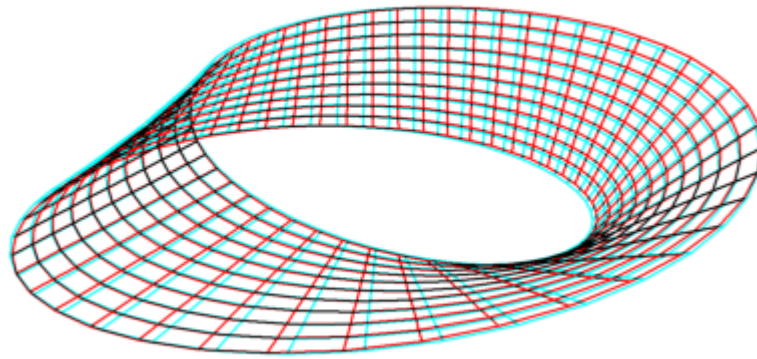
```
>plot3d("x^2*y^3-y",>spectral,>contour, zoom=3.2) :
```



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

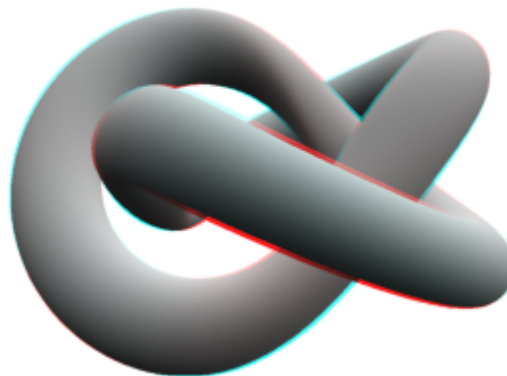
Untuk demo berikut, kami mengatur parameter u- dan v-, dan menghasilkan koordinat ruang dari ini.

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



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

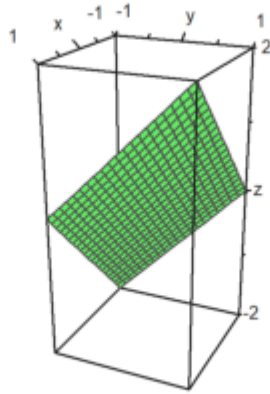
```
>u:=linspace(-pi,pi,160); v:=linspace(-pi,pi,400)'; ...
x:=(4*(1+.25*sin(3*v))+cos(u))*cos(2*v); ...
y:=(4*(1+.25*sin(3*v))+cos(u))*sin(2*v); ...
z=sin(u)+2*cos(3*v); ...
plot3d(x,y,z,frame=0,scale=1.5,hue=1,light=[1,0,-1],zoom=2.8,>anaglyph):
```



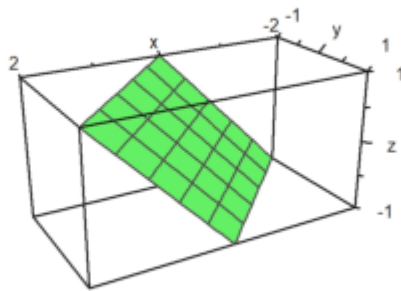
Contoh soal merencanakan data 3D

Grafik fungsi

```
>a=-1:0.1:1; b=(-1:0.1:1)'; plot3d(a,b,a+b,grid=20):
```

```
>plot3d(a+b,a,b,angle=150°,grid=5):
```



Plot Statistik

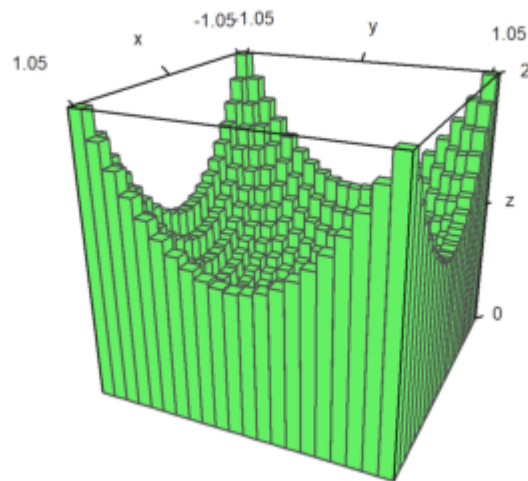
Plot bar juga dimungkinkan. Untuk ini, kita harus menyediakan

- x: vektor baris dengan $n+1$ elemen
- y: vektor kolom dengan $n+1$ elemen
- z: matriks nilai $n \times n$.

z bisa lebih besar, tetapi hanya nilai $n \times n$ yang akan digunakan.

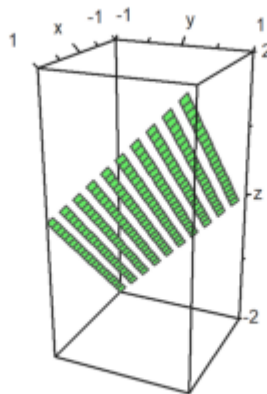
Dalam contoh, pertama-tama kita menghitung nilainya. Kemudian kita sesuaikan x dan y, sehingga vektor berpusat pada nilai yang digunakan.

```
>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):
```



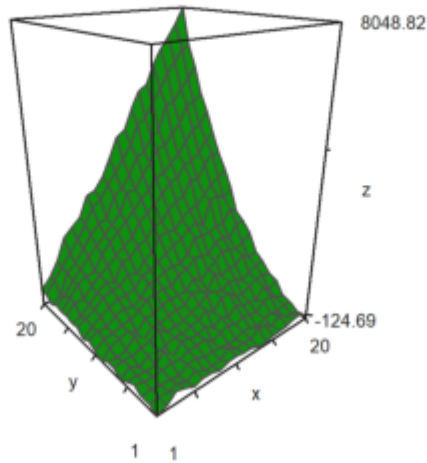
Dimungkinkan untuk membagi plot permukaan menjadi dua atau lebih bagian.

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

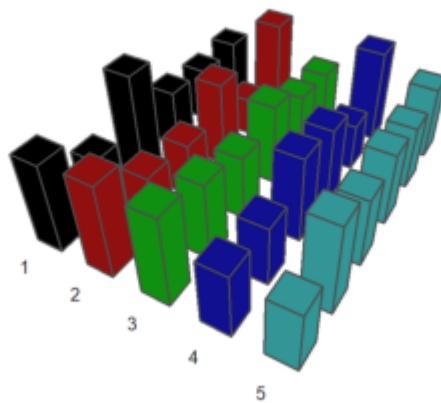


Jika memuat atau menghasilkan matriks data M dari file dan perlu memplotnya dalam 3D, Anda dapat menskalakan matriks ke $[-1,1]$ dengan `scale(M)`, atau menskalakan matriks dengan `>zscale`. Ini dapat dikombinasikan dengan faktor penskalaan individu yang diterapkan sebagai tambahan.

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



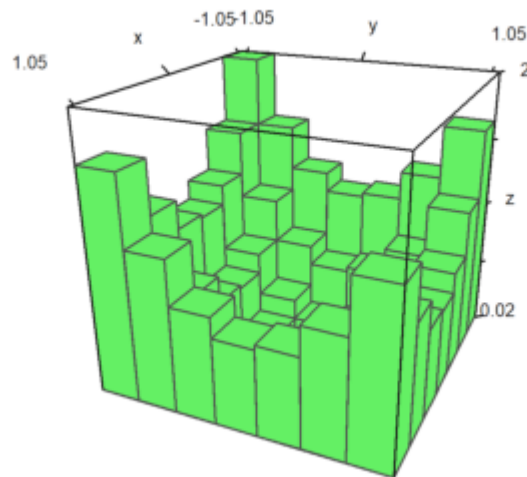
```
>Z=intrandom(5,100,6); v=zeros(5,6); ...
loop 1 to 5; v[#]=getmultiplicities(1:6,Z[#]); end; ...
columnplot3d(v',scols=1:5,ccols=[1:5]):
```



Contoh soal dan pembahasan plot statistik

Grafik

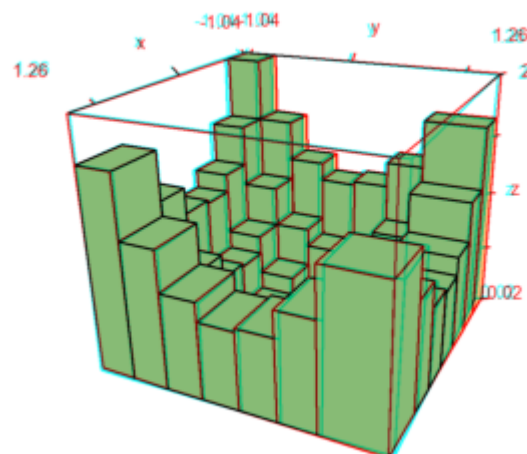
```
>x=-1:0.3:1; y=x'; z=x^2+y^2; ...
xa=(x|1.1)-0.05; ya=(y|1.1)-0.05; ...
plot3d(xa,ya,z,bar=true):
```



> x = -1:0.3:1; Kode ini membuat larik x yang berisi angka dari -1 hingga 1 dengan selang 0.3. Hasilnya adalah x akan berisi [-1, -0.7, -0.4, ..., 1].
 > y = x'; Ini menghasilkan larik y yang merupakan transposisi dari larik x, sehingga y akan memiliki nilai yang sama dengan x, yaitu [-1, -0.7, -0.4, ..., 1].
 > z = x^2 + y^2; Di sini, Anda menghitung larik z berdasarkan ekspresi matematika $x^2 + y^2$. Ini berarti setiap elemen z adalah hasil dari perhitungan yang melibatkan kuadrat elemen-elemen yang sesuai dari x dan y.
 > xa = (x|1.1) - 0.05; Ini mencoba mengubah larik x dengan menggantikan setiap elemen yang kurang dari 1.1 dengan hasil perhitungan $(x|1.1) - 0.05$. Operasi $(x|1.1)$ mungkin berarti pengambilan nilai maksimum antara setiap elemen x dan 1.1, dan kemudian dikurangi 0.05.
 > ya = (y|1.1) - 0.05; Ini mencoba mengubah larik y dengan mengurangi 0.05 dari setiap elemen yang ada pada larik y setelah dinaikkan pangkat sebanyak 1.1.
 > plot3d(xa, ya, z, bar=true); Ini adalah perintah untuk membuat plot 3D. Plot ini akan menggunakan data dari larik xa, ya, dan z sebagai data inputnya. Opsi bar=true mungkin digunakan untuk menghasilkan plot batang 3D.

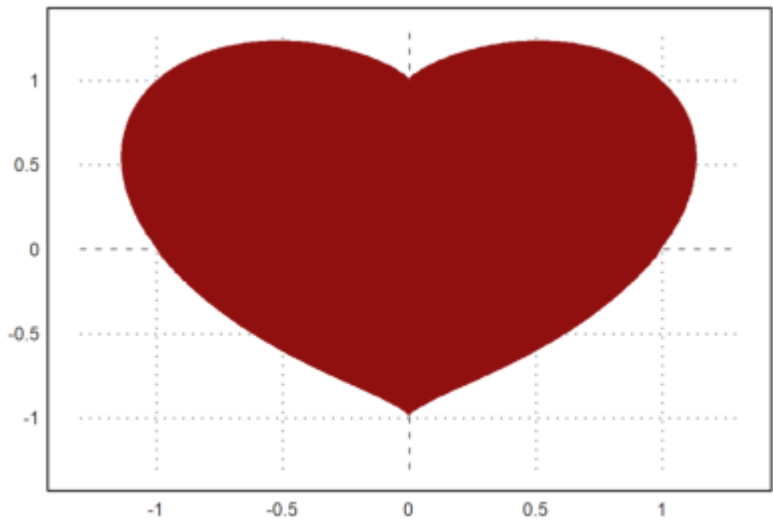
```

>x=-1:0.3:1; y=x'; z=x^2+y^2; ...
xa=(x|1.3)-0.04; ya=(y|1.3)-0.04; ...
plot3d(xa, ya, z, bar=true, >anaglyph) :
  
```



dengan menggunakan >anaglyph, kita dapat mengaktifkan efek anaglyph pada plot, yang biasanya digunakan dalam pemrosesan citra 3D untuk memberikan efek tiga dimensi saat dilihat dengan kacamata anaglyph.

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



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

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

Kami ingin memutar kurva jantung di sekitar sumbu y. Berikut adalah ungkapan, yang mendefinisikan hati:

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

Selanjutnya kita atur

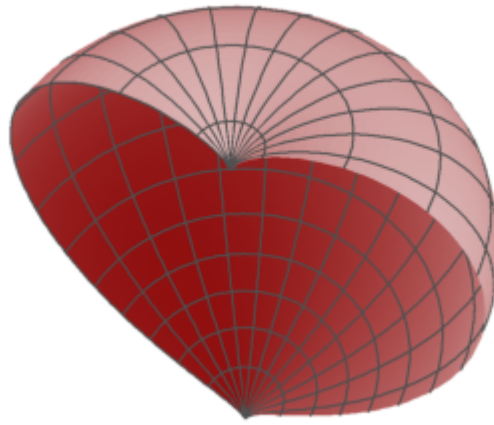
$$x = r \cdot \cos(a), \quad y = r \cdot \sin(a).$$

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

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

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

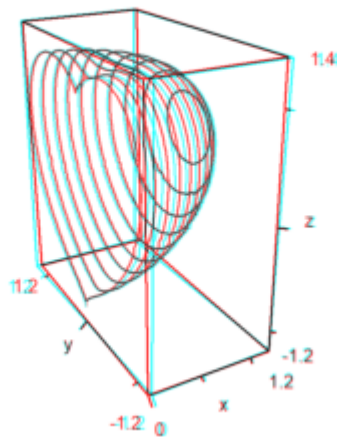
```
>function map f(a) := bisect("fr",0,2;a); ...
t=linspace(-pi/2,pi/2,100); r=f(t); ...
s=linspace(pi,2pi,100)'; ...
plot3d(r*cos(t)*sin(s),r*cos(t)*cos(s),r*sin(t), ...
>hue,<frame,color=red,zoom=4,amb=0,max=0.7,grid=12,height=50°):
```



Berikut ini adalah plot 3D dari gambar di atas yang diputar di sekitar sumbu z. Kami mendefinisikan fungsi, yang menggambarkan objek.

```
>function f(x,y,z) ...
  r=x^2+y^2;
  return (r+z^2-1)^3-r*z^3;
endfunction
```

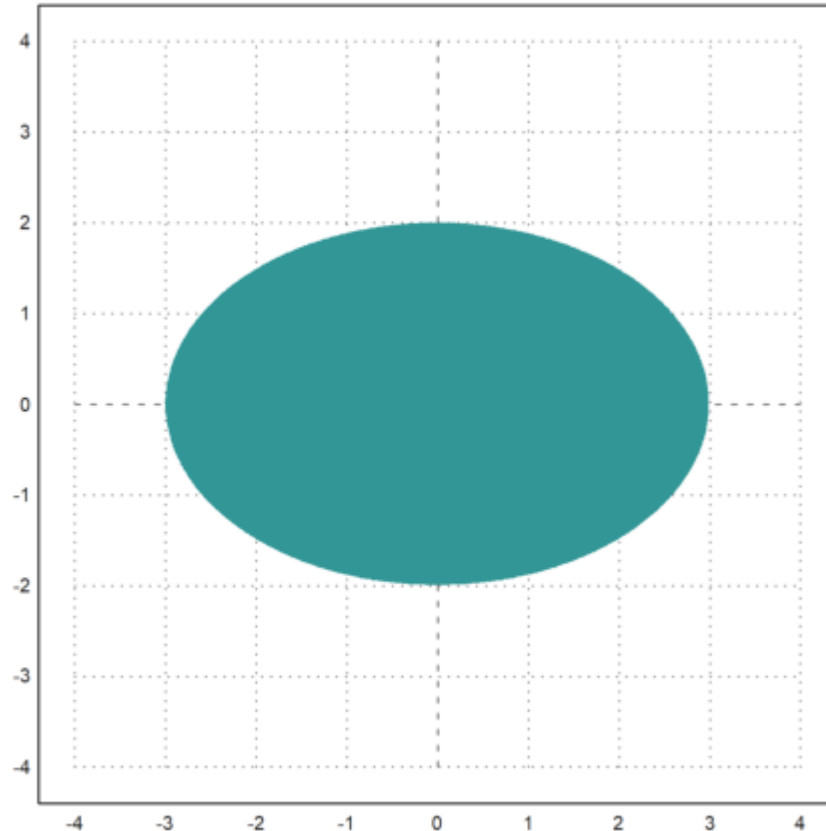
```
>plot3d("f(x,y,z)", ...
  xmin=0,xmax=1.2,ymin=-1.2,ymax=1.2,zmin=-1.2,zmax=1.4, ...
  implicit=1,angle=-30°,zoom=2.5,n=[10,60,60],>anaglyph):
```



Contoh soal dan pembahasan permukaan benda putar

Grafik

```
>plot2d("((x^2)/3^2)+((y^2)/2^2)-1",r=4, ...
  style="#",color=cyan,<outline, ...
  level=[-2;0],n=120):
```



> plot2d("((x^2)/3^2) + ((y^2)/2^2) - 1", r=4, ...: Ini adalah perintah untuk membuat plot 2D dari persamaan elips di mana radius maksimumnya adalah 4.

> style="": Dalam parameter style, Anda mungkin telah mengatur tampilan plot ke tampilan default (kosong) atau tanpa gaya tertentu.

> color=cyan: Ini mengatur warna plot, dalam hal ini, plot akan berwarna biru muda (cyan).

<outline: Ini mungkin digunakan untuk menyoroti atau menggambar garis tepi dalam plot.

> level=[-2;0]: Parameter ini menunjukkan level atau rentang nilai yang akan diplot dalam grafik. Di sini, nilai yang akan diplot berada dalam rentang -2 hingga 0.

> n=120: Ini mengatur jumlah titik data yang akan digunakan untuk membuat plot, dalam hal ini, 120 titik data akan digunakan.

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

$$\frac{y^2}{4} + \frac{x^2}{9} - 1$$

Kami ingin memutar kurva jantung di sekitar sumbu y. Berikut adalah ungkapan, yang mendefinisikan hati:

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

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)

$$\frac{(\cos(2a) + 1) r^2}{18} + \frac{(1 - \cos(2a)) r^2}{8} - 1$$

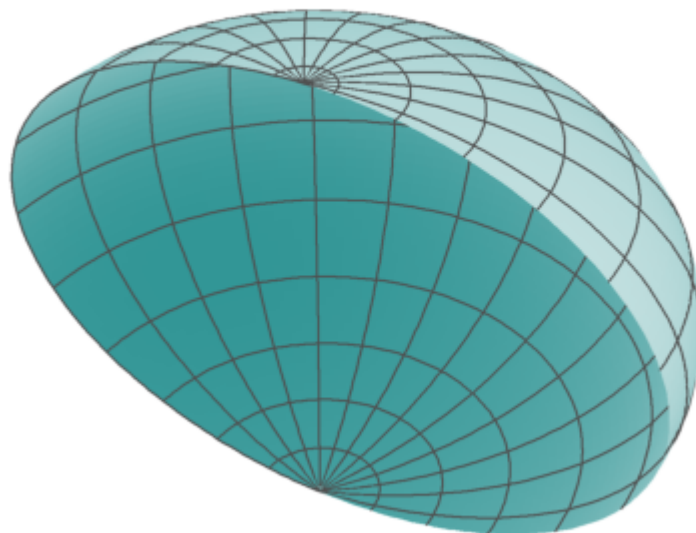
> function fr(r, a): Ini adalah deklarasi fungsi yang dinamakan fr dengan dua parameter, yaitu r dan a.

> ekspresi with $[x = r * \cos(a), y = r * \sin(a)]$: Ini adalah bagian dari perintah yang mendefinisikan ekspresi yang akan dihitung dalam fungsi fr. Dalam ekspresi ini, x dan y digantikan oleh nilai-nilai yang sesuai, yaitu $x = r * \cos(a)$ dan $y = r * \sin(a)$.

> trigreduce: Ini adalah perintah yang mungkin digunakan untuk menggantikan ekspresi trigonometri dengan bentuk yang lebih sederhana atau tereduksi.

> \$fr(r, a): Ini adalah hasil dari ekspresi yang menghitung fungsi fr dengan parameter r dan a. Hasil ini akan menggantikan nilai ekspresi yang telah didefinisikan dalam fungsi fr.

```
>function map f(a) := bisect("fr",0,4;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=cyan,zoom=4,amb=0,max=0.7,grid=12,angle=120°,height=35°):
```



> function map f(a) := bisect("fr", 0, 4; a):: Ini adalah definisi fungsi map yang disebut f. Fungsi ini menerima parameter a dan memanggil fungsi fr dengan nilai a sebagai argumen. Fungsi bisect digunakan untuk mencari akar atau solusi dari fungsi fr di dalam rentang $[0, 4]$ dengan menggunakan nilai a.

> t = linspace(-pi/2, pi/2, 100):: Ini menghasilkan larik t yang berisi 100 titik dalam rentang dari -p/2 hingga p/2.

> r = f(t):: Ini menghitung nilai r dengan memanggil fungsi f (yang dalam hal ini mencari akar fungsi fr) dengan menggunakan nilai-nilai dalam larik t.

> s = linspace(pi, 2 * pi, 100):: Ini menghasilkan larik s yang berisi 100 titik dalam rentang dari p hingga 2p.

> plot3d(r * cos(t) * sin(s), r * cos(t) * cos(s), r * sin(t), ...: Ini adalah perintah untuk membuat plot 3D dengan koordinat yang dihitung dari nilai-nilai r, t, dan s. Plot ini menggunakan warna yang berubah-ubah (hue) dan memiliki beberapa parameter lainnya seperti frame, zoom, angle, height, dan lainnya.

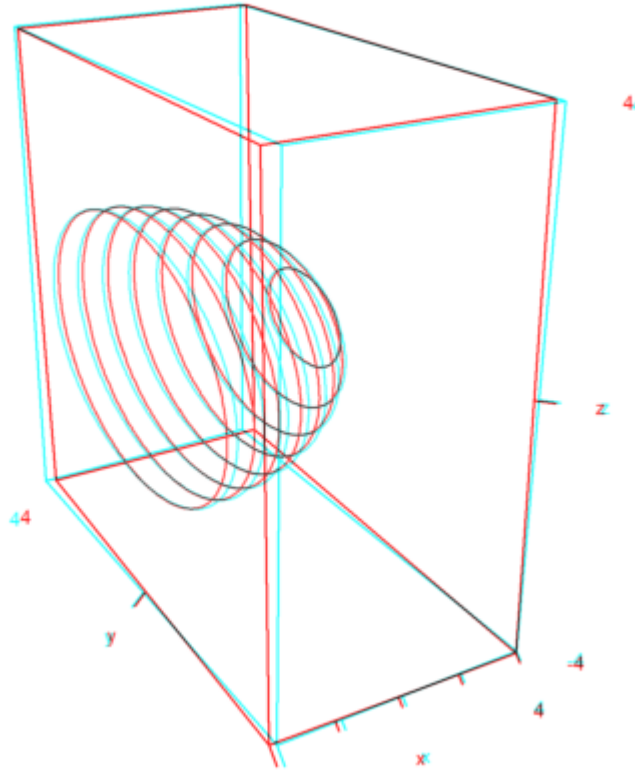
Berikut ini adalah plot 3D dari gambar di atas yang diputar di sekitar sumbu z. Kami mendefinisikan fungsi, yang menggambarkan objek.

```
>function f(x,y,z) ...
r=x^2+y^2;
```



```
return (r/3^2)+(z^2/2^2)-1;
endfunction
```

```
>plot3d("f(x,y,z)", ...
xmin=0,xmax=4,ymin=-4,ymax=4,zmin=-4,zmax=4, ...
implicit=1,angle=-30°,zoom=3,n=[10,60,60],frame=1,>anaglyph):
```



- implicit=1: potong sejajar dengan bidang y-z
- implicit=2: memotong sejajar dengan bidang x-z
- implicit=4: memotong sejajar dengan bidang x-y

> n=[10, 60, 60]: Parameter ini mengatur jumlah titik data dalam masing-masing arah x, y, dan z dalam plot 3D. Dalam hal ini, ada 10 titik dalam arah x, 60 titik dalam arah y, dan 60 titik dalam arah z.

> frame=1: Ini mungkin adalah parameter yang mengatur tampilan frame atau frame plot.

Plot 3D Khusus

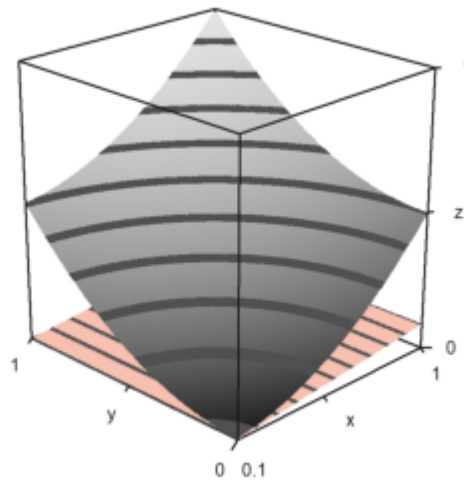
Fungsi plot3d bagus untuk dimiliki, tetapi tidak memenuhi semua kebutuhan. Selain rutinitas yang lebih mendasar, dimungkinkan untuk mendapatkan plot berbingkai dari objek apa pun yang Anda sukai.

Meskipun Euler bukan program 3D, ia dapat menggabungkan beberapa objek dasar. Kami mencoba memvisualisasikan paraboloid dan garis singgungnya.

```
>function myplot ...
y=0:0.01:1; x=(0.1:0.01:1)';
plot3d(x,y,0.2*(x-0.1)/2,<scale,<frame,>hue, ..
hues=0.5,>contour,color=orange);
h=holding(1);
plot3d(x,y,(x^2+y^2)/2,<scale,<frame,>contour,>hue);
holding(h);
endfunction
```

Sekarang framedplot() menyediakan frame, dan mengatur tampilan.

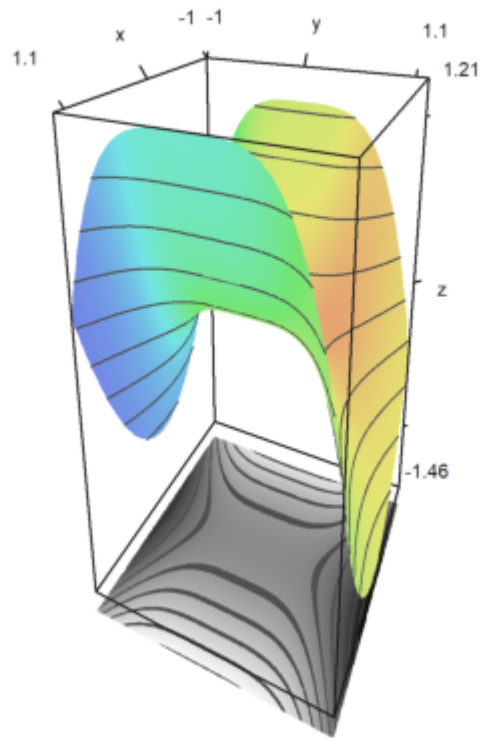
```
>framedplot("myplot",[0.1,1,0,1,0,1],angle=-45°, ...  
  center=[0,0,-0.7],zoom=6):
```



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

```
>x=-1:0.02:1.1; y=x'; z=x^2-y^4;  
>function myplot (x,y,z) ...  
  
  zoom(2);  
  wi=fullwindow();  
  plotcontourplane(x,y,z,level="auto",<scale);  
  plot3d(x,y,z,>hue,<scale,>add,color=white,level="thin");  
  window(wi);  
  reset();  
endfunction
```

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



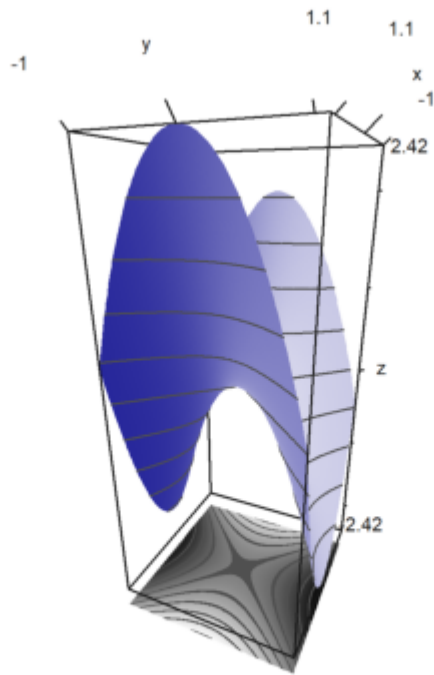
Contoh soal plot 3D khusus

Grafik

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

```
zoom(1);  
wi=fullwindow();  
plotcontourplane(x,y,z,level="auto",<scale);  
plot3d(x,y,z,>hue,<scale,>add,color=blue,level="thin");  
window(wi);  
reset();  
endfunction
```

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



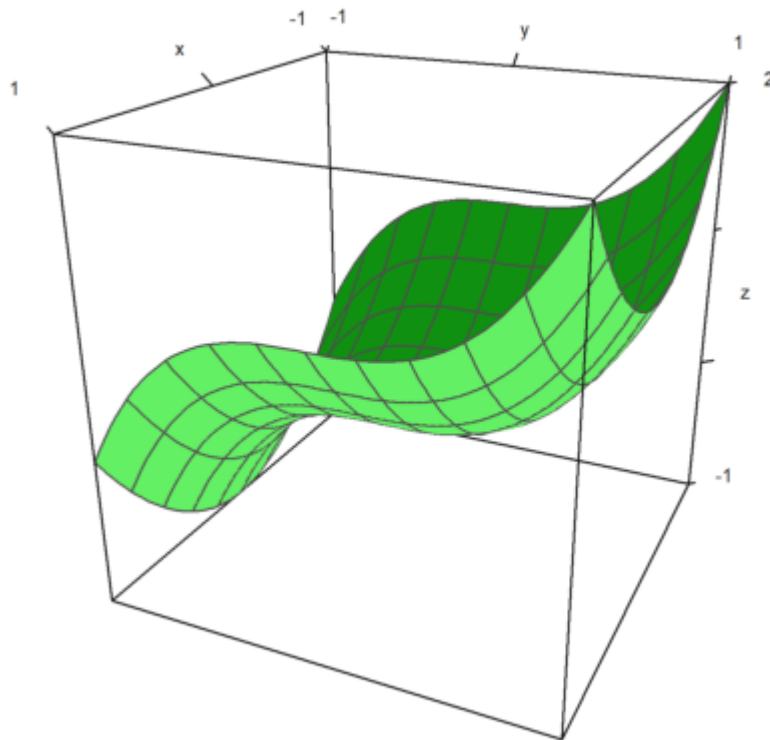
Animasi

Euler dapat menggunakan frame untuk menghitung animasi terlebih dahulu.

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

Silakan pelajari sumber rotasi untuk melihat lebih detail.

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

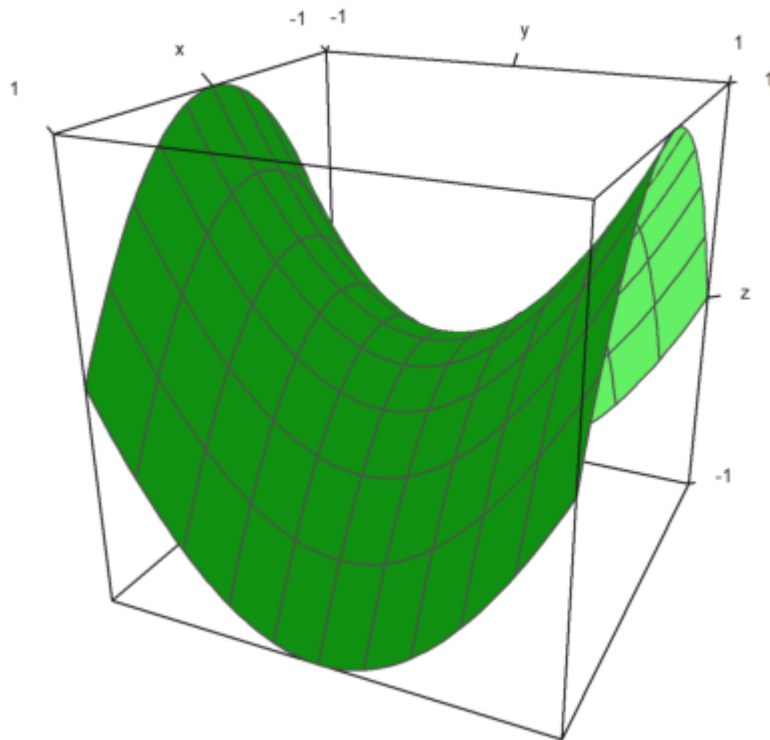


Contoh soal animasi

Gambarkan grafik animasi dari fungsi

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

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



Menggambar Povray

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

Anda perlu menginstal Povray (32bit atau 64bit) dari <http://www.povray.org/>, dan meletakkan sub-direktori "bin" dari

Povray ke jalur lingkungan, atau mengatur variabel "defaultpovray" dengan path lengkap yang menunjuk ke "pvengine.exe".

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

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

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

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

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

```
>load povray;
```

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

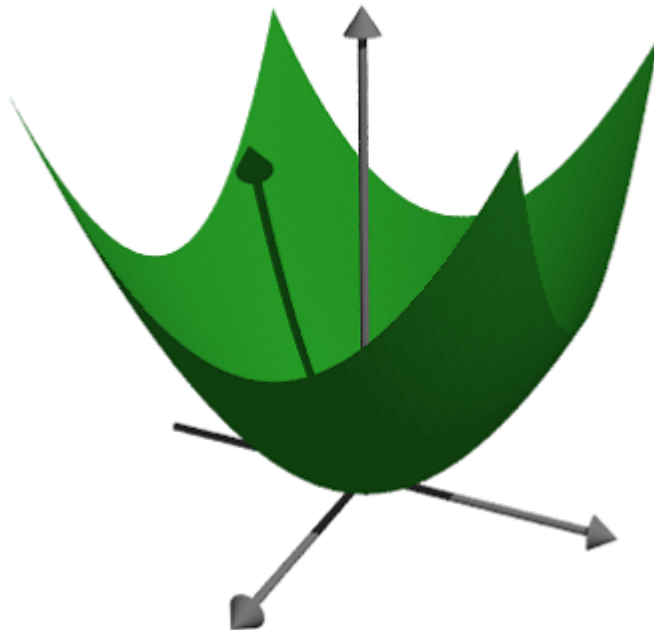
```
>defaultpovray="C:\Users\Microsoft User\AppData\Roaming\POV-Ray\v3.6\bin\pvengine.exe"
```

```
C:\Users\Microsoft User\AppData\Roaming\POV-Ray\v3.6\bin\pvengine.exe
```

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

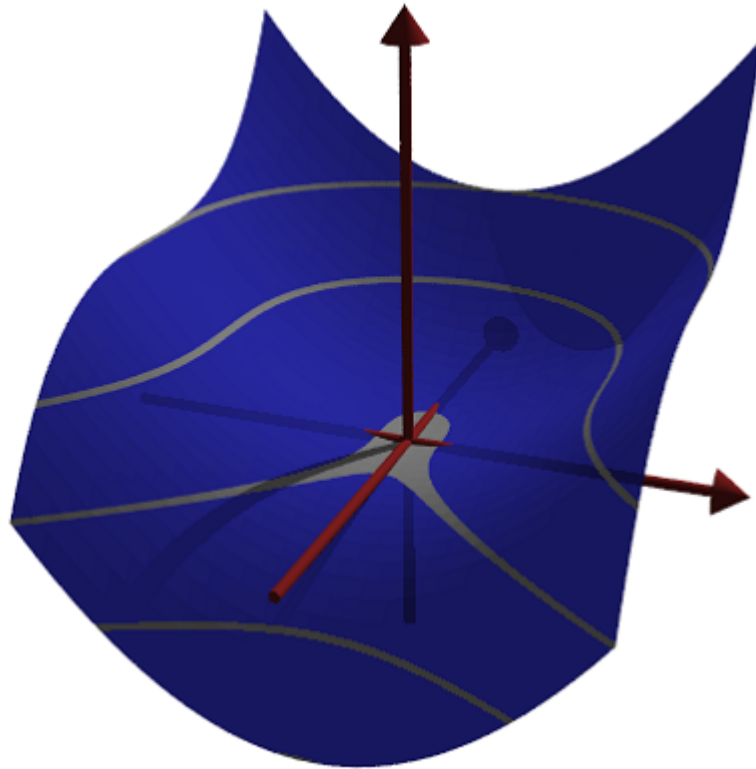
Jika Anda memulai perintah berikut, GUI Povray akan terbuka, menjalankan file, dan menutup secara otomatis. Karena alasan keamanan, Anda akan ditanya, apakah Anda ingin mengizinkan file exe untuk dijalankan. Anda dapat menekan batal untuk menghentikan pertanyaan lebih lanjut. Anda mungkin harus menekan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

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



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

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



Terkadang perlu untuk mencegah penskalaan fungsi, dan menskalakan fungsi dengan tangan.

Kami memplot himpunan titik di bidang kompleks, di mana produk dari jarak ke 1 dan -1 sama dengan 1.

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




Merencanakan dengan Koordinat

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

Dalam contoh kita memutar fungsi di sekitar sumbu z.

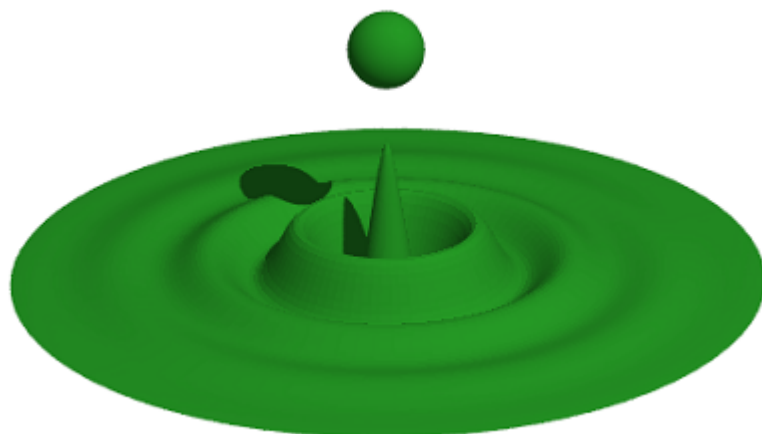
```
>function f(x) := x^3-x+1; ...  
x=-1:0.01:1; t=linspace(0,2pi,8)'; ...  
Z=x; X=cos(t)*f(x); Y=sin(t)*f(x); ...  
pov3d(X,Y,Z,angle=40°,height=20°,axis=0,zoom=4,light=[10,-5,5]);
```



Dalam contoh berikut, kami memplot gelombang teredam. Kami menghasilkan gelombang dengan bahasa matriks Euler.

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

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



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

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

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

$$\begin{matrix} 2 & 3 \\ x & y \end{matrix}$$

Persamaan permukaannya adalah $[x,y,Z]$. Kami menghitung dua turunan ke x dan y ini dan mengambil produk silang sebagai normal.

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

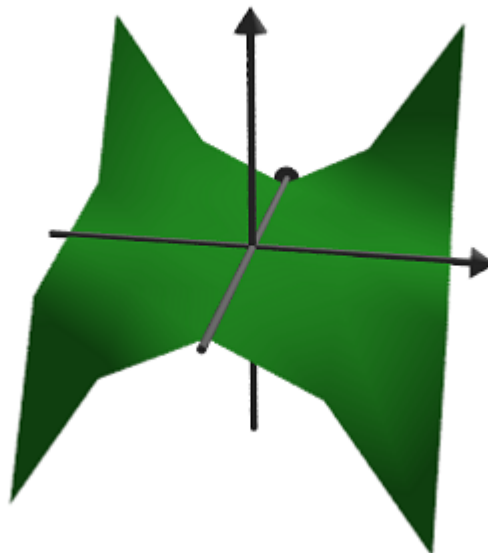
Kami mendefinisikan normal sebagai produk silang dari turunan ini, dan mendefinisikan fungsi koordinat.

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

$$[-2xy^3, -3x^2y^2, 1]$$

Kami hanya menggunakan 25 poin.

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



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

Simpul Trefoil

Untuk tampilan yang bagus dengan tidak terlalu banyak titik, kami menambahkan vektor normal di sini. Kami menggunakan Maxima untuk menghitung normal bagi kami. Pertama, ketiga fungsi koordinat

sebagai ekspresi simbolik.

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

Kemudian kedua vektor turunan ke x dan y.

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

Sekarang normal, yang merupakan produk silang dari dua turunan.

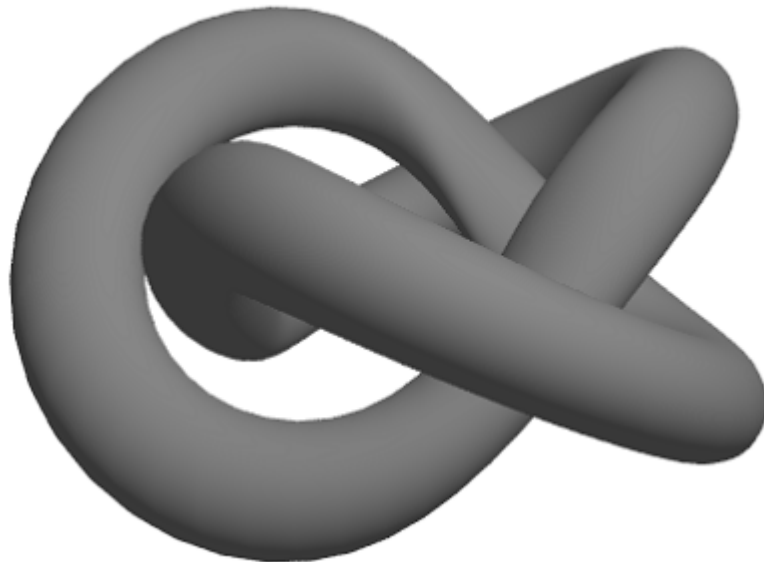
```
>dn &= crossproduct(dx,dy);
```

Kami sekarang mengevaluasi semua ini secara numerik.

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

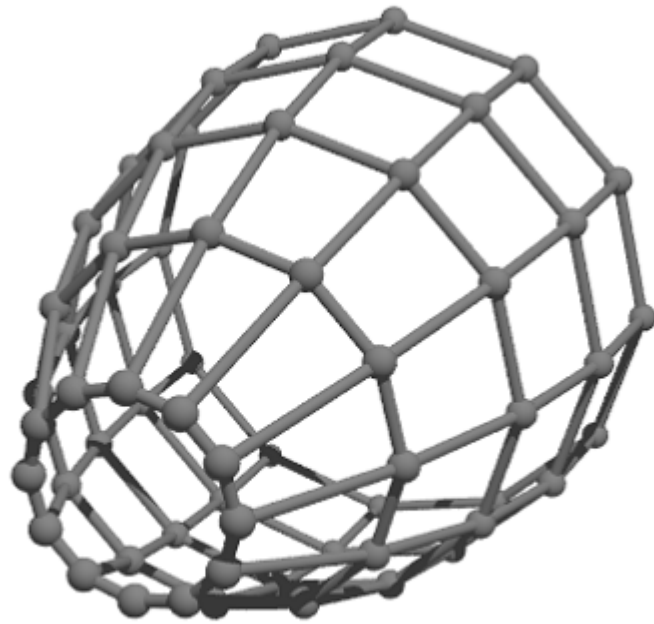
Vektor normal adalah evaluasi dari ekspresi simbolik dn[i] untuk i=1,2,3. Sintaks untuk ini adalah &"expression"(parameters). Ini adalah alternatif dari metode pada contoh sebelumnya, di mana kita mendefinisikan ekspresi simbolik NX, NY, NZ terlebih dahulu.

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



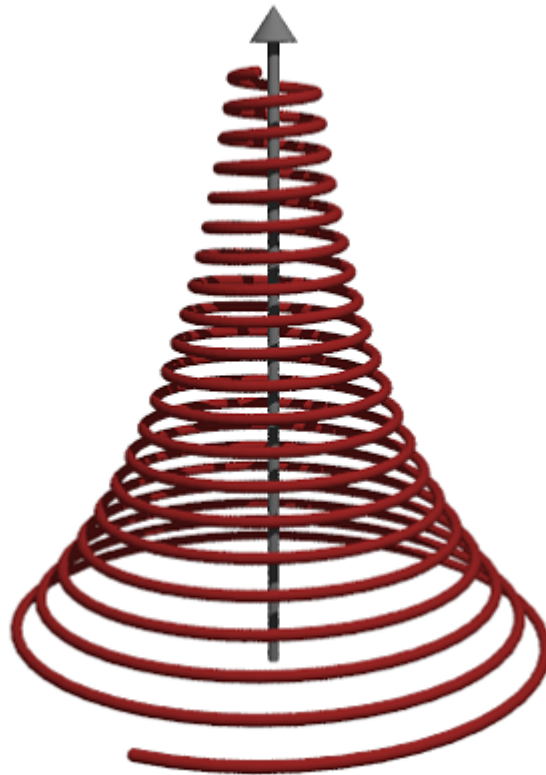
Kami juga dapat menghasilkan grid dalam 3D.

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



Dengan `povgrid()`, kurva dimungkinkan.

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



Objek Povray

Di atas, kami menggunakan pov3d untuk memplot permukaan. Antarmuka povray di Euler juga dapat menghasilkan objek Povray. Objek-objek ini disimpan sebagai string di Euler, dan perlu ditulis ke file Povray.

Kami memulai output dengan povstart().

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

Pertama kita mendefinisikan tiga silinder, dan menyimpannya dalam string di Euler.

Fungsi povx() dll. hanya mengembalikan vektor [1,0,0], yang dapat digunakan sebagai gantinya.

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

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, kami menambahkan tekstur ke objek dalam tiga warna berbeda.

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

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

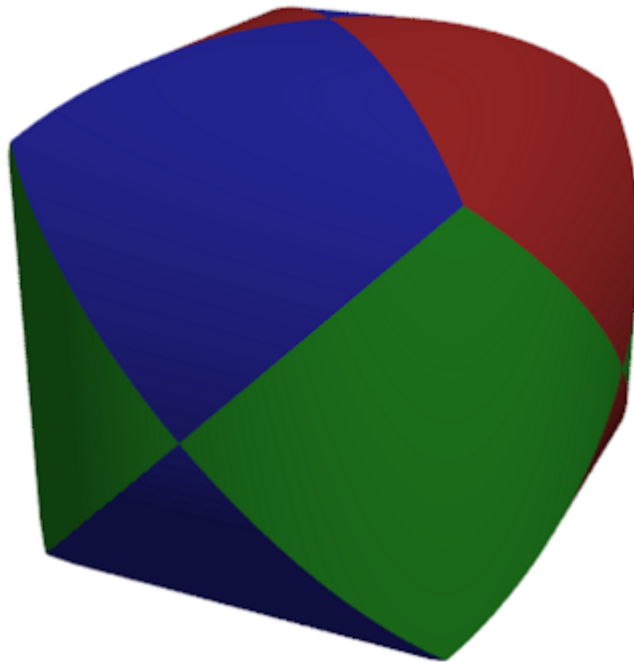
```
texture { pigment { color rgbf <0.101961,0.2,0.301961,0.1> } }  
finish { ambient 0.5 }
```

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

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

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

```
>povend;
```



Fungsi berikut menghasilkan fraktal secara rekursif.

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

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

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

```

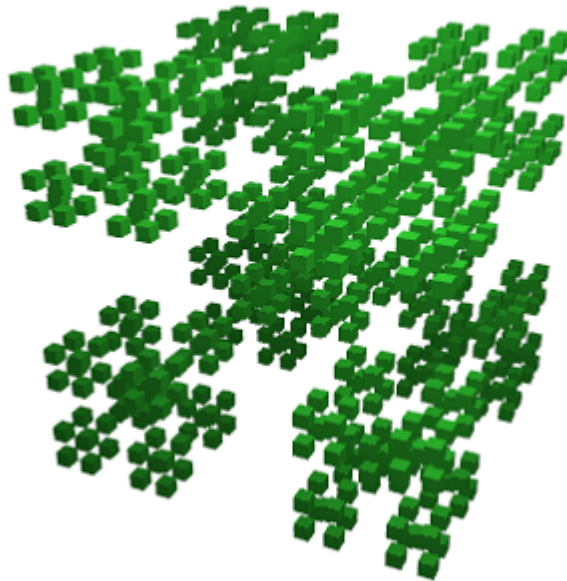
    fractal (x+2*h,y+2*h,z+2*h,h,n-1);
    fractal (x+h,y+h,z+h,h,n-1);
endif;
endfunction

```

```

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

```



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

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

Untuk demonstrasi ini, kami mendefinisikan objek di Povray, alih-alih menggunakan string di Euler. Definisi ditulis ke file segera.

Koordinat kotak -1 berarti [-1,-1,-1].

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

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

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

Kami menghasilkan kubus kedua, dan memutar dan menskalakannya sedikit.

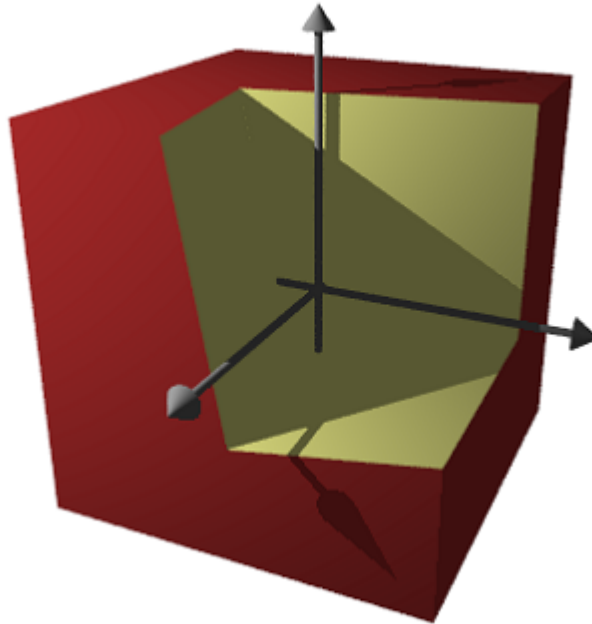
```
>c2=povobject ("mycube",povlook (yellow),translate=[1,1,1], ...
rotate=xrotate (10°)+yrotate (10°), scale=1.2);
```

Kemudian kita ambil selisih kedua benda tersebut.


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

Sekarang tambahkan tiga sumbu.

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



Contoh soal povray

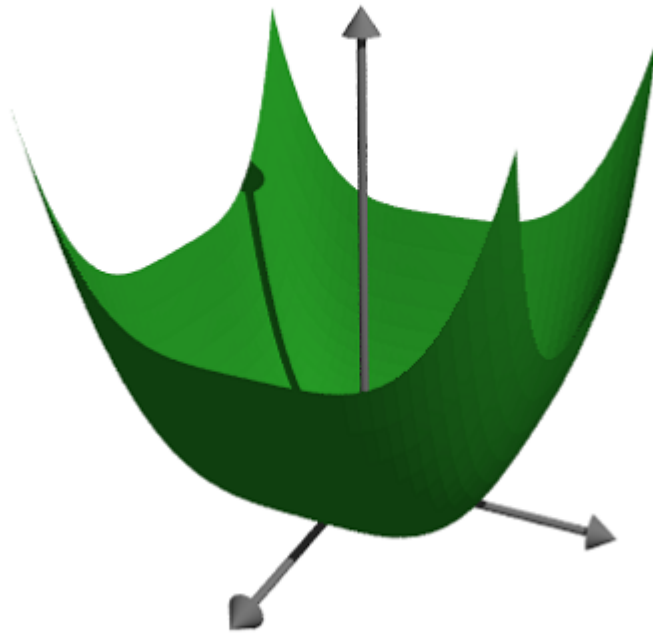
Gambarkan grafik fungsi berikut

$$f(x,y) = x^4 + y^4$$

```
>load povray;  
>defaultpovray="C:\Users\Microsoft User\AppData\Roaming\POV-Ray\v3.6\bin\pvengine.exe"
```

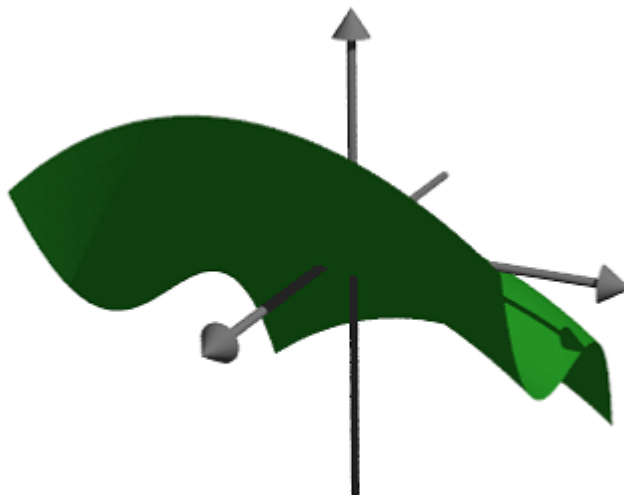
```
C:\Users\Microsoft User\AppData\Roaming\POV-Ray\v3.6\bin\pvengine.exe
```

```
>pov3d("x^4+y^4", zoom=3);
```



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

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



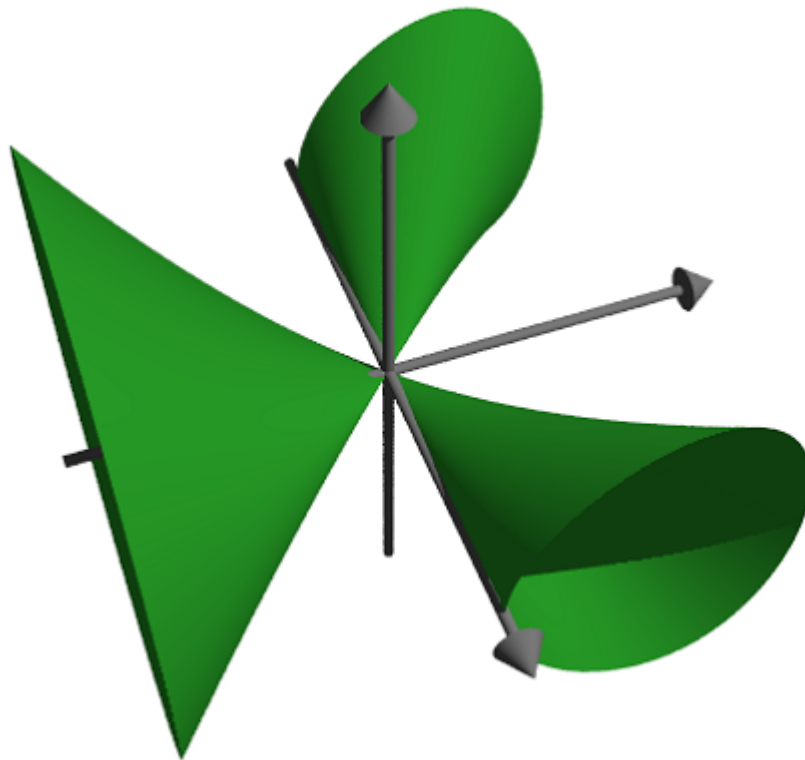
Povray dapat memplot himpunan di mana $f(x,y,z)=0$, seperti parameter implisit di plot3d. Namun, hasilnya terlihat jauh lebih baik.

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

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

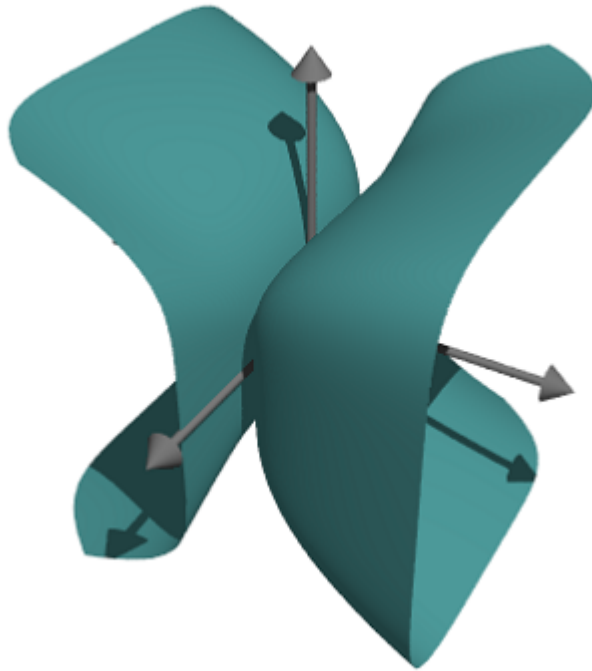
Buat permukaan implisit. Perhatikan sintaks yang berbeda dalam ekspresi.

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

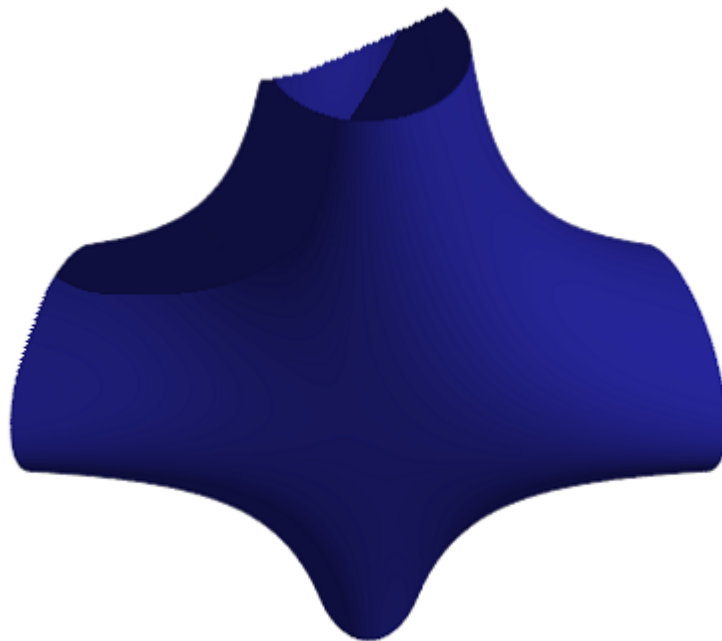


Contoh soal dan pembahasan fungsi implisit

```
>povstart (angle=120°, height=35°, zoom=3) ;  
>writeln (povsurface ("pow(x,5)*y-pow(y,2)+pow(z,6)", povlook(cyan))) ; ...  
writeAxes() ; ...  
povend() ;
```



```
>povstart (angle=90°, height=40°, zoom=2);  
>writeln (povsurface ("pow(x,2)+pow(y,2)*pow(z,2)-1", povlook(blue), povbox(-2,2,"")));  
>povend();
```



>load povray : memuat file povray

>defaultpovray : Perintah berikut ini menghasilkan file povray di direktori pengguna Anda, dan menjalankan

Povray untuk melacak sinar pada file ini.

>writeln(povsurface(...)) : perintah untuk menuliskan kode

povlook : mengatur tampilan warna

povbox(-2,2,""): Ini mendefinisikan sebuah kotak (box) dengan panjang sisi 4 (dari -2 hingga 2) dan tidak memiliki tekstur atau warna.

Objek Jala

Dalam contoh ini, kami menunjukkan cara membuat objek mesh, dan menggambarinya dengan informasi tambahan.

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

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

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

Berikut ini mendefinisikan objek mesh, dan langsung menuliskannya 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.

```
>c1=povdisc ([0.5,0.5,0], [1,1,0], 2); ...
  ll=povdisc ([0,0,1/4], [0,0,1], 2);
```

Tulis permukaan dikurangi dua cakram.

```
>writeln (povdifference (mesh, povunion ([c1, ll]), povlook (green)));
```

Tulis dua persimpangan.

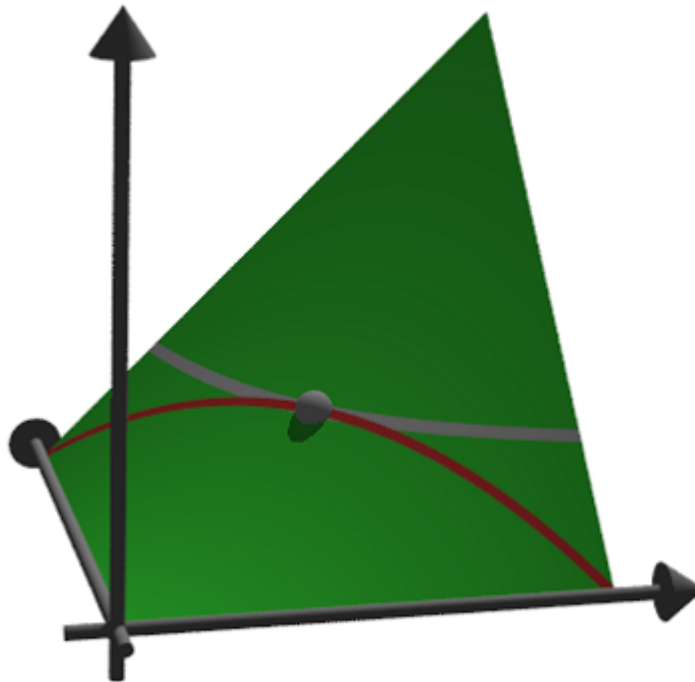
```
>writeln (povintersection ([mesh, c1], povlook (red))); ...
  writeln (povintersection ([mesh, ll], povlook (gray)));
```

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();
```



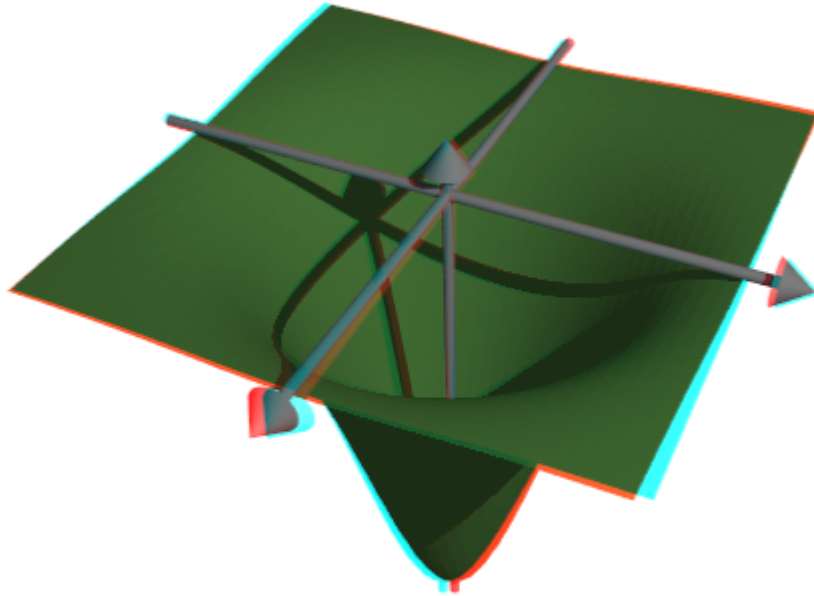
Anaglyph di Povray

Untuk menghasilkan anaglyph untuk kacamata merah/sian, Povray harus berjalan 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/sian untuk melihat contoh berikut dengan benar.

Fungsi `pov3d()` memiliki sakelar sederhana untuk menghasilkan anaglyphs.

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

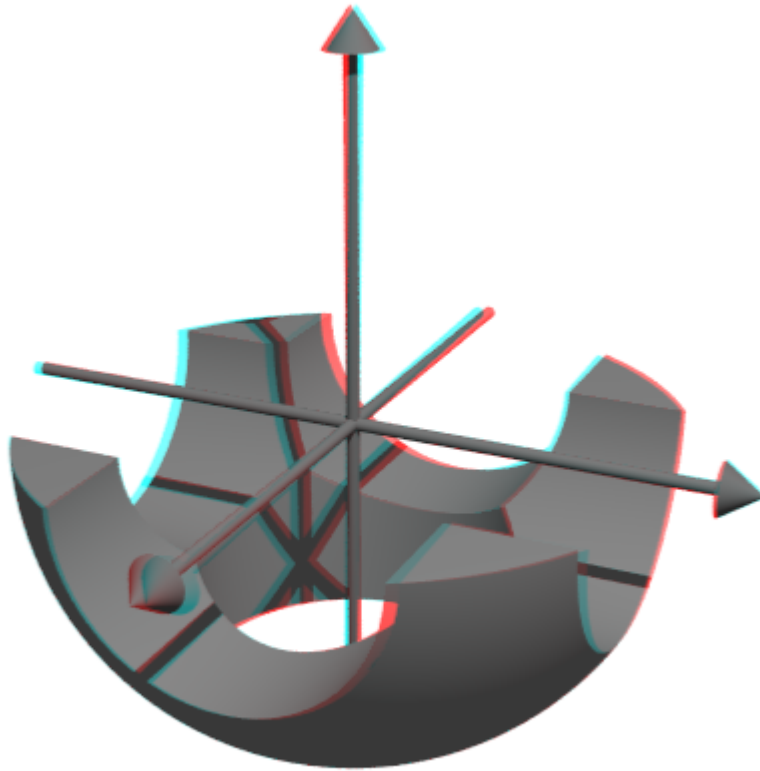


Jika Anda membuat adegan dengan objek, Anda perlu menempatkan generasi adegan ke dalam fungsi, dan menjalankannya dua kali dengan nilai yang berbeda untuk parameter anaglyph.

```
>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);
```

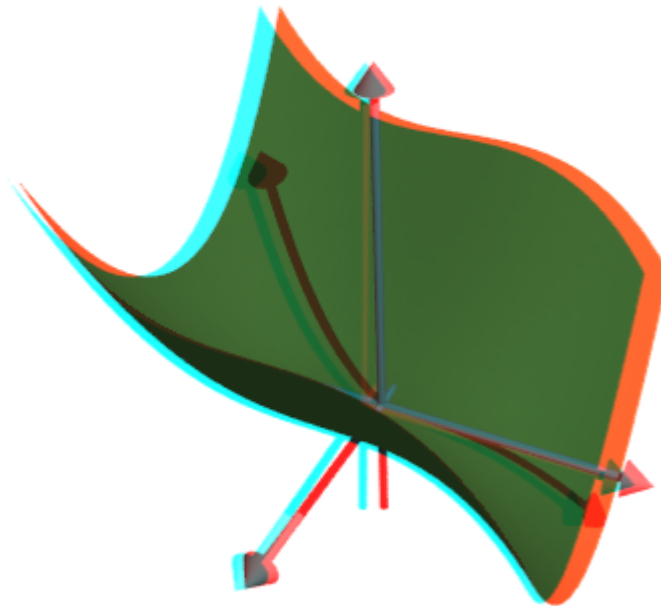


Contoh soal anaglyph di povray

Gambarkan grafik fungsi berikut

$$f(x, y) = 7x^2 - y^3$$

```
>pov3d("7x^2-y^3",r=4,height=35°,>anaglyph, ...  
center=[1,0,1],zoom=3);
```

Mendefinisikan Objek sendiri

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

Kami mendemonstrasikan sebuah torus. Perintah Povray untuk ini adalah "torus". Jadi kami mengembalikan 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 kami.

```
>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, kami menggunakan Phong Shading.

```
>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. Kami memecahkan

$$Ax \leq b, \quad x \geq 0, \quad c \cdot x \rightarrow \max$$

dan menunjukkan titik 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 mendefinisikan dua objek. Yang pertama adalah pesawat

$$a \cdot x \leq b$$

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

Kemudian kita mendefinisikan persimpangan dari semua setengah ruang dan sebuah kubus.

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

Kita sekarang dapat menggambar bentuknya.

```
>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.

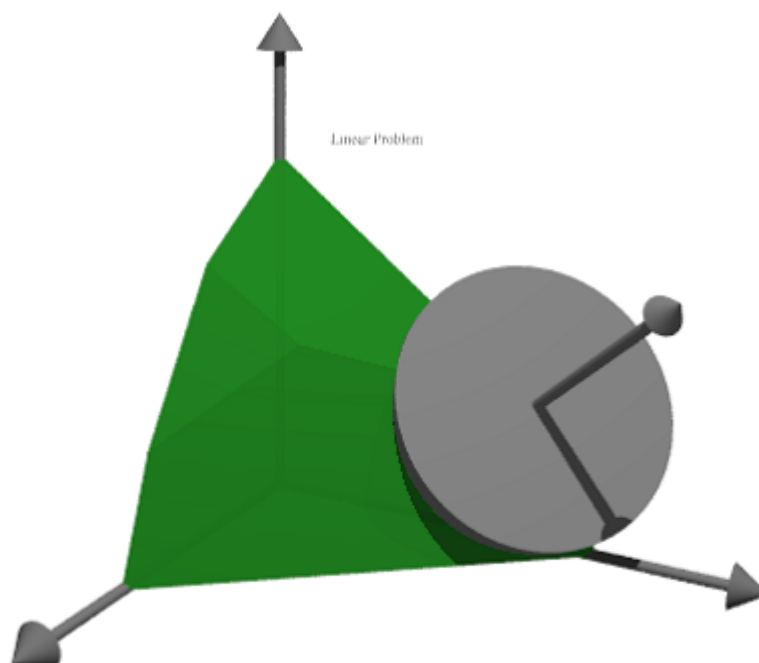
```
>writeln(povintersection([povsphere(x,0.5),povplane(c,c.x')], ...
  povlook(red,0.9)));
```

Dan kesalahan ke arah yang optimal.

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

Kami menambahkan teks ke layar. Teks hanyalah objek 3D. Kita perlu menempatkan dan memutarinya menurut pandangan kita.

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



Contoh lain mendefinisikan objek sendiri

```
>function povdonat (r1,r2,look="") ...
  return "torus {"+r1+"",""+r2+look+"}";
endfunction
```

```
>t1=povdonat(0.5,0.10)
```

```
torus {0.5,0.1}
```

```
>t2=povobject(t1,rotate=xrotate(120°),translate=[0.4,0,0])
```

```
object { torus {0.5,0.1}
  rotate 120 *x
  translate <0.4,0,0>
}
```

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

```
>povstart (center=[0.2,0,0],angle=90°,zoom=3,aspect=1.5); ...
writeln(povobject(t1,povlook(cyan,phong=1))); ...
writeln(povobject(t2,povlook(cyan,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=330,w=560);
```



Lebih Banyak Contoh

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

- [Examples/Dandelin Spheres](#)
- [Examples/Donat Math](#)
- [Examples/Trefoil Knot](#)
- [Examples/Optimization by Affine Scaling](#)

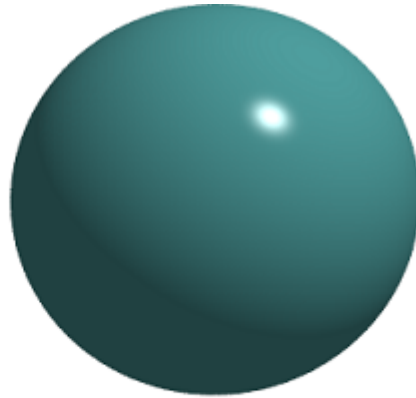
Contoh-contoh lain

```
>load povray;  
>defaultpovray="C:\Users\Microsoft User\AppData\Roaming\POV-Ray\v3.6\bin\pvengine.exe"
```

```
C:\Users\Microsoft User\AppData\Roaming\POV-Ray\v3.6\bin\pvengine.exe
```

Menggambar bola

```
>function povbola (r1,r2) ...  
    return "sphere {"+r1+", "+r2+"}";  
endfunction  
  
>t1=povbola(0.4,0.8); ...  
povstart (center=[0.4,0,0], angle=90°, zoom=3, aspect=1.5); ...  
writeln (povobject (t1, povlook (cyan, phong=1))); ...  
povend (h=350, w=550);
```



Menggambar suatu benda

```
>function povbenda () ...  
    return "cylinder {<1,0,1>, <0,2,0>, 2 open}";  
endfunction  
  
>t2=povbenda (); ...  
povstart (center=[0,2,0], angle=90°, zoom=1, aspect=1.5); ...  
writeln (povobject (t2, povlook (red, phong=2))); ...  
povend (h=300, w=500);
```

