# Menggambar Plot 3D dengan EMT

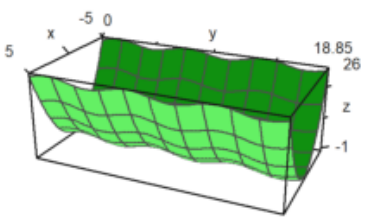
Ini adalah pengenalan plot 3D di Euler. Kita memerlukan plot 3D untuk memvisualisasikan fungsi dua variabel.

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

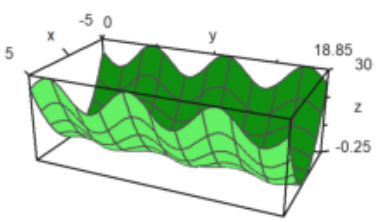
Euler dapat memetakan:
- permukaan dengan bayangan dan garis level atau rentang level,
- awan titik-titik,
-kurva parametrik,
- permukaan implisit.

Plot 3D dari sebuah fungsi menggunakan plot3d. Cara termudah adalah memplot ekspresi dalam x dan y. Parameter r mengatur rentang plot di sekitar (0,0).
Translated with DeepL.com (free version)

```
>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.

## Fungsi Dua Variabel

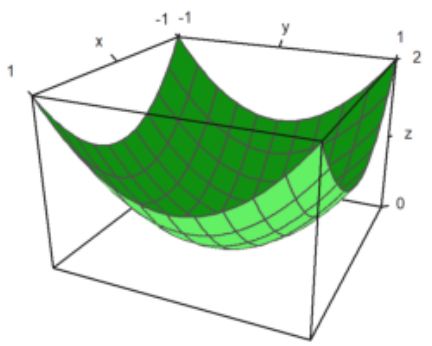
Untuk grafik suatu fungsi, gunakan:
- Ekspresi sederhana dalam x dan y,
- Nama fungsi dari dua variabel
- Atau matriks data

Standarnya adalah kisi-kisi kawat yang terisi dengan warna yang berbeda di kedua siis. Perhatikan bahwa jumlah default interval grid adalah 10, namun plot menggunakan jumlah default 40x40 persegi panjang untuk membangun permukaan. Hal ini dapat diubah.

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

Disini kita menggunakan default n=40 dan grid=10.

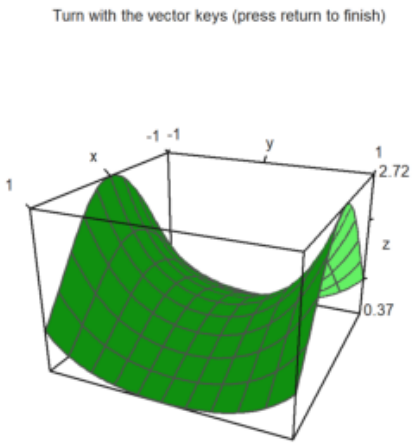
```
>plot3d("x^2+y^2"):
```




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

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

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

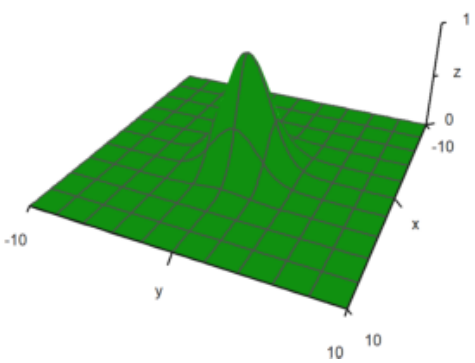



Rentang plot untuk fungsi dapat ditentukan dengan
- a, b: rentang x
- c, d: rentang y
- r: bujur sangkar simetris di sekitar (0,0).
- n: jumlah subinterval untuk plot.

Terdapat beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.

fscale: skala untuk nilai fungsi (standarnya adalah <fscale).
scale: angka atau vektor 1x2 untuk menskalakan 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,>user):
```




Tampilan dapat diubah dengan berbagai cara.
- distance: jarak pandang ke plot
- zoom: nilai zoom
- angle: sudut ke sumbu y negatif dalam radian
- height: ketinggian tampilan dalam radian

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

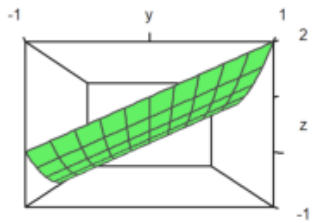
```
>view
```

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

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

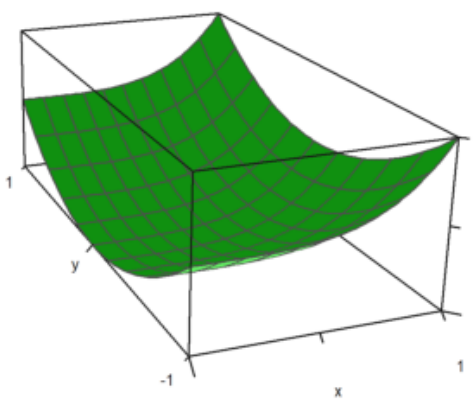
Dalam contoh berikut, angle=10 dan height=0 terlihat dari sumbu y negatif. Label sumbu untuk y disembunyikan dalam kasus ini.

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



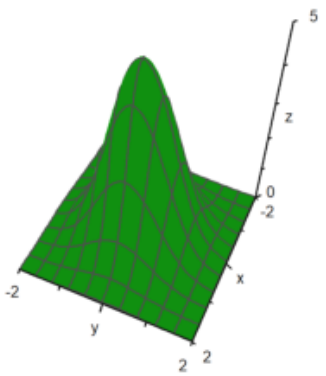

Plot selalu terlihat ke bagian tengah kubus plot. Kalian dapat memindahkan bagian tengah dengan parameter center.

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



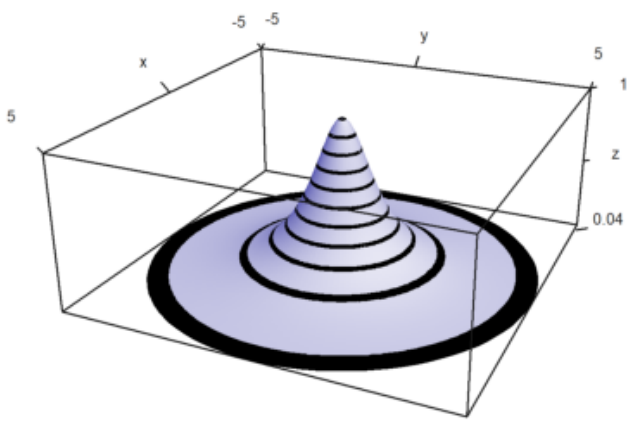
Plot sudah diatur secara default agar sesuai dengan kubus satuan untuk dilihat. Jadi kalian tidak perlu menggunah distance atau zoomnya untuk menyesuaikan dengan ukuran plot. Namun, Labels merujuk pada ukuran sebenarnya.
Jika kalian mematikan perintah ini dengan scale=false, kalian perlu meninjaklanjutinya, supaya plot yang kalian buat tetap sesuai dengan plotting window, dengan mengubah jarak pandang (viewing distance) atau zoom, dan memindahkan pusatnya.

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

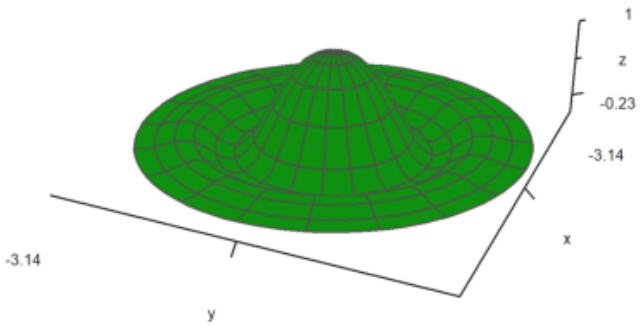



Plot polar juga tersedia. Parameter polar=true digunakan untuk menggambar plot polar. Fungsinya harus masih dalam fungsi x dan y. Parameter "fscale" mengatur fungsi dengan skalanya sendiri. Jika tidak, fungsi tersebut akan diskalakan supaya sesuai dengan kubus/cube.

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

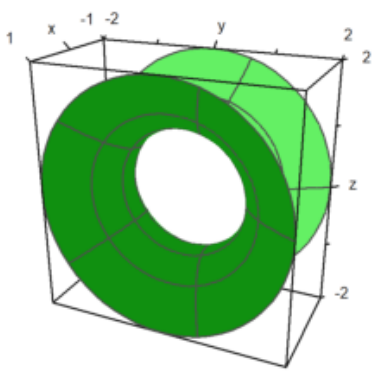


```
>function f(r) := exp(-r/2)*cos(r); ...
>plot3d("f(x^2+y^2)",>polar,scale=[1,1,0.4],r=pi,frame=3,zoom=4):
```

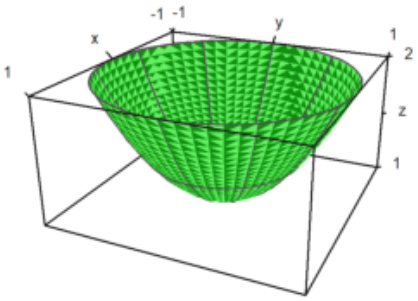



Parameter "rotate" digunakan untuk merotasikan fungsi di x di sekitar x-axis.
- rotate=1: menggunakan x-axis
- rotate=2: menggunakan z-axis

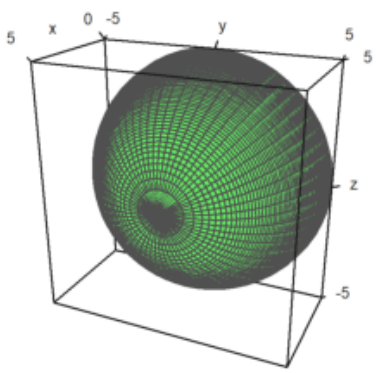
```
>plot3d("x^2+1",a=-1,b=1,rotate=true,grid=5):
```



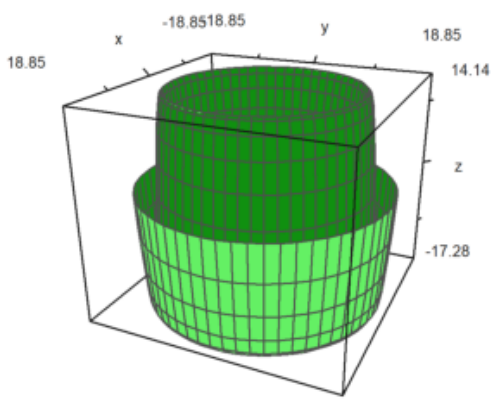

```
>plot3d("x^2+1",a=-1,b=1,rotate=2,grid=5):
```




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

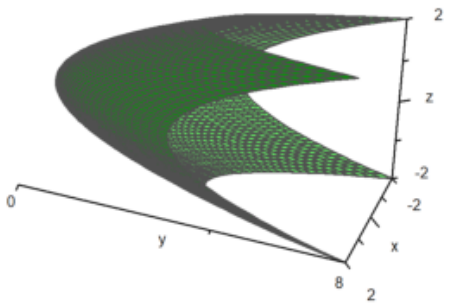



```
>plot3d("x*sin(x)",a=0,b=6pi,rotate=2):
```



Berikut adalah plot dengan tiga fungsi.

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



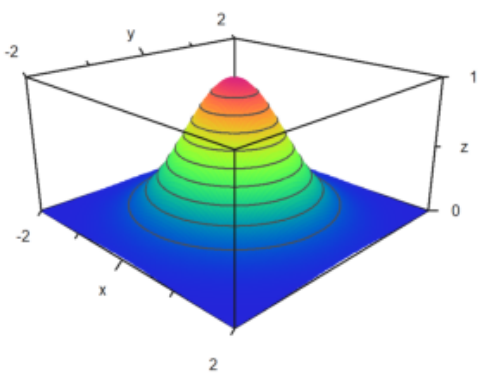

## Plot Kontur

Untuk plot, Euler menambahkan garis grid atau garis kisi-kisi. Sebagai gantinya, dimungkinkan untuk menggunakan garis level dan rona satu warna atau spectral rona warna. Euler dapat menggambar ketinggian fungsi dalam plot dengan menggunakakn bayangan. Dalam semua plot 3D Euler dapat menghasilkan anaglyph merah/can.
- >hue: Untuk mengaktifkan shading atau bayangan, bukan wires
- >contour: Untuk memplot garis kontur secara otomatis ke dalam plot yang dibuat
- level=...(or level): Vektor nilai untuk garis kontur.

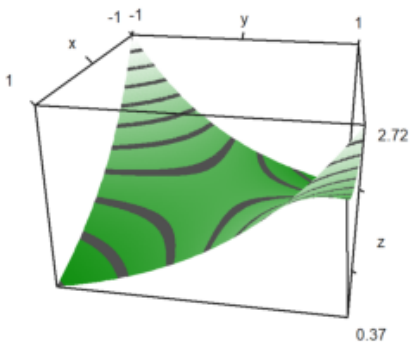
Default untuk level adalah level="auto", yang mana nantinya perintah ini akan mengkomputasikan beberapa garis dengan level yang berbeda secara otomatis. Seperti yang terlihat dalam plot, level sebenarnya adalah rentang level.

Gaya default dapat diubah. Untuk plot kontur berikut, kita akan menggunakan grid yang lebih halus untuk 100x100 titik, skala fungsi dan plot, dan menggunakan sudut penglihatan 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°):
```

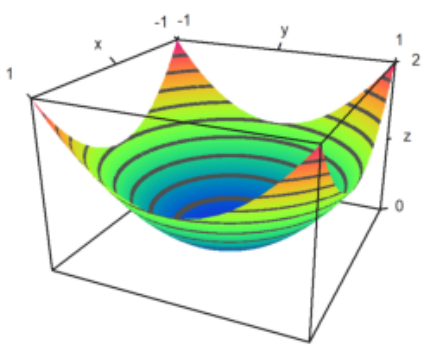


```
>plot3d("exp(x*y)",angle=100°,>contour,color=green):
```



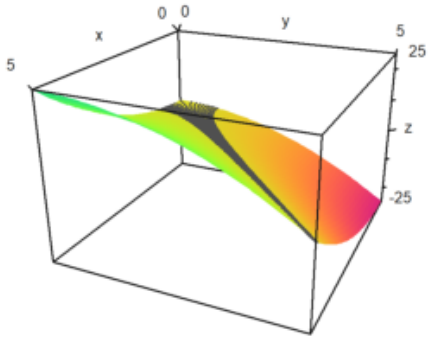

Untuk shading, defaultnya adalah menggunakan wana abu-abu. Tetapi warna ini dapat diubah karena interval warna atau interval spectral dapat digunakan.
- >spectral: Untuk skema default spectral
- color: Untuk menambahkan warna tetentu atau skema spectral tertentu
Untuk contoh berikut, kita menggunakan skema default spectral dan memperkecil nilai n untuk mendapatkan tampilan yang lebih halus.

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



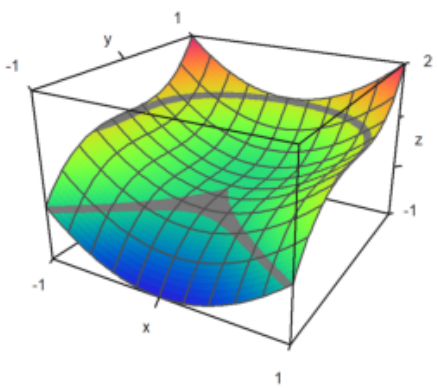

Alih-alih garis level otomatis, kita akan mengatur nilai-nilai dari garis level. Hal ini akan menghasilkan garis level yang lebih tipis alih-alih interval level.

```
>plot3d("x^2-y^2",0,5,0,5,level=-1:0.1:1,color=redgreen):
```




Dalam plot berikut, kita akan menggunakan 2 level yang sangat luas dari -0.1 sampai 1, dan dari 0.9 sampai 1.
Level berikut akan dimasukkan sebagai matriks dengan batas level sebagai kolom.
Selain itu, kita membentangkan grid dengan 10 interval dalam setiap arah.

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

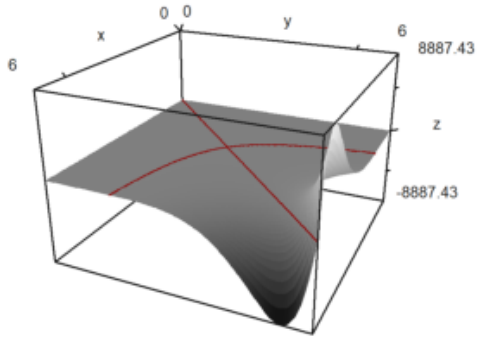



Dalam contoh berikut, kita memplot set, dimana

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

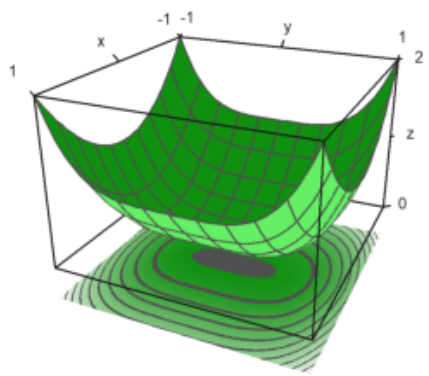
Kita menggunakan garis tunggal tipis untuk garis levelnya.

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



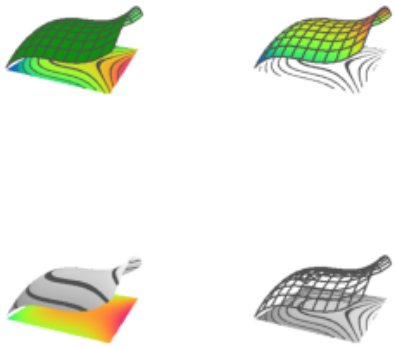

Pada Euler, kita dapat menampilkan garis konturnya di bawah plot. Warna dan jarak ke plot dapat diatur secara spesifik nilainya.

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



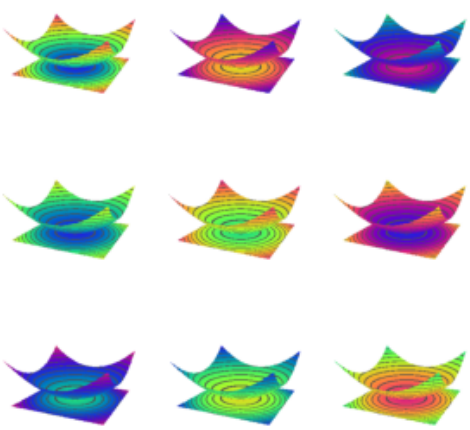

Berikut adalah beberapa contoh mengenai gaya. Kita selalu menonaktifkan frame, dan menggunakan warna tertentu untuk plot dan gridnya.

```
>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):
```



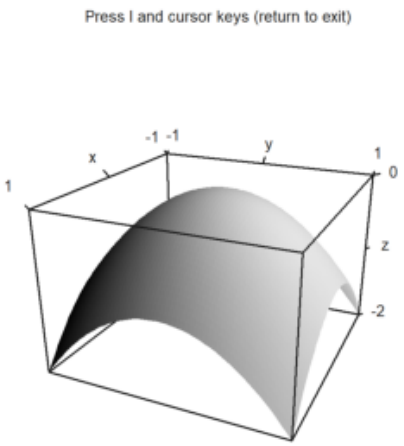
Ada beberapa skema spectral yang lain, yaitu yang dinomori dari 1 hingga 9. Tetapi kalian juga dapat menggunakan perintah color=value, dimana value:
- spectral: interval dari biru ke merah
- white: untuk warna yang lebih halus
yellowblue,purplegreen,blueyellow,greenred
- blueyellow,greenpurple,yellowblue,redgreen

```
>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 dengan kursor kunci selama interaksi pengguna. Hal itu juga dapat diatur dengan parameter.
- light: arah untuk cahaya
- amb: cahaya disekitar antara 0 dan 1
Ingat bahwa program tidak membuat beda sisi dari plot. Jika tidak ada shadows atau bayangan, maka kalian perlu menggunakan 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)"):
```

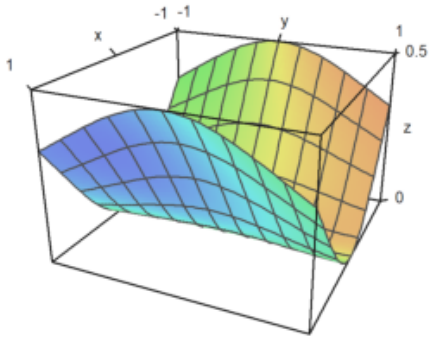



Parameter warna mengubah warna permukaan. Warna garis level juga dapat diubah.

```
>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):
```

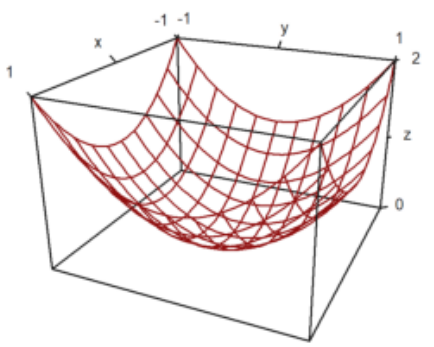
color=0 akan memberikan efek warna spesial, yaitu efek warna pelangi.

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




Kita juga dapat mengatur permukaan plotnya supaya transaparan.

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




## Plot Implisit

Dalam 3D juga terdapat fungsi implisit. Euler akan menghasilkan potongan melalui objek. Fitur dari plot3d juga termasuk plot implisit. Plot ini menampilkan set nol dari fungsi tiga variabel. Solusi dari

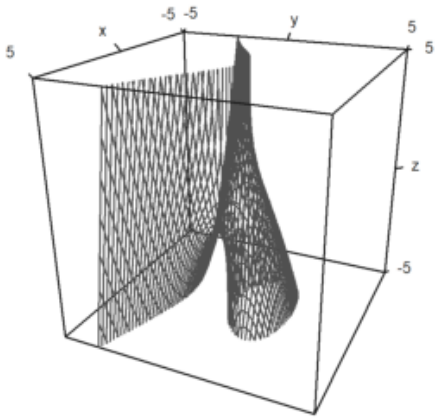
$$f(x, y, z) = 0$$

dapat divisualisasikan dengan potongan yang sejajar dengan bidang xy, bidang xz, dan bidang yz.
- implicit=1: artinya potongan akan sejajar dengan bidang yz
- implicit=2: artinya potongan akan sejajar dengan bidang xz
- implicit=4: artinya potongan akan sejajar dengan bidang xy

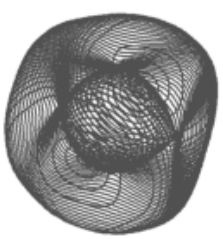
Tambahakn nilai ini jika kalian mau. Dalam contoh berikut kita akan memplot

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

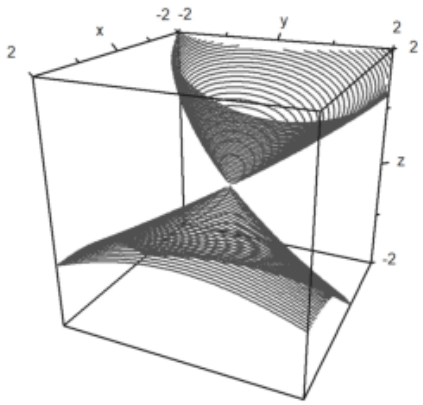
```
>plot3d("x^2+y^3+z*y-1",r=5,implicit=3):
```




```
>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 juga dapat menerima data. Untuk objek 3D, kalian perlu menyiapkan matriks nilai x,y,dan z, atau 3 fungsi atau ekspresi fx(x,y),fy(x,y),fz(x,y).

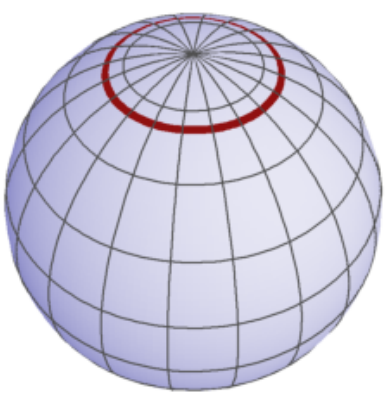
$$\gamma(t,s) = (x(t,s), y(t,s), z(t,s))$$

Karena x,y,dan z adalah matriks, kita mengasumsikan bahwa (t,s) dijalankan dengan grid persegi. Hasilnya, kita akan memplot gambar dari segitiga

Kalian dapat menggunakan bahasa matriks dalam Euler untuk memproduksi titik koordinat ini secara efektif.

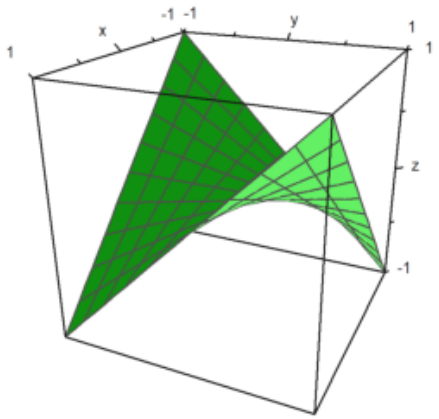
Dalam contoh berikut, kita akan menggunakan cektor dari nilai t dan kolom vektor dari nilai s untuk parameter permukaan dari bola. Saat menggambar, kita dapat menandai bagian tertentu, dalam kasus kita, kita menandai bagian polarnya.

```
>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, dimana grafiknya adalah sebuah fungsi.

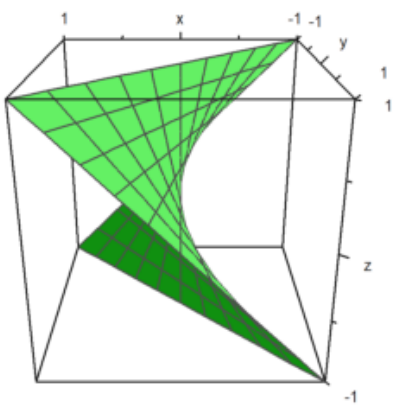
```
>t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,t*s,grid=10):
```




Bagaimanapun, kita dapat membuat berbagai jenis permukaan. Berikut adalah permukaan yang sama dengan fungsi

$$x = y\,z$$

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




Dengan usaha yang lebih, kita dapat membuat berbagai jenis permukaan.

Dalam contoh berikut kita akan membuat tampilan bayangan dari bola berubah. Koordinat bola biasanya adalah

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

dengan

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

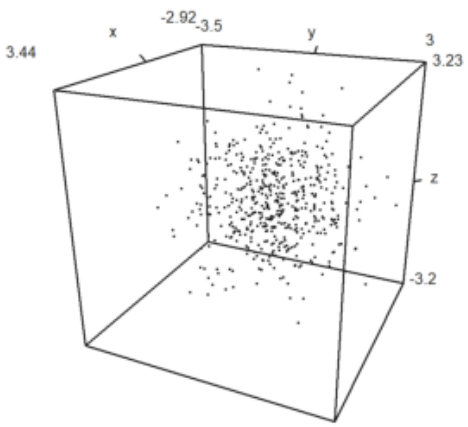
Kita akan mengubah ini dengan faktor

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

```
>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):
```

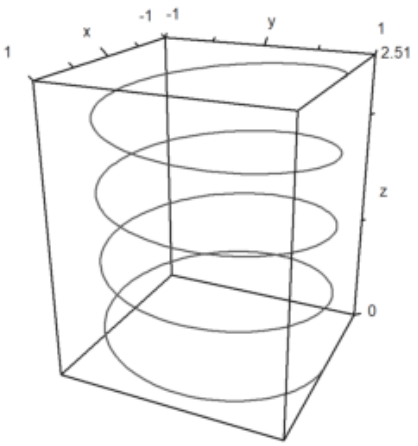
Tentu saja titik awan juga memungkinkan. Untuk memplot titik data di ruang, kita membutuhkan tiga vektor untuk koordinat titikya.
Untuk mengaturnya masih sama dengan pada plot2d, yaitu dengan points=true;

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

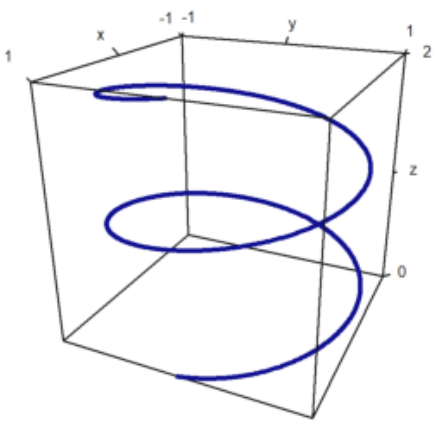


Kita juga dapa memplot kurva di 3D. Dalam kasus ini, akan lebih mudah untuk mengkomputasikan terlebih dahulu titik dari kurvanya. Untuk kurva dalam bidang, kita akan menggunakan deretan dari koordinat dan parameter wire=true.

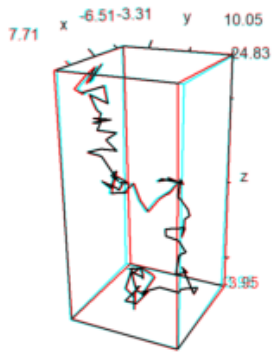
```
>t=linspace(0,8pi,500); ...
>plot3d(sin(t),cos(t),t/10,>wire,zoom=3):
```




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

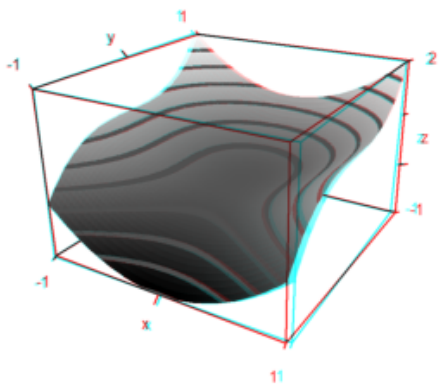



```
>X=cumsum(normal(3,100)); ...
> plot3d(X[1],X[2],X[3],>anaglyph,>wire):
```



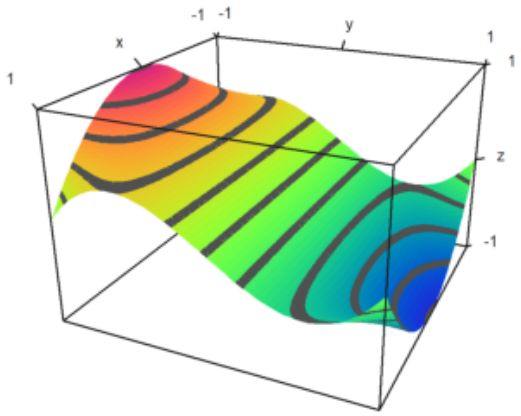

EMT juga dapat memplot dalam anaglyph mode. Untuk melihat plot mode anaglyph, kalian membutuhkan kacamata red/cyan.

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




Seringkali, skema warna spectral digunakan untuk plot. Hal ini menekankan ketinggian dari fungsinya.

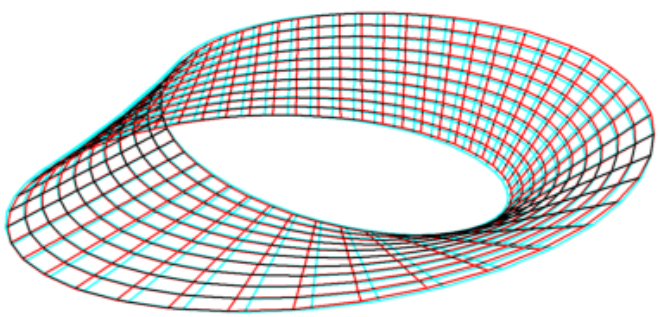
```
>plot3d("x^2*y^3-y",>spectral,>contour,zoom=3.2):
```




Euler juga dapat memplot parameter permukaan, saat parameternya adalah nilai x,y,dan z dari sebuah gambar dalam bentuk grid persegi panjang di ruang 3D.

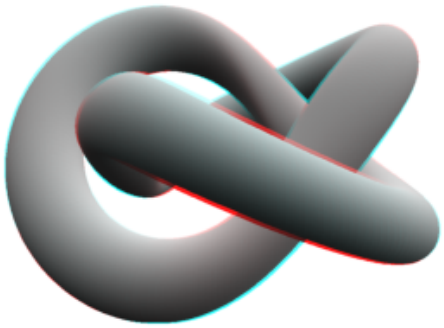
Untuk ocntoh berikut, kita mengeset parameter u dan v, dan menghasilkan koordinat ruang dari parameter tadi.

```
>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, dengan kacamata rec/cyan yang majestik.

```
>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

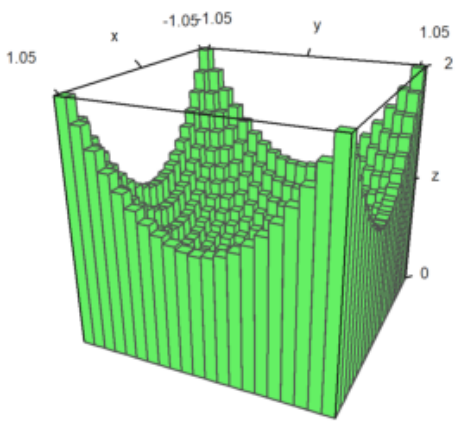
Bar plot juga dapat dihasilkan menggunakan EMT. Untuk membuatnya, kita perlu menyediakan

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

z dapat lebih besar, namun hanya nilai nxn yang akan digunakan.

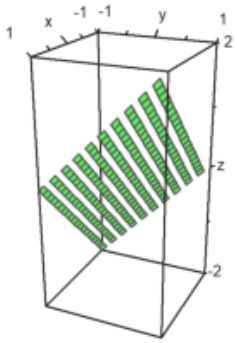
Dalam contoh berikut, kita lebih dulu mengkomputasi nilai. Lalu kita menyesuaikan x dan y sehingga pusat vektor dalam nilai digunakan.

```
>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):
```



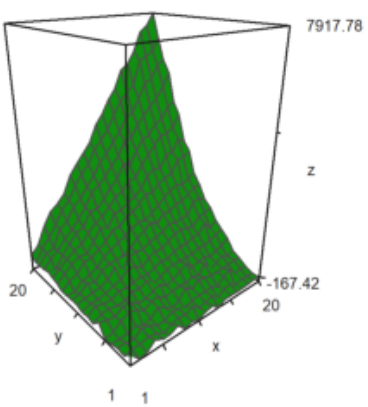

Kita juga dapat membagi permukaan plotnya dalam dua atau lebih bagian.

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

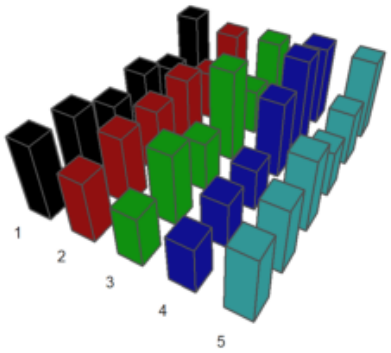



Jika memuat atau menghasilkan matriks data M dari file dan perlu memplotnya dalam 3D, kalian dapat mengatur skala matriks ke [-1,1] dengan scale(M), atau mengatur skala matriks dengan >zscale. Hal ini dapat dikombinasikan dengan faktor penskalaan individual yang diterapkan sebagai tambahan.

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

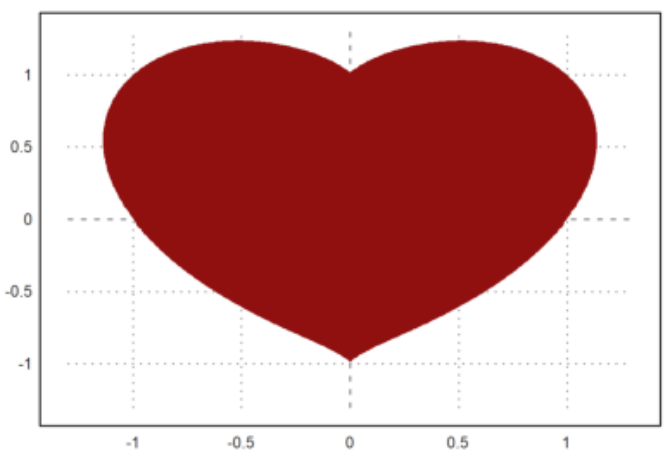



```
>Z=intrandom(5,100,6); v=zeros(5,6); ...
>loop 1 to 5; v[#]=getmultiplicities(1:6,Z[#]); end; ...
>columnsplot3d(v',scols=1:5,ccols=[1:5]):
```




## 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 kurva bentuk hati tersebut di sekitar sumbu y. Berikut adalah ekspresinya, yang mendefinisikan kurva bentuk hati:

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

Selanjutnya kita mengatur:

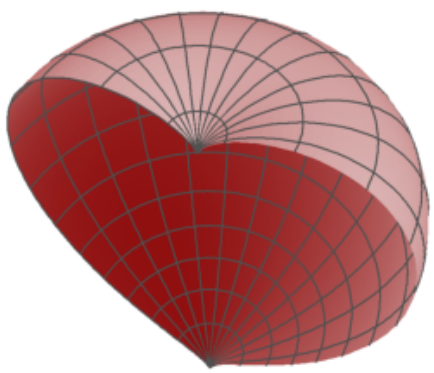
$$x = r.cos(a), \quad y = r.sin(a).$$

```
>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 berlaku juga untuk fungsi numerik, yang menyelesaikan r jika diberikan a. Dengan fungsi tersebut kita dapat memplot bentuk hati yang akan 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°):
```

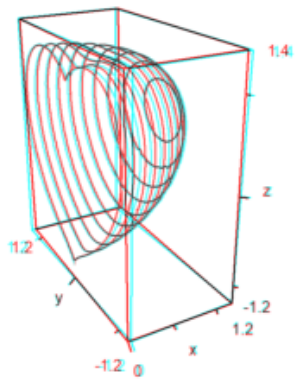


Berikut adalah plot 3D dari bentuk di atas, yang diputar mengelilingi sumbu z. Kita mendefinisikan fungsinya, yang mendeskripsikan objek tersebut.

```
>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=[10,100,60],>anaglyph):
```




## Plot 3D Khusus

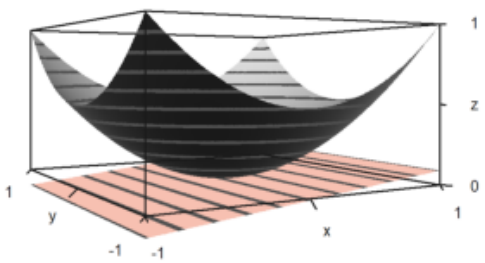
Fungsi plot3d memang bagus, tapi fungsi tersebut belum dapat memenuhi semua yang kita butuhkan. Selain hal-hal dasaar, fungsi tersebut memungkinkan untuk menambahkan frame pada semua objek yang kalian suka.

Meskipun Euler bukan program 3D, dia dapat menggabungkankan beberapa objek dasar. Kita akan mencoba untuk memvisualisasikan paraboloid 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
```

Fungsi framedplot() memungkinkan kita untuk menambahkan frame dan mengatur viewsnya.

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

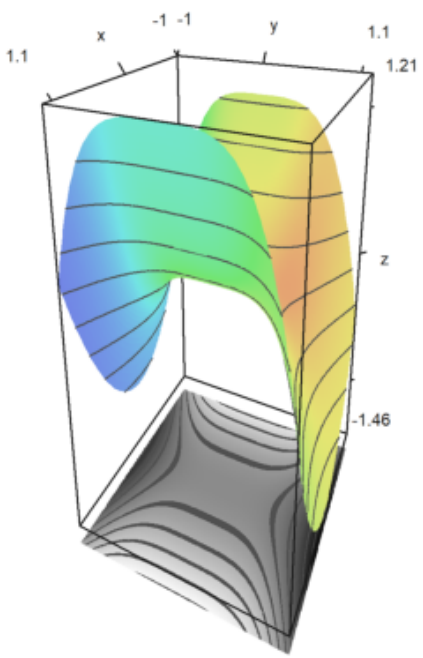


Dengan cara yang sama, kalian dapat memplot bidnag konturnya secara manual. Ingat bahwa plot3d() mengatur jendela kek fullwindow() secara default, tetapi plotcontourplane() mengasumsikannya.

```
>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

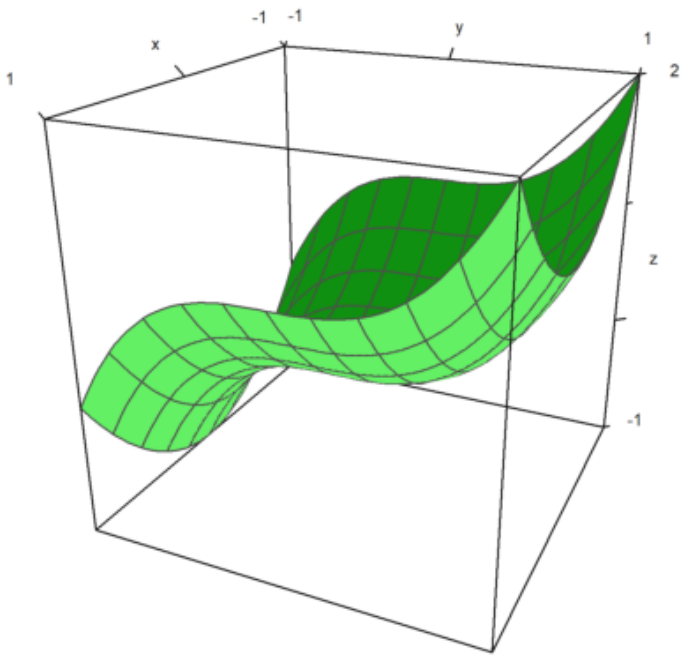
Euler dapat menggunakan frame untuk memperkirakan animasi.

Salah satu fungsinya yang menggunakan teknik ini adalah rotasi. Fungsi ini dapat mengubah sudut pandang adn menggambar ulang plot 3D. Fungsi ini memanggil addpage() untuk setiap plot baru. Akhirnya fungsi ini menganimasikan plot tersebut.

Silakan pelajari sumber rotate untuk melihat lebih detail.

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

```
Press space to stop, return to end
```




## Menggambar Povray

Dengan bantuan file Euler povray, Euler dapat menghasilkan file Povray. Hasilkan akan sangat indah untuk dipandang.

Kalian perlu menginstall Povray (32bit atau 64bit) dari http://www.povray.org/, dan meletakkan sub direktori "bin" dari Povray ke dalam environment, atau mengatur variabel "defaultpovray" dengan jalur lengkap yang mengarah ke "pvengine.exe".

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

Fungsi pov3d memiliki semangat yang sama dengan plot3d. Fungsi ini dapat menghasilkan grafik dari sebuah fungsi f(x,y), atau sebuah permukaan dengan koordinat X,Y,Z dalam bentuk matriks, termasuk garis-garis level yang bersifat 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-fungsi ini, mulai file Povray dengan povstart(). Kemudian gunakan writeln(...) untuk menulis objek ke file scene. Terakhir, akhiri file dengan povend(). Secara default, raytracer akan dimulai, dan PNG akan dimasukkan ke dalam buku catatan 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. Fungsi ini memiliki parameter untuk warna, transparansi, Phong Shading, dll.

Perhatikan bahwa Povray universe memiliki sistem koordinat lain. Antarmuka ini menerjemahkan semua koordinat ke sistem Povray. Jadi Anda dapat tetap berpikir dalam sistem koordinat Euler dengan z yang mengarah vertikal ke atas, dan sumbu x, y, z di tangan kanan.
nda perlu memuat file povray.
Translated with DeepL.com (free version)

```
>load povray;
```

Pastikakn direktori bin Povray berada di jalur yang benar. Jika tidak, edit variabel berikut sehingga berisi jalur ke eksekusi povray.

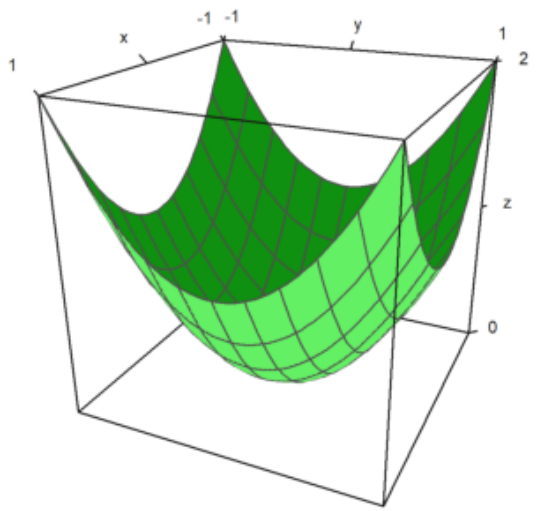
```
>defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

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

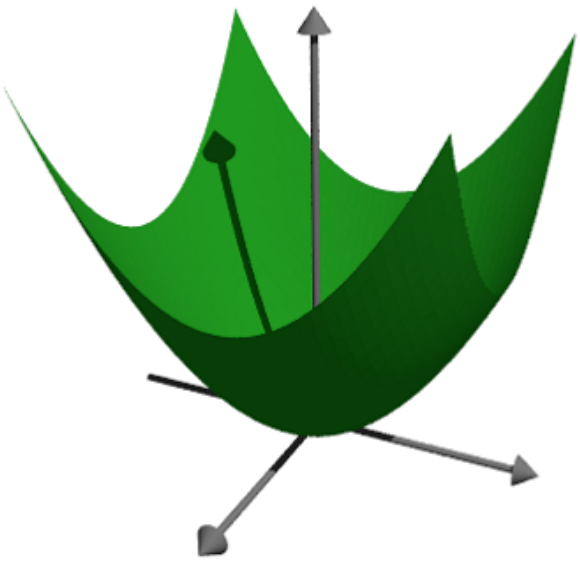
Untuk kesan pertama, kita akan memplot fungsi yang sederhana. Perintah berikut akan menghasilkan file povray di direktori user kalian, dan menjalankan Povray untuk melacak sinar pada file ini.

Jika kalian menjalankakn perintah berikut, Povray GUI seharusnya terbuka, menjalankan file, dan tertutup secara otomatis. Karena hal keamanan, kalian akan ditanya, jika kaliani ingin mengizinkan file exe untuk dijalankan. Kalian dapat mengklik cancel jika ingin menghentikan pertanyaan berikutnya. Kalian perlu mengklik OK di jendela Povray untuk mengetahu dialog awal Povray.

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

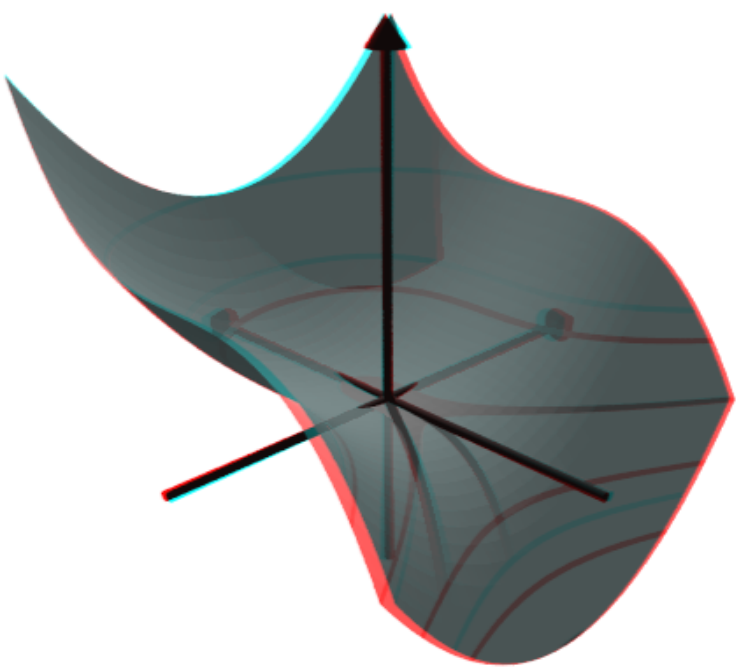



```
>pov3d("x^2+y^2",zoom=3);
```



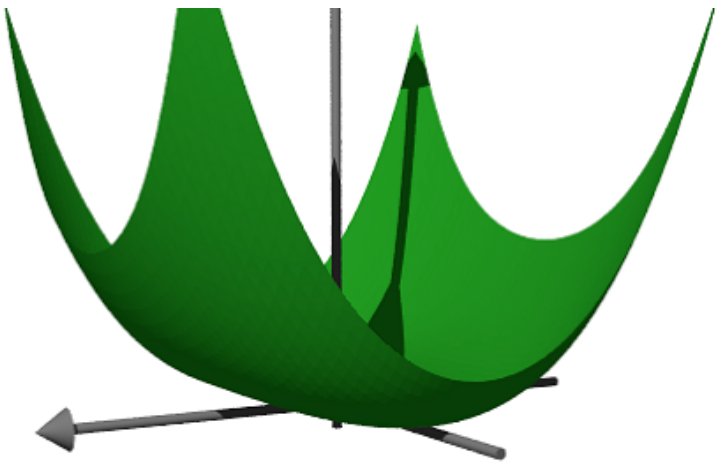
Kita dapat membuat fungsi menjadi transparan dan menambahkan beberapa hal. Kita juga dapat menambahkan level garis untuk plot fungsi.

```
>pov3d("x^2+y^3",axiscolor=red,angle=-45°,>anaglyph, ...
>  look=povlook(cyan,0.2),level=-1:0.5:1,zoom=3.8);
```



Terkadang kita perlu untuk mencegah penskalaan fungsi, dan menskalakan fungsi dengan tangan.
Kita memplot kumpulan titik pada bidang kompleks, dimana hasil kali jarak ke 1 dan -1 sama dengan 1.

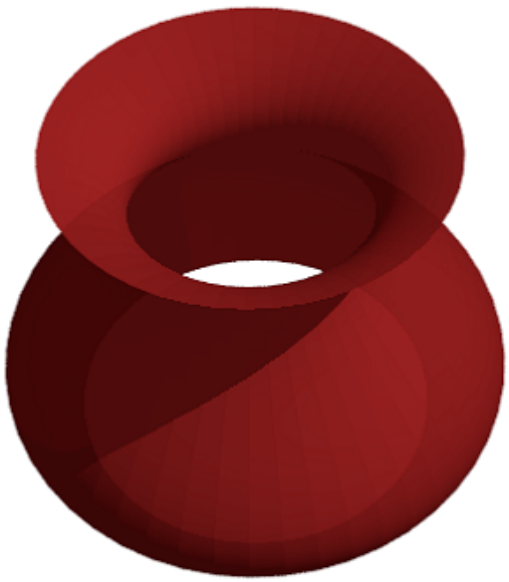
```
>pov3d("((x-1)^2+y^2)*((x+1)^2+y^2)/40",r=2, ...
>  angle=-120°,level=1/40,dlevel=0.005,light=[-1,1,1],height=10°,n=50, ...
>  <fscale,zoom=3.8);
```



## Memplot dengan Koordinat

Alih-alih dengan fungsi, kita dapat memplot dengan koordinat. Seperti dalam plot3d, kita membutuhkan tiga matriks untuk mendefinisikakn suatu objek.
Sebagai contoh kita memutar fungsi pada sumbu z.

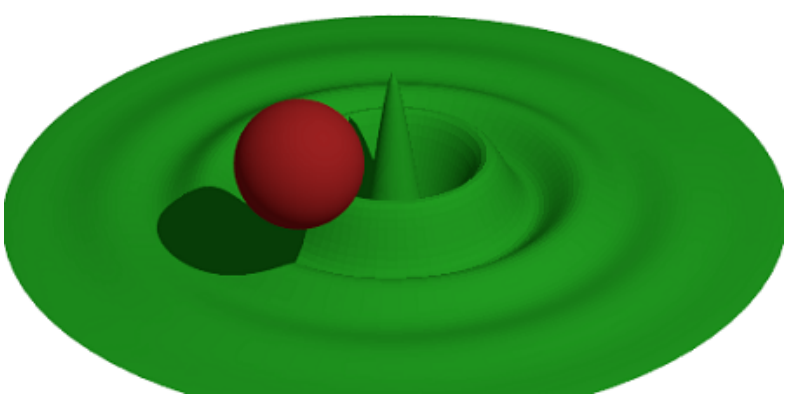
```
>function f(x) := x^3-x+1; ...
>x=-1:0.01:1; t=linspace(0,2pi,50)'; ...
>Z=x; X=cos(t)*f(x); Y=sin(t)*f(x); ...
>pov3d(X,Y,Z,angle=40°,look=povlook(red,0.1),height=50°,axis=0,zoom=4,light=[10,5,15]);
```



Pda contoh berikut, kita memplot gelombang teredam. Kita menghasilkan gelombang dengan bahasa matriks Euler.

Kita juga menunjukkan bagaimana objek tambahan dapat ditambahkan ke adegan pov3d. Untuk menghasilkan objeknya, perhatikan contoh berikut. Ingat bahwa plot3d mengatur plot sehingga plot yang dihasilkan akan sesuai 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=6,axis=0,height=30°,add=povsphere([0.5,0,0.25],0.15,povlook(red)), ...
>  w=500,h=300);
```



Dengan metode bayangan canggih Povray, hanya sedikit titik yang bisa menghasilkan permukaan yang sangat halus. Hanya pada batas-batas dan bayangan, trik ini bisa terlihat jelas.

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

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

$$x^2 y^3$$

Persamaan untuk permukaannya adalah [x,y,z]. Kita akan menghitung dua turunan terhadap x dan y dari persamaan ini lalu mengambil hasil cross produknya sebagai normal.

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

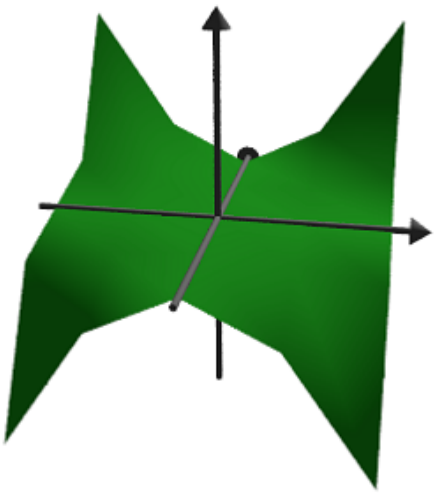
Kita mendefinisikan vektor normal sebagai cross produk dari turunan ini, dan mendefinisikan koordinat fungsinya.

```
>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 akan 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 dibuat oleh A. Busser di Povray. Ada versi yang lebih baik dai ini dalam contoh.

See: Examples\Trefoil Knot | Trefoil Knot

Untuk tampilan yang bagus dengan tidak terlalu banayk titik, kita tambahkan vektor normal di sini. Kita menggunakan Maxima untuk menghitung normal untuk kita. Pertama, tiga fungsi untuk 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 dua turunan vektor untuk x dan y.

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

Selanjutnya normal, yang merupakan corss produk dari dua turunan.

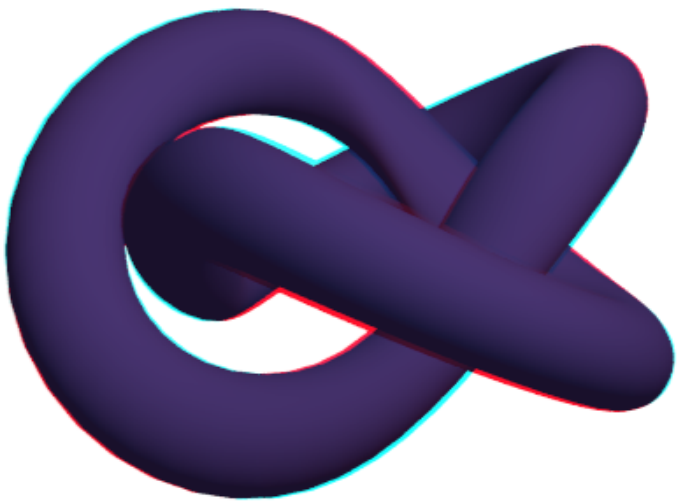
```
>dn &= crossproduct(dx,dy);
```

Selanjutny kita menghitung semua tadi secara numerik.

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

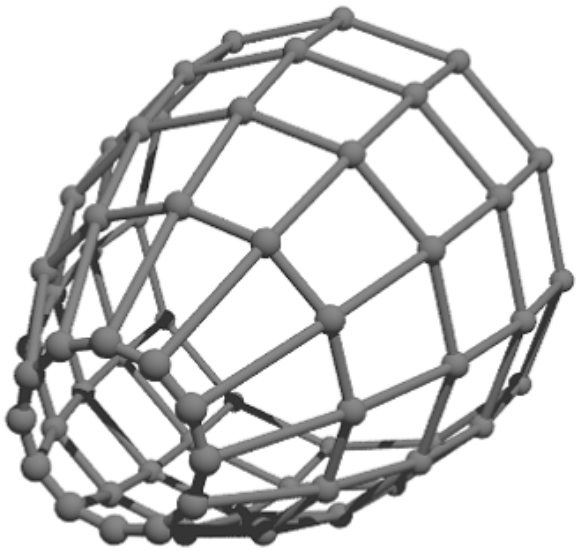
Vektor normal adalah evaluasi dari ekspresi simbolik dn[i] untuk i=1,2,3. Sintaks untuk ini adalah &"ekspresi"(parameter). Ini adalah sebuah alternatif dari metode pada contoh sebelumnya, dimana kita mendefinisikan ekspresi simbolik NX, NY, NZ terlebih dahulu.

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



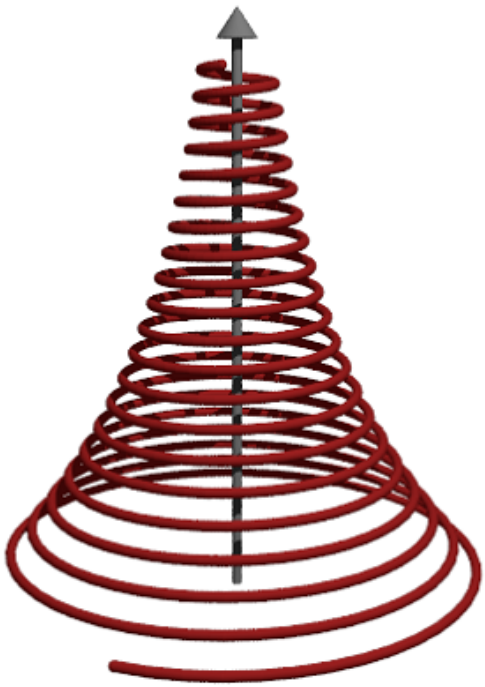
Kita juga dapat menghasilkan grid di 3D.

```
>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 adalah hal yang mungkin.

```
>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();
```



## Povray Objek

Di atas, kita sudah 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 definisikan tiga silinder, dan menyimpannya di string pada Euler.
Fungsi povx() etc. hanyak enghasilkan vektor [1,0,0], yang dapat digunakan sebagai gantinya.

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

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

```
>c2
```

```
cylinder { <0,0,-1>, <0,0,1>, 1
 texture { pigment { color rgb <0.941176,0.941176,0.392157> }  }
 finish { ambient 0.2 }
 }
```

Seperti yang terlihat, kita menambahkan tekstur ke dalam objek dengan 3 warna yang berbeda.

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

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

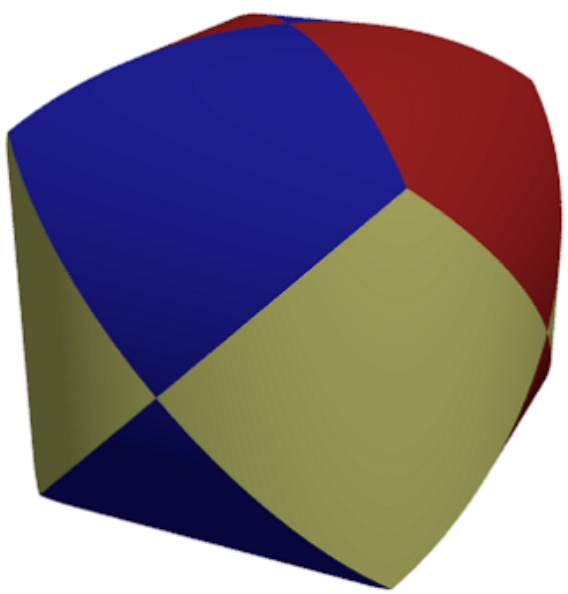
```
   texture { pigment { color rgbf <0.101961,0.2,0.301961,0.1> }  }
   finish { ambient 0.5 }
```

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

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

Perpotongan dari tiga silinder sulit untuk divisualisasikan, jika kalian tidak pernah melihatnya sebelumnya.

```
>povend;
```



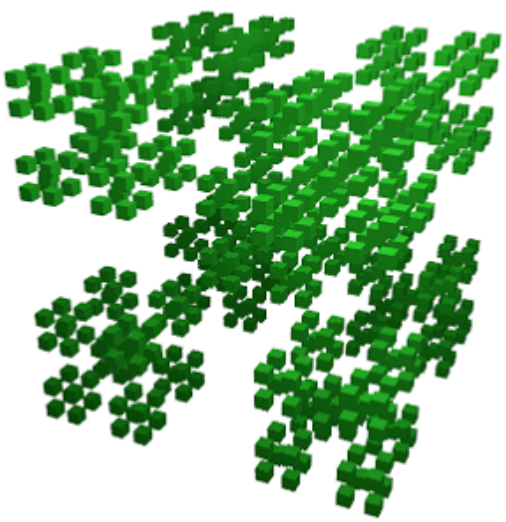
Fungsi berikut menghasilkan fraktal secara rekursif.

Fungsi yang pertama memperlihatkan, bagaimana Euler mengatasi objek sederhana Povray. Fungsi povbox() mengembalikan sebuah string, memuat 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 pemotongan satu objek dari objek lainnya. Seperti persimpangan, ada bagian dari objek CSG Povray.

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

Pada demonstrasi ini, kita mendefinisikan objek di Povray, alih-alih menggunakan string di Euler. Definisinya akan tertulis di file secara langsung.

Kotak koordinat dari -1 artinya [-1,-1,-1].

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

Kita dapat menggunkan objek ini di povobject(), yang akan mengembalikan string seperti biasanya.

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

Kita menghasilkan kubus kedua, dan merotasi dan menskalanya sedikit.

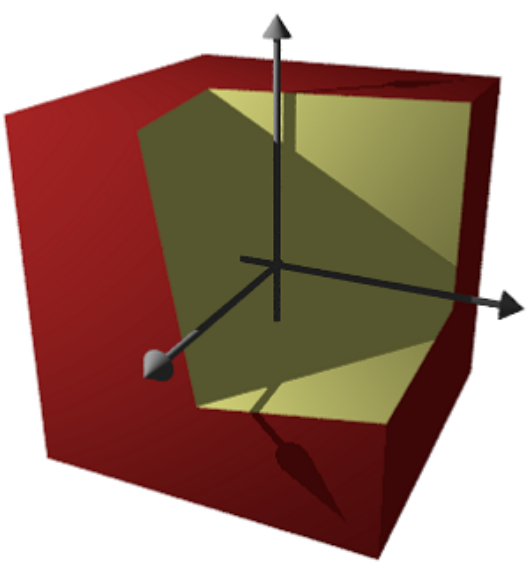
```
>c2=povobject("mycube",povlook(yellow),translate=[1,1,1], ...
>  rotate=xrotate(10°)+yrotate(10°), scale=1.2);
```

Lalu kita mengambil perbedaan dari kedua objek.

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

Selanjutnya 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 dimana f(x,y,z)=0, sama seperti parameter implisit pada plot3d. Hasilnya akan terlihat lebih baik.

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

$$((x^2+y^2-c^2)^2+(z^2-1)^2)*((y^2+z^2-c^2)^2+(x^2-1)^2)*((z^2+x^2-c^2)^2+(y^2-1)^2)=d$$

```
>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();
```

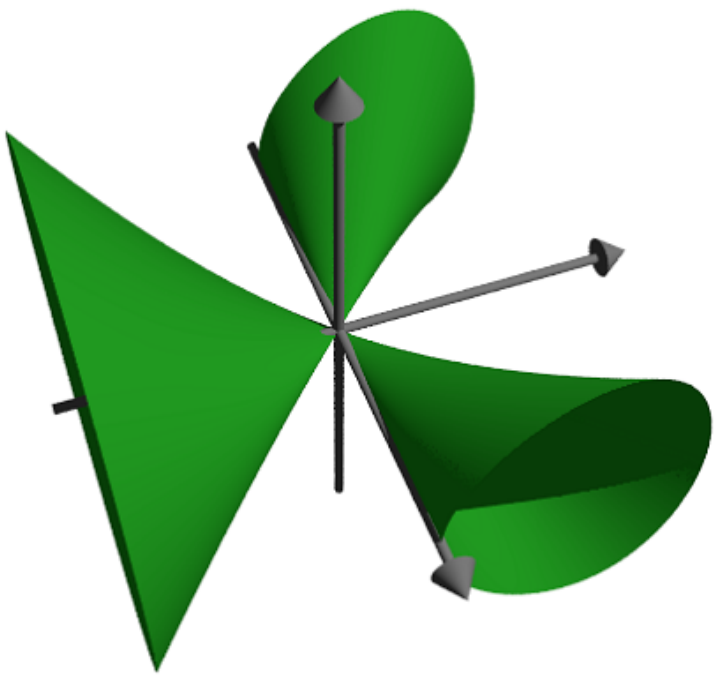
```
  Error : Povray error!

  Error generated by error() command

  povray:
      error("Povray error!");
  Try "trace errors" to inspect local variables after errors.
  povend:
      povray(file,w,h,aspect,exit);
```

```
>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);
```

Membuat permukaan implisit. Ingat perbedaan sintaks di ekspresi.

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



## Objek Jaring

Pada contoh berikut, kita akab menampilkan bagaimana cara membuat objek jaring, dan menggambarnya dengan informasi tambahan.

Kita ingin memaksimalkan xy, dibawah kondisi x+y=1 dan mendemonstrasikan sentuhan tangensial dari garis level.

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

Kita tidak dapat menyimpan objek pada string seperti sebelumnya, karena ukurannya terlalu besar. Jadi kita defnisiikan objek di file Povray menggunakan declare. Fungsi povtriangle() memporses hal ini secara otomatis. Fungsi ini dapat menerima vektor normal sama seperti pov3d().

Berikut definisi dari objek jaring dan menulisnya secara langsung di 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 tentukan 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);
```

Tuliskan permukaan dikurangi kedua cakram.

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

Tulis kedua perpotongan tersebut.

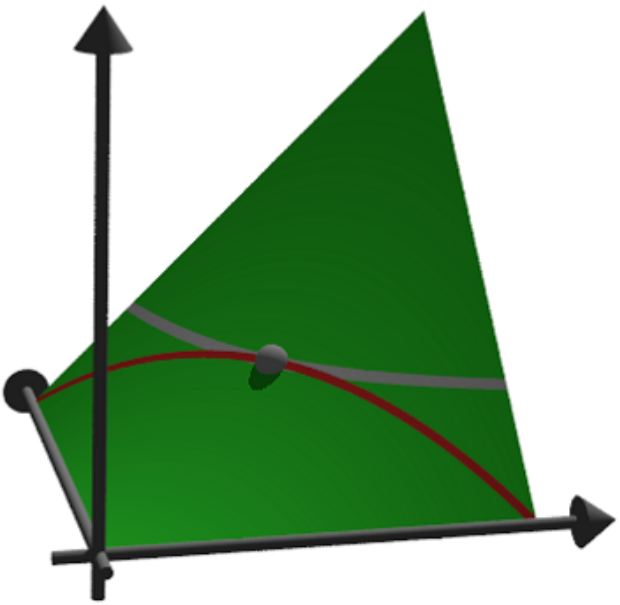
```
>writeln(povintersection([mesh,cl],povlook(red)); ...
>writeln(povintersection([mesh,ll],povlook(gray)));
```

Tuliskan satu titik secara maksimal.

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

Tambahkan sumbu dan akhiri

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



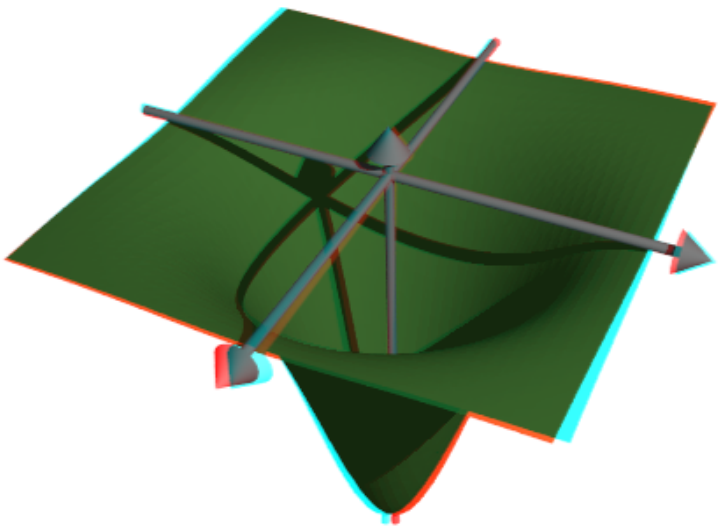
## Anaglyphs pada Povray

Untuk menghasilkan anaglyph bagi kacamata red/cyan, Povray harus dijalankan sebanyak dua kali dari posisi kamera yang berbeda. Hal itu menghasilkan dua file Povray dan dua file PNG, yang akan diproses dengan fungsi loadanaglyph().

Tentu saja, kalian perlu kacamata rec/cyan untuk melihat contoh berikut secara benar.

Fungsi pov3d() memiliki tombol sederhana untuk menghasilkan anaglyph.

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



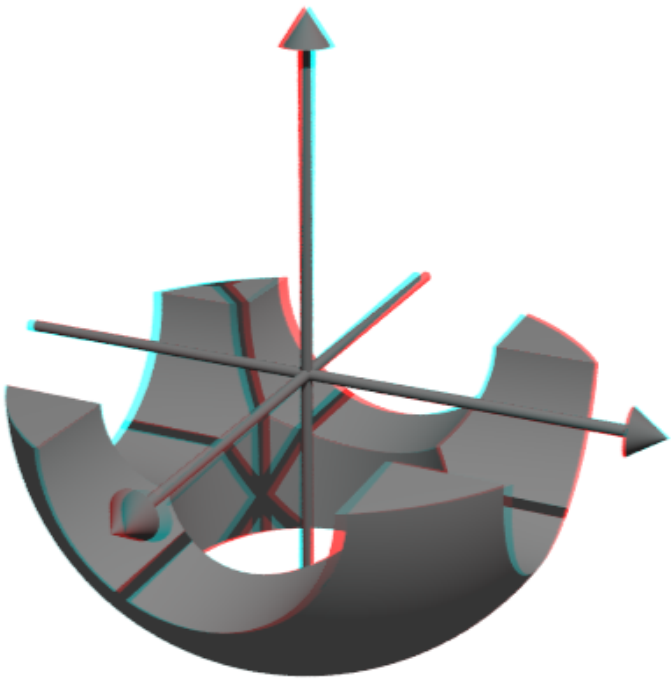
Jika kalian membuat scene dengan objek, kalian harus menempatkan pembuatan scene ke dalam suatu fungsi, dan menjalankannya dua kali dengan nilai yang berbeda untuk parameter anaglyph.

```
>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 pada povstart() dan povend() yang digabungkan.

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



## Mendefinisikan Objek Sendiri

Antarmuka Povary Euler berisi banyak objek. Namun kalian tidak dibatasi pada objek-objek tersebut. Kalian dapat membuat objek sendiri, yang menggabungkan objek-objek lain, atau objek yang benar benar baru.

Kami mendemonstrasikan sebuah torus. Perinth Povray untuk ini adalah "torus". Jadi kita mengembalikan sebuah string dengan perintah ini dan parameternya. Pehatikan bahwa torus selalu berpusat pada titik asal.

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

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

Berikut adalah torus pertama kita

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

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

Mari kita gunakan objek ini untuk membuat torus kedua, ditransasikan 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>
 }
```

Now we place these objects into a scene. For the look, we use 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, program ini tidak menampilkan kesalahan. Oleh karen aitu, kalian harus menggunakan

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

jika ada yang tidak berhasil. Ini akan membiarkan jendela Povray tetap terbuka.

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

Berikut adalah contoh yang lebih rumit. Kita menyelesaikan

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

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 cek jika contoh ini mempunyai solusi semua.

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

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

Ternyata, contoh tersebut memiliki solusi.
Selanjutnya kita definisikan dua objek. Yang pertama merupakan bidang

$$a \cdot x \leq b$$

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

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

Lalu kita definisikan perpotongan dari semua setengah ruang dan 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
```

Sekarang kita dapat memplot scene

```
>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 adalah lingkaran di sekitar optimum.

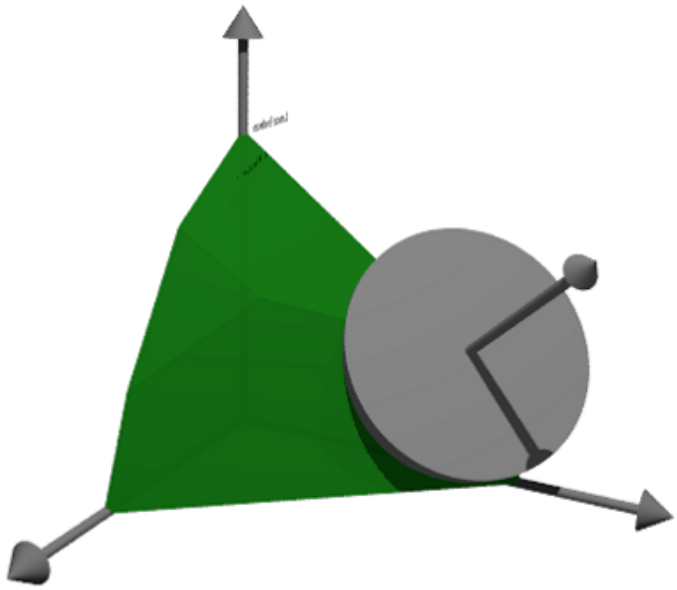
```
>writeln(povintersection([povsphere(x,0.5),povplane(c,c.x')], ...
>  povlook(red,0.9)));
```

Dan kesalahan pada arah yang optimal.

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

Kita menambahkan teks ke layar. Teks hanyalah sebuah objek 3D. Kita perlu menempatkan dan memutarnya sesuai dengan pandangan kita.

```
>writeln(povtext("Linear Problem",[0,0.2,1.3],size=0.05,rotate=5°)); ...
>povend();
```



## Lebih banyak contoh

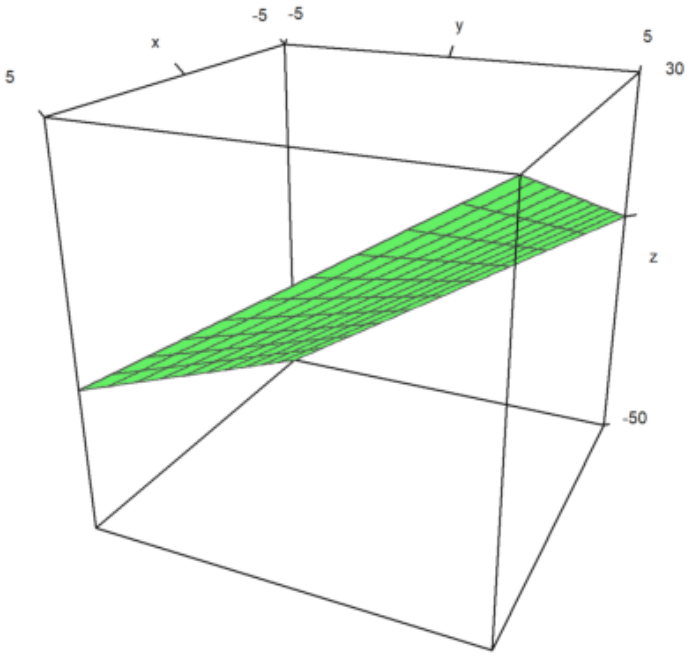
Kalian dapat menemukan lebih banyak contoh untuk Povray di Euler pada file berikut.

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

## Soal Tambahan 1. Gambarlah bidang

$$f(x,y) = 3x + 5y - 10$$

```
>plot3d("3*x+5*y-10",r=5):
```

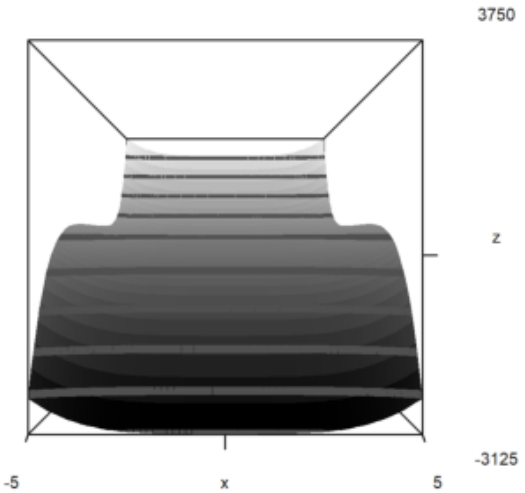



2. Gambarlah bidang

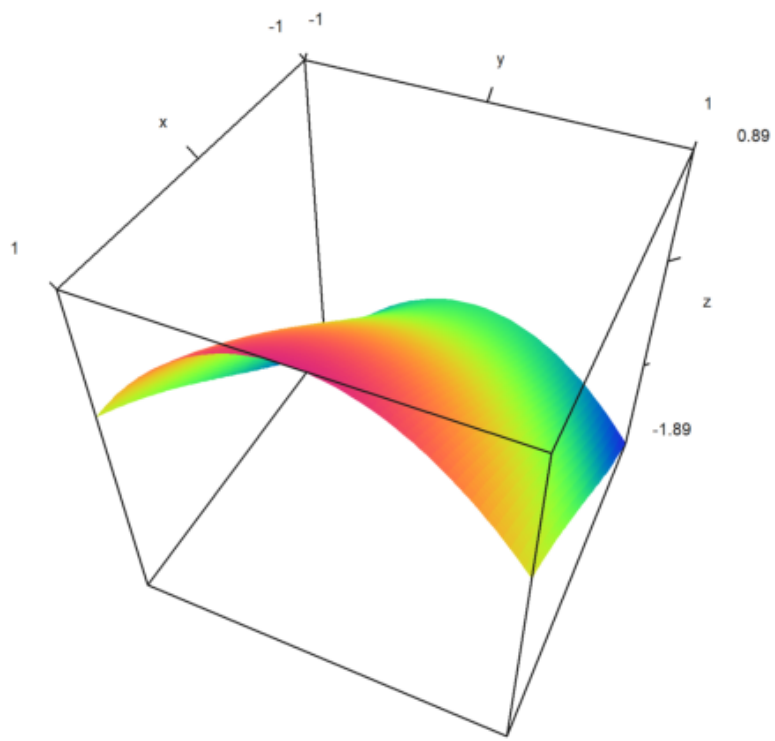
$$f(x,y) = x^4 + y^5)$$

dengan distance = 3, zoom = 1, angle dan height = 0, dan aktifkan fitur kontur

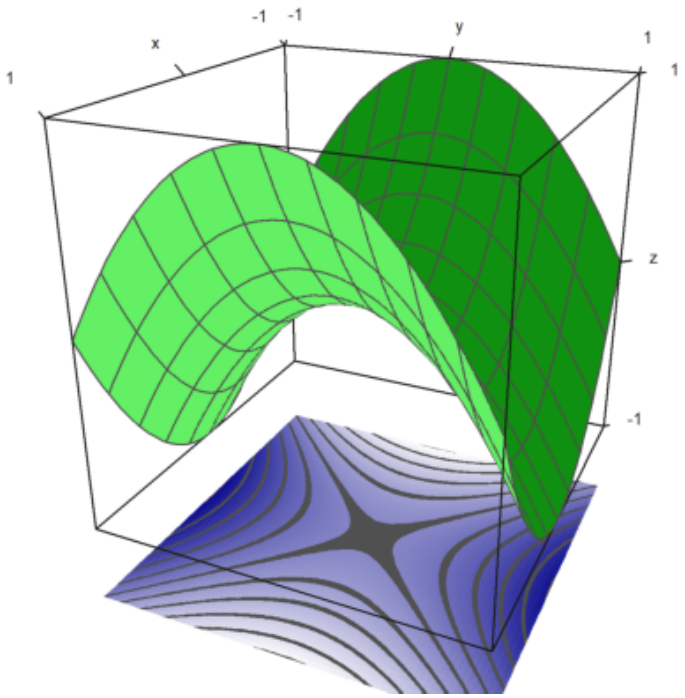
```
>plot3d("x^4+y^5",r=5,distance=3,zoom=1,angle=0,height=0,>contour):
```




```
>plot3d("x-(x^3/9)-y^2",height=45°,center=[0,0,0],>spectral):
```



```
>plot3d("x^2-y^2",>cp,cpcolor=blue,cpdelta=0.2):
```




```
>plot3d("sin(x^2*y)",>anaglyph,angle=30°):
```