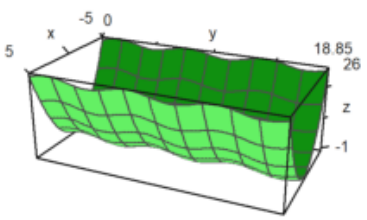
Menggambar Plot 3D dengan EMT

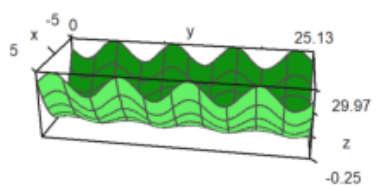
Ini adalah pengenalan plot 3D di Euler. Kita memerlukan plot 3D untuk memvisualisasikan fungsi dua variabel.
Euler menggambar fungsi tersebut menggunakan algoritma pengurutan untuk menyembunyikan bagian di latar belakang. Secara umum Euler menggunakan pusat proyeksi. Standarnya adalah dari kuadran x-y positif menuju titik asal x=y=z=0, tetapi sudut=0Â° dilihat dari ke dalam arah sumbu y. Sudut pandang dan ketinggian dapat diubah.
Euler dapat merencanakan
- permukaan dengan garis penetasan dan level atau rentang level,
- awan titik,
- kurva parametrik,
- permukaan implisit.

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

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

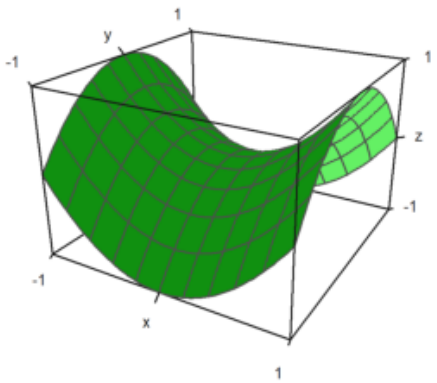



```
>plot3d("x^2+x*sin(y)",-5,5,0*pi,8*pi,angle=100°):
```



Contoh soal:
Sketsakan grafik fungsi f(x,y)=x^2-y^2

```
> plot3d("x^2-y^2", angle=pi/6):
```

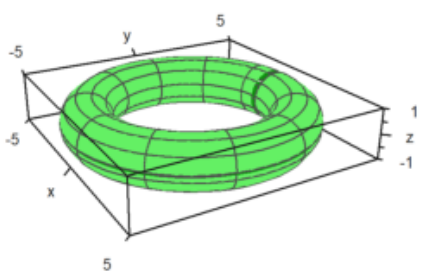


Silakan lakukan modifikasi agar gambar "talang bergelombang" tersebut tidak lurus melainkan melengkung/melingkar, baik melingkar secara mendatar maupun melingkar turun/naik (seperti papan peluncur pada kolam renang. Temukan rumusnya.
Persamaan parameter donat torus adalah

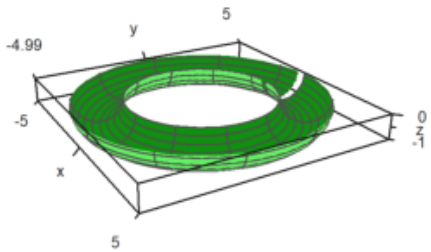
$$r(s,t) = ((b + a \cdot cos(s))sin(t)), (b + a \cdot cos(s))cos(t), a \cdot sin(s))$$

dimana b adalah jari-jari donat, dan a adalah jari-jari lingkaran yang merupakan perpotongan donat dengan bidang y=kx.
Disini saya akan menggunakan a=1 dan b=4

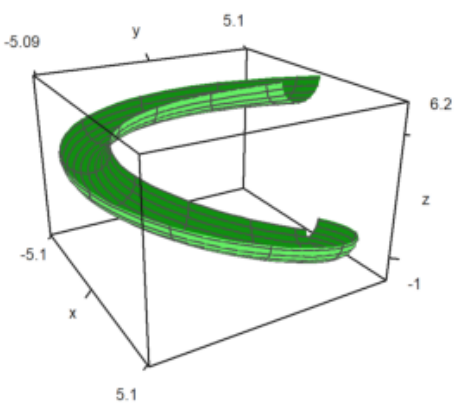
```
> t=0:0.1:2*pi; s=(0:0.1:2*pi)'; ...
>plot3d((4+cos(s))*sin(t),(4+cos(s))*cos(t),sin(s), color=green, angle=60°,grid=10):
```



```
>t=0:0.1:2*pi; s=(pi:0.1:2*pi)'; ...
>plot3d((4+cos(s))*sin(t),(4+cos(s))*cos(t),sin(s), color=green, angle=60°,grid=10):
```




```
> t=0:0.1:2*pi; s=(pi:0.1:2*pi)'; ...
>plot3d((4+cos(s))*sin(t)+cos(t),(4+cos(s))*cos(t)+sin(t),sin(s)+t, color=green, angle=60°
```



Fungsi Dua Variabel

Fungsi Dua Variabel didefinisikan sebagai fungsi bernilai real dari dua variabel real, yakni fungsi f yang memadankan setiap pasangan terurut (x,y) pada suatu himpunan D dari bidang dengan bilangan real tunggal f(x,y)
Berikut diberikan beberapa contoh fungsi dengan dua variabel

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

$$g(x,y) = 2x\sqrt{y}$$

Himpuban D disebut sebagai Daerah Asal fungsi, disebut sebagai daerah asal alami (natural domain) jika tidak dinyatakan secara khsus, yaitu himpunan semua titik (x,y) pada suatu bidang dimana fungsi tersebut bermakna dan menghasilkan nilai bilangan real.
Daerah asal alami fungsi nomor 1 adalah seluruh bidang, sementara daerah asal alami fungsi nomor 2 adalah

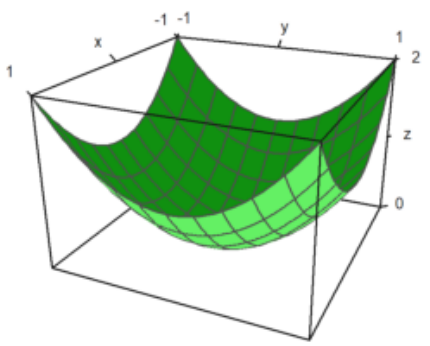
$$\{(x,y) : -\infty < x < \infty, y >= 0\}$$

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

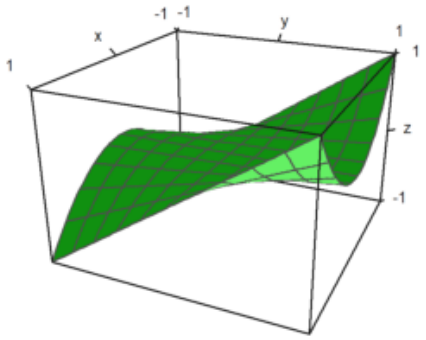
Standarnya adalah kisi-kisi kawat terisi dengan warna berbeda di kedua sisi. Perhatikan bahwa jumlah interval grid default adalah 10, tetapi plot menggunakan jumlah default persegi panjang 40x40 untuk membuat permukaannya. Ini bisa diubah.
- n=40, n=[40,40]: jumlah garis kisi di setiap arah
- grid=10, grid=[10,10]: jumlah garis grid di setiap arah.

Kami menggunakan default n=40 dan grid=10.

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

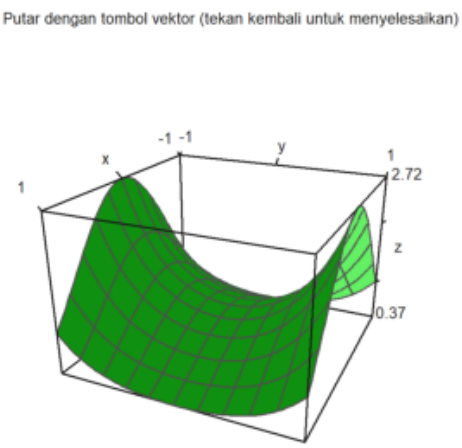



```
>plot3d("x^2*y"):
```



Interaksi pengguna dimungkinkan dengan parameter >pengguna. Pengguna dapat menekan tombol berikut.
- 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: disetel ulang ke default
- kembali: akhiri interaksi

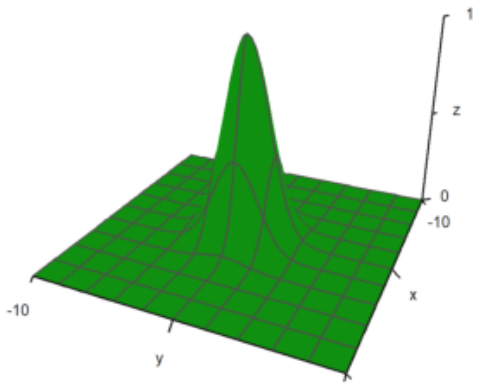
```
>plot3d("exp(-x^2+y^2)",>user, ...
> title="Putar dengan tombol vektor (tekan kembali untuk menyelesaikan)"):
```



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

Ada beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.
fscale: menskalakan ke nilai fungsi (defaultnya adalah <fscale).
skala: angka atau vektor 1x2 untuk diskalakan ke arah x dan y.
bingkai: jenis bingkai (default 1).

```
>plot3d("exp(-(x^2+y^2)/5)",r=10,n=80,fscale=7,scale=1.2,frame=3,>user):
```



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.

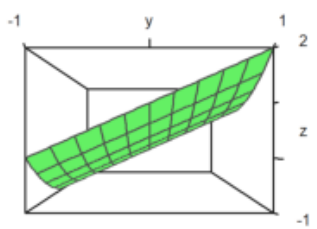
Nilai default dapat diperiksa atau diubah dengan fungsi view(). Ini mengembalikan parameter dalam urutan di atas.

```
> view
```

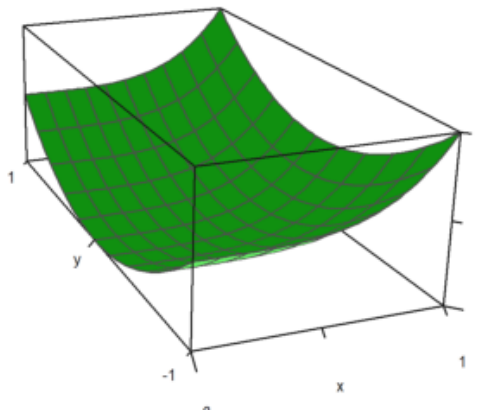
```
  [5,  2.6,  2,  0.4]
```

Jarak yang lebih dekat memerlukan lebih sedikit zoom. Efeknya lebih mirip lebar sudut lensa.
Dalam contoh berikut, sudut=0 dan tinggi=0 dilihat dari negatif sumbu y. Label sumbu untuk y disembunyikan dalam kasus ini.

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



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

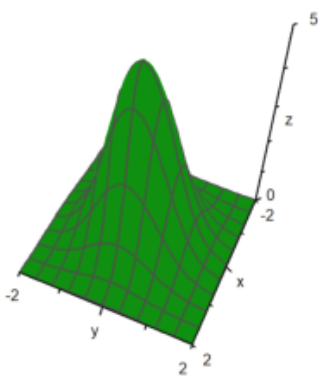


Plot selalu terlihat di tengah kubus plot. Anda dapat memindahkan bagian tengah dengan parameter tengah.

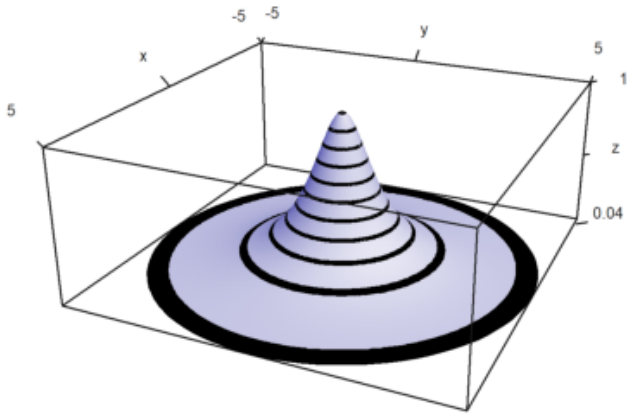
```
>
```

Plot diskalakan agar sesuai dengan unit kubus untuk dilihat. Jadi tidak perlu mengubah jarak atau zoom tergantung ukuran plot. Namun labelnya mengacu pada ukuran sebenarnya.
Jika Anda mematikannya dengan scale=false, Anda perlu berhati-hati agar plot tetap masuk ke dalam jendela plotting, dengan mengubah jarak pandang atau zoom, dan memindahkan bagian tengah.

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

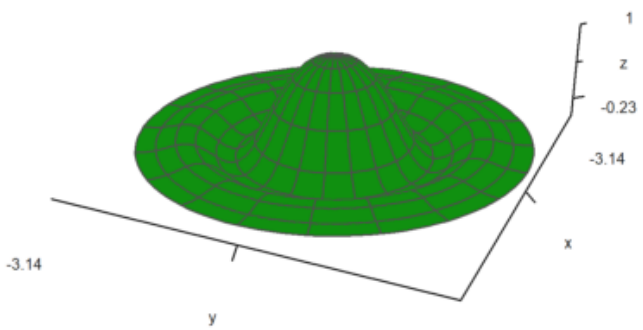


```
>plot3d("1/(x^2+y^2+1)",r=5,>polar, ...
>fscale=2,>hue,n=100,zoom=4,>contour,color=blue):
```



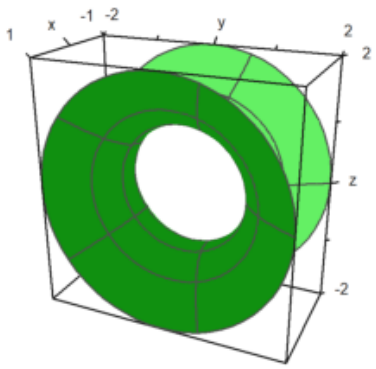
Plot kutub juga tersedia. Parameter polar=true menggambar plot kutub. Fungsi tersebut harus tetap merupakan fungsi dari x dan y. Itu parameter "fscale" menskalakan fungsi dengan skalanya sendiri. Kalau tidak, fungsinya akan diskalakan agar sesuai dengan kubus.

```
>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):
```

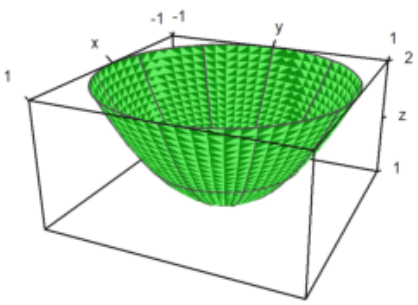


Parameter memutar memutar fungsi dalam x di sekitar sumbu x.
- rotate=1: Menggunakan sumbu x
- rotate=2: Menggunakan sumbu z

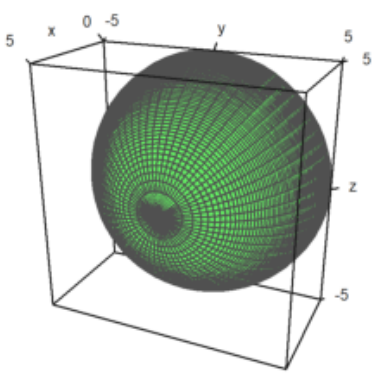
```
>plot3d("x^2+1",a=-1,b=1,rotate=true,grid=5):
```



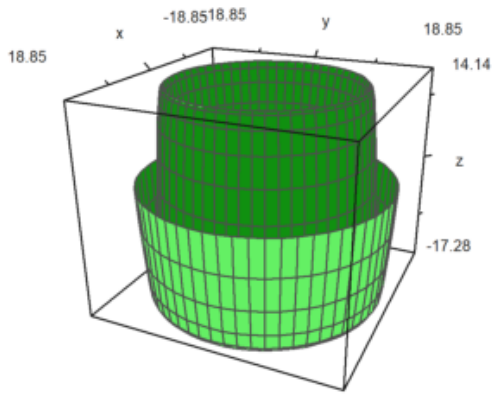

```
>plot3d("x^2+1",a=-1,b=1,rotate=2,grid=5):
```




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

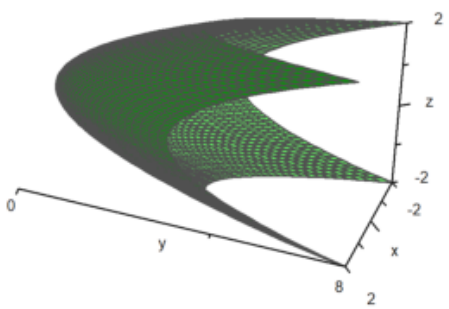


```
>plot3d("x*sin(x)",a=0,b=6pi,rotate=2):
```



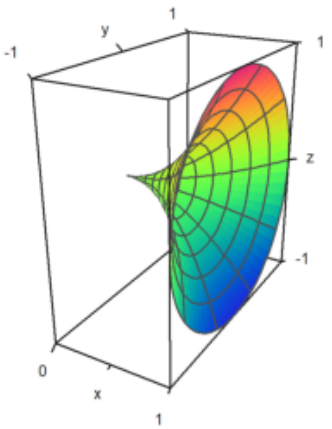
Berikut adalah plot dengan tiga fungsi.

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



Gambarkan grafik permukaan dari kurva y=x^2 untuk interval [0,1] yang diputar terhadap sumbu x!

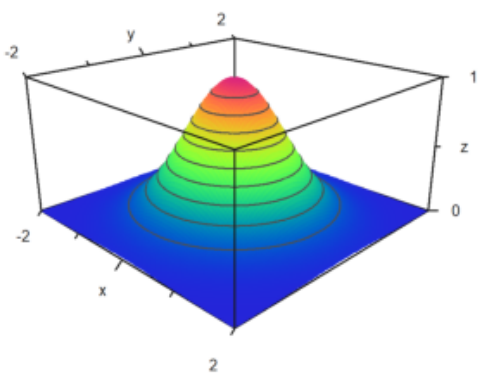
```
>plot3d("x^2", rotate=1, angle=pi/6, a=0, b=1, >spectral, grid=10):
```



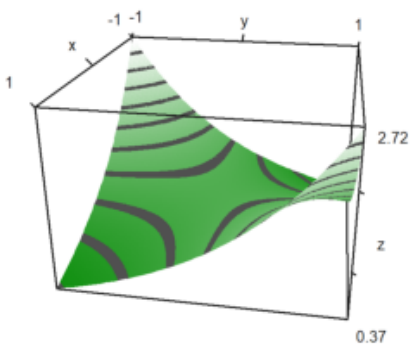
Plot Kontur

Bidang horizontal yang memiliki ketinggan k mempunya persamaan z=k. Apabila bidang ini dipotongkan pada suatu permukaan akan diperoleh kurva yang posisinya horizontal. Kurva ini juga berada pada ketinggian k.
Proyeksi kurva tersebut pada bidang xy disebut kurva ketinggian dengan ketinggian k. Dengan demikian kurva ketinggian berada pada bidang XOY. Selanjutnta bila kita mengubah nilai ketinggian k akan diperoleh banyak kurva ketinggian, kumpulan dari kurva-kurva ketinggin ini disebut peta kontur. Dengan demikian gambar peta kontyr ada pada bidang xy.
Untuk plotnya, Euler menambahkan garis grid. Sebaliknya dimungkinkan untuk menggunakan garis level dan a rona satu warna atau rona warna spektral. Euler dapat menggambar ketinggian fungsi pada a plot dengan bayangan. Di semua plot 3D, Euler dapat menghasilkan anaglyph merah/cyan.
->hue: Mengaktifkan bayangan cahaya, bukan kabel.
- >kontur: Membuat plot garis kontur otomatis pada plot.
- level=... (atau level): Vektor nilai garis kontur.
Defaultnya adalah level="auto", yang menghitung beberapa garis level secara otomatis. Seperti kamu lihat di plot, levelnya sebenarnya adalah rentang level.
Gaya default dapat diubah. Untuk plot kontur berikut, kami menggunakan grid yang lebih halus untuk 100x100 poin, skalakan fungsi dan plot, dan gunakan 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°):
```

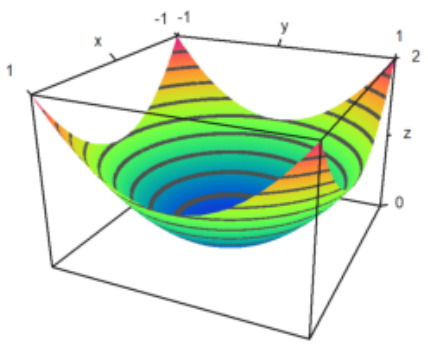


```
>plot3d("exp(x*y)",angle=100°,>contour,color=green):
```



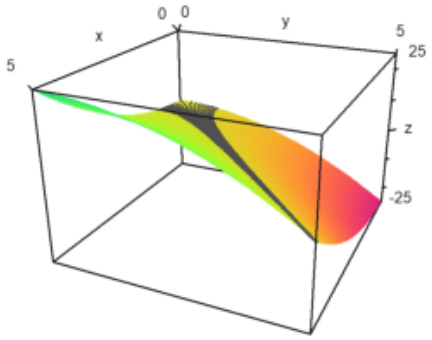
Bayangan default menggunakan warna abu-abu. Namun rentang warna spektral juga tersedia.
- >spektral: Menggunakan skema spektral default
- color=...: Menggunakan warna khusus atau skema spektral
Untuk plot berikut, kami menggunakan skema spektral default dan meningkatkan jumlah poin untuk mendapatkan tampilan yang sangat mulus.

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



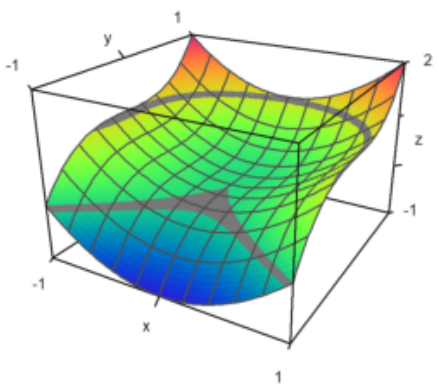
Selain garis level otomatis, kita juga dapat mengatur nilai garis level. Ini akan menghasilkan garis tipis dan rata dari rentang level.

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



Pada plot berikut, kami menggunakan dua pita level yang sangat lebar dari -0,1 hingga 1, dan dari 0,9 hingga 1. Ini dimasukkan sebagai matriks dengan batas level sebagai kolom.
Selain itu, kami melapisi grid dengan 10 interval di setiap arah.

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

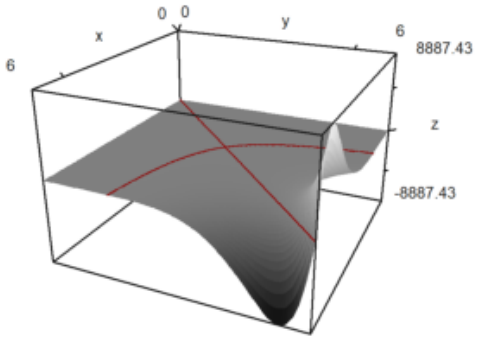


Dalam contoh berikut, kita memplot himpunan, di mana

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

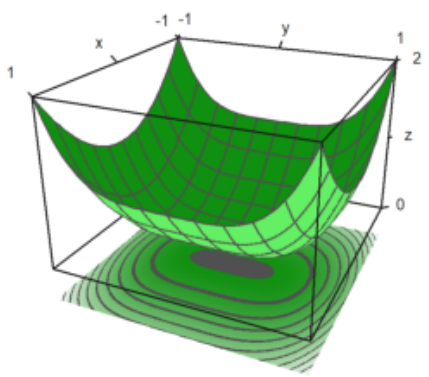
Kami menggunakan satu garis tipis untuk garis level.

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



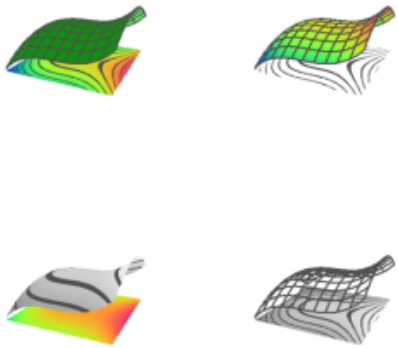
Dimungkinkan untuk menampilkan bidang kontur di bawah plot. Sebuah warna dan jarak ke plot dapat ditentukan.

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



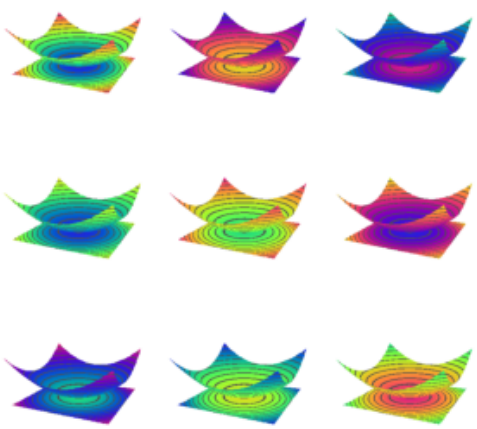
Berikut adalah beberapa gaya lainnya. Kami selalu mematikan bingkai dan menggunakannya berbagai skema warna untuk plot dan grid.

```
>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, yang diberi nomor dari 1 hingga 9. Namun Anda juga dapat menggunakan warna=nilai, di mana nilai
- spektral: untuk rentang dari biru hingga merah
- putih: untuk rentang yang lebih redup
- kuningbiru,unguhijau,birukuning,hijaumerah
- birukuning, hijauungu,kuningbiru,merahhijau

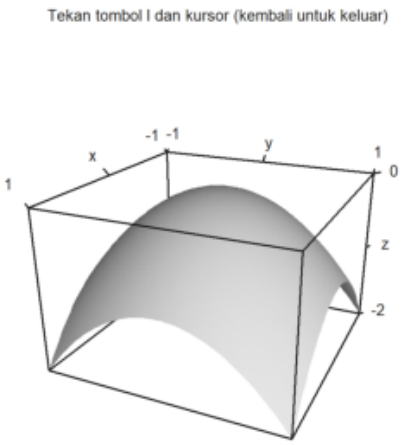
```
>figure(3,3); ...
>for i=1:9; ...
> figure(i); plot3d("x^2+y^2",spectral=i,>contour,>cp,<frame,zoom=4); ...
>end; ...
>figure(0):
```



Sumber cahaya dapat diubah dengan l dan tombol kursor selama interaksi pengguna. Itu juga dapat diatur dengan parameter.
- cahaya: arah cahaya
- dengan: cahaya sekitar antara 0 dan 1

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

```
>plot3d("-x^2-y^2", ...
> hue=true,light=[4,3,1],amb=0,user=true, ...
> title="Tekan tombol l dan kursor (kembali untuk keluar)"):
```

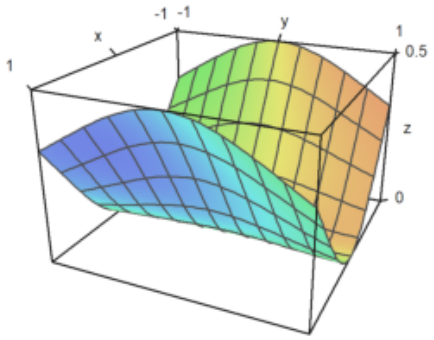


Parameter warna mengubah warna permukaan. Warna garis level juga bisa diubah.

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

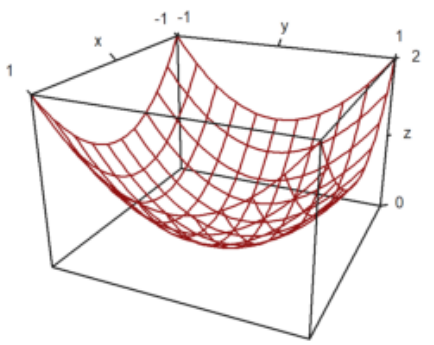
Warna 0 memberikan efek pelangi khusus.

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



Permukaannya juga bisa transparan.

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



Plot Implisit

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

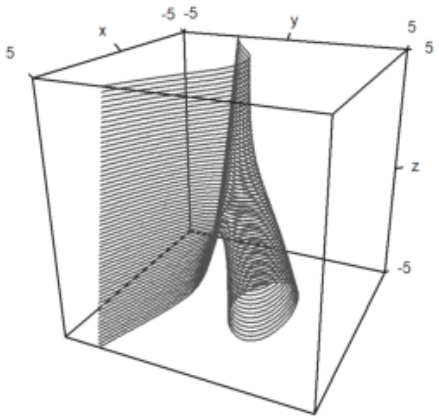
$$f(x,y,z)=0$$

dapat divisualisasikan dalam potongan yang sejajar dengan bidang x-y-, x-z- dan y-z.
- implisit=1: dipotong sejajar dengan bidang y-z
- implisit=2: dipotong sejajar dengan bidang x-z
- implisit=4: dipotong sejajar dengan bidang x-y

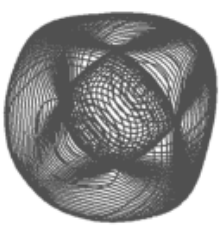
Tambahkan nilai-nilai ini, jika Anda mau. Dalam contoh kita memplot

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

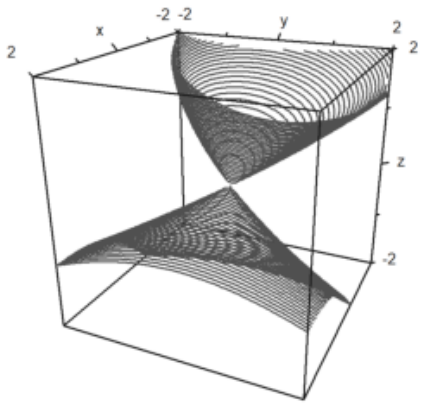
```
>plot3d("x^2+y^3+z*y-1",r=5,implicit=4):
```




```
>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):
```



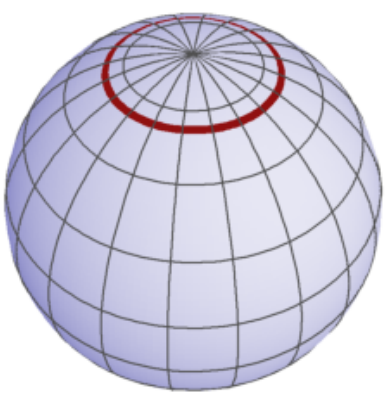
Merencanakan Data 3D

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

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

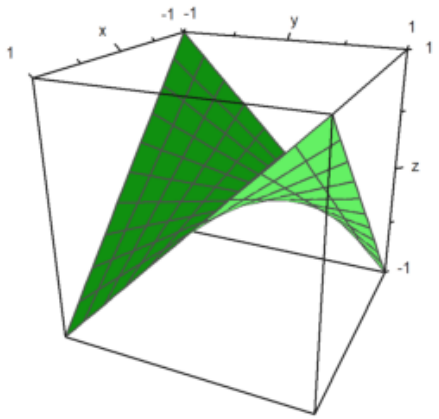
Karena x,y,z adalah matriks, kita asumsikan (t,s) melewati grid persegi. Hasilnya, Anda dapat memplot gambar persegi panjang di ruang hampa.
Anda dapat menggunakan bahasa matriks Euler untuk menghasilkan koordinat secara efektif.
Dalam contoh berikut, kami menggunakan vektor dengan nilai t dan vektor kolom dengan nilai s untuk membuat parameter pada permukaan bola. Dalam gambar kita dapat menandai wilayah, dalam kasus kita wilayah 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 ini contohnya, yaitu grafik suatu fungsi.

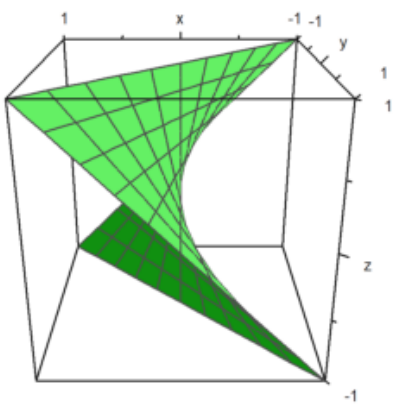
```
>t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,t*s,grid=10):
```



Namun, kita bisa membuat segala jenis permukaan. Ini permukaan yang sama sebagai fungsi

$$x = y\,z$$

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

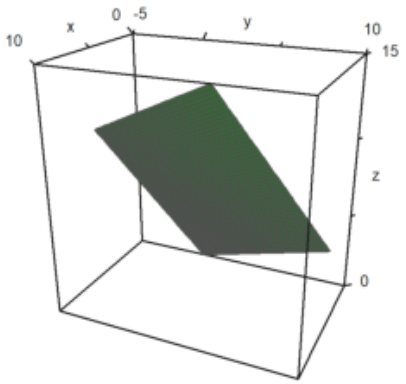


Gambarkan grafik dari fungsi 2 parameter

$$f(s,t) = (x(s,t), y(s,t), z(s,t))$$

dengan x(s,t)=t+s, y(s,t)=2t-s, z(s,t)=t+2s
Dan tentukan bentuk grafik yang terbentuk dari fungsi dua parameter tersebut!

```
>t=0:0.1:5; s=(0:0.1:5)'; ...
>X=t+s; Y=2t-s; Z=t+2s; ...
>plot3d(X,Y,Z):
```



Dari gambar tersebut, terlihat grafiknya berbentuk bidang lurus

Dengan usaha lebih, kami dapat menghasilkan banyak permukaan.
Dalam contoh berikut kita membuat tampilan bayangan dari bola yang terdistorsi. Koordinat bola yang biasa adalah

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

dengan

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

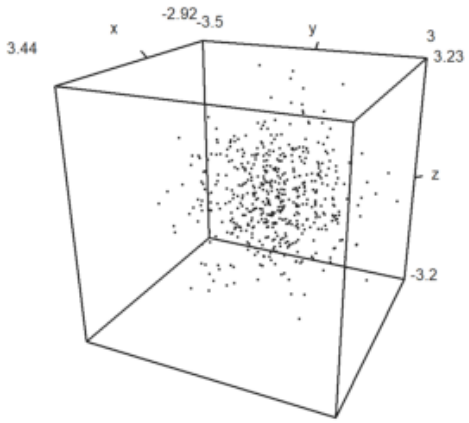
Kami mendistorsi ini dengan sebuah faktor

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

```
>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):
```

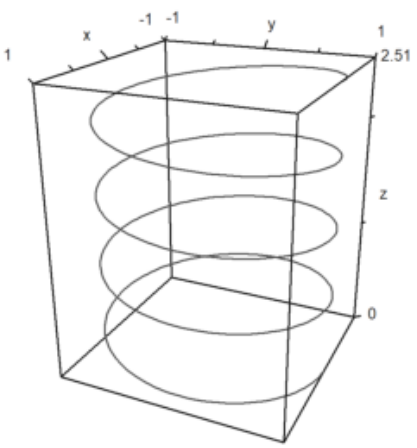
Tentu saja, point cloud juga dimungkinkan. Untuk memplot data titik dalam ruang, kita memerlukan tiga vektor untuk koordinatnya intinya.
Gayanya sama seperti di plot2d dengan points=true;

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

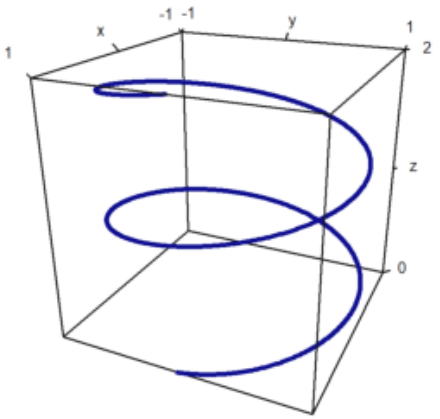


Dimungkinkan juga untuk memplot kurva dalam 3D. Dalam hal ini, lebih mudah untuk menghitung sebelumnya titik-titik kurva. Untuk kurva pada bidang kita menggunakan barisan koordinat dan kawat parameter=benar.

```
>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):
```

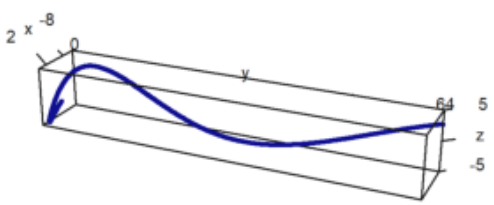


Contoh Soal:
Diketahui posisi seekor lebah pada saat t berada pada koordinat

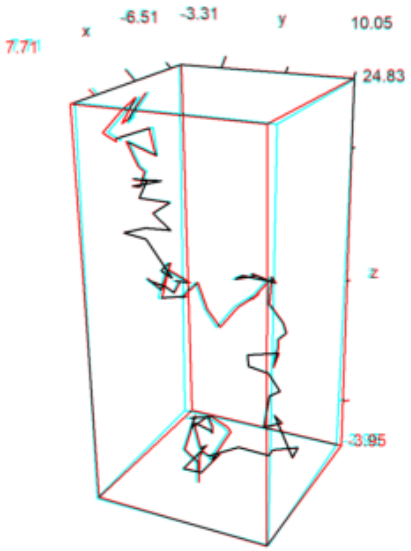
$$(t-8,(t-8)^2,5sin(t-10))$$

Gambarkan lintasan lebah dari posisi awal sampai ke t=10 !

```
> t=linspace(0,10,1000); ...
>aspect(); ...
>plot3d(t-8,(t-8)^2,5*sin(t-10),>wire, linewidth=3, wirecolor=blue):
```

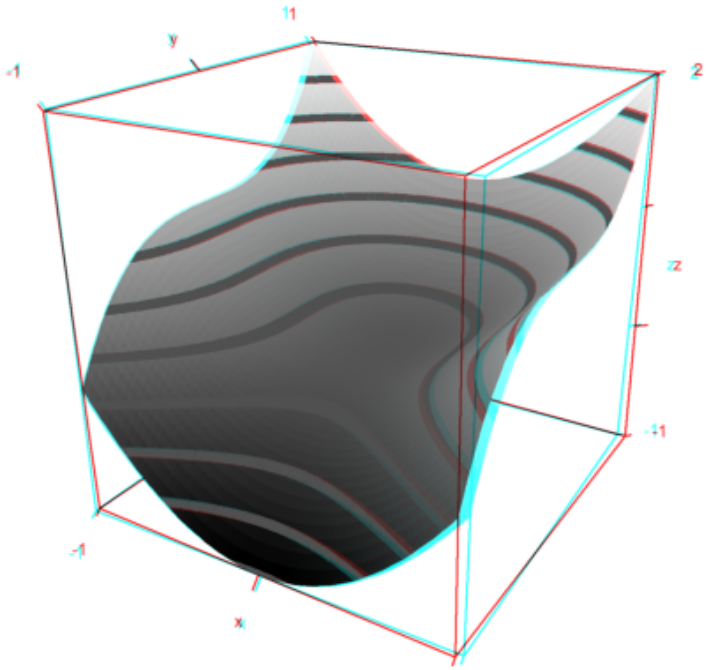


```
>X=cumsum(normal(3,100)); ...
> plot3d(X[1],X[2],X[3],>anaglyph,>wire):
```



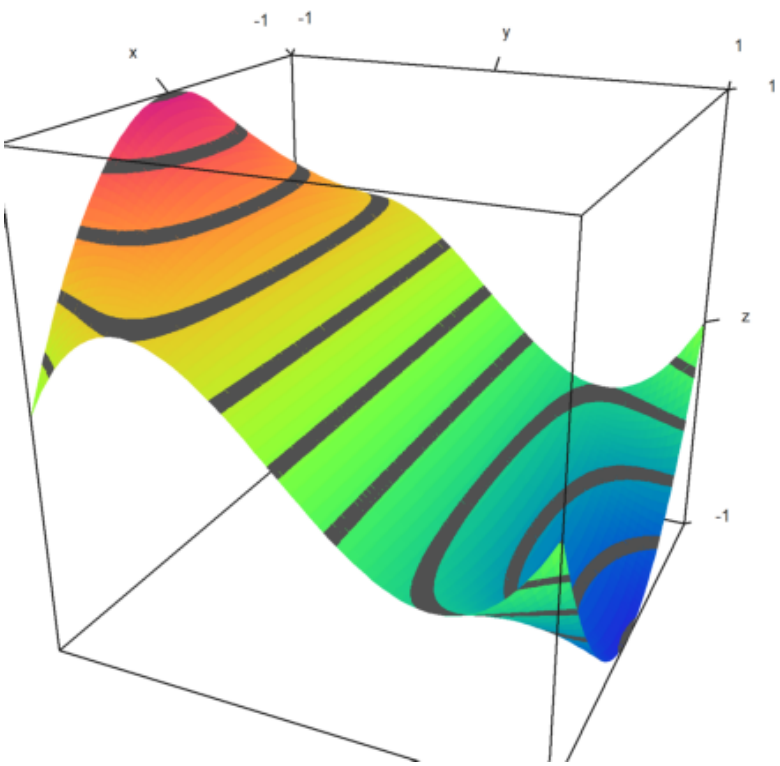
EMT juga dapat memplot dalam mode anaglyph. Untuk melihat plot seperti itu, Anda memerlukan kacamata berwarna merah/sian.

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



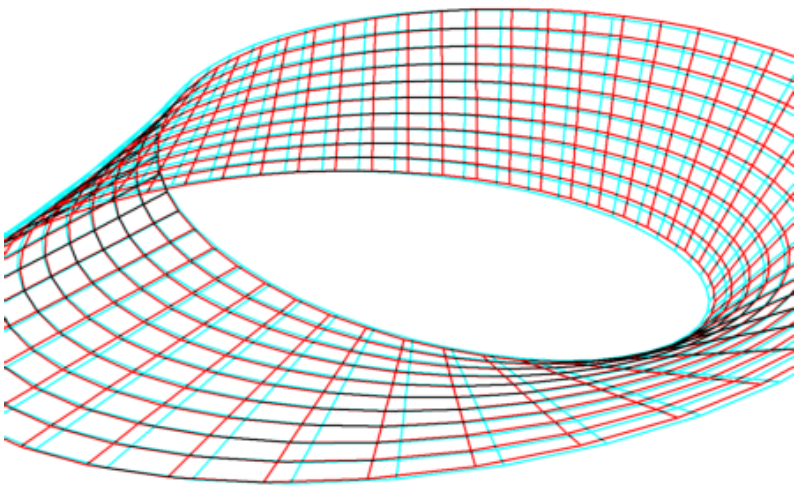
Seringkali, skema warna spektral digunakan untuk plot. Hal ini menekankan ketinggian fungsi.

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



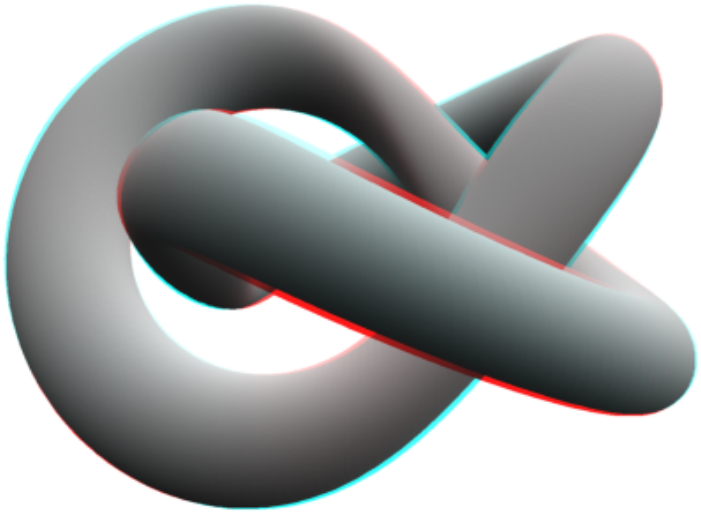
Euler juga dapat memplot permukaan yang diparameterisasi, jika parameternya adalah x-, y-, dan nilai z dari gambar kotak persegi panjang di ruang angkasa.
Untuk demo berikut, kami menyiapkan parameter u- dan v-, dan menghasilkan koordinat ruang dari ini.

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



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

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

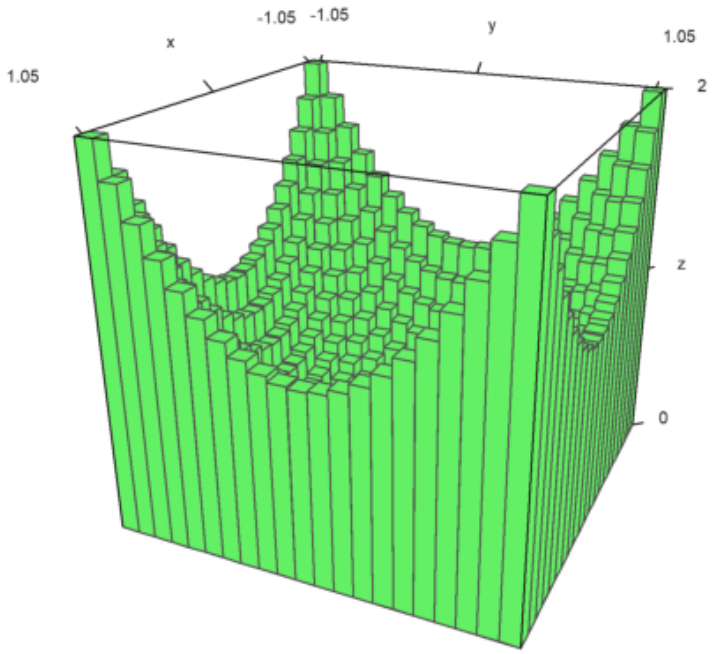


Plot Statistik

Plot batang juga dimungkinkan. Untuk itu, kita harus menyediakannya
- x: vektor baris dengan n+1 elemen
- y: vektor kolom dengan n+1 elemen
- z: matriks nilai nxn.

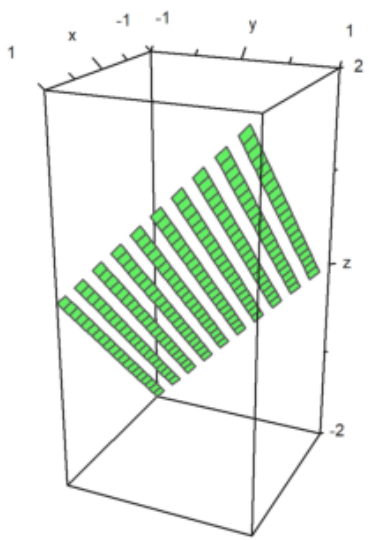
z bisa lebih besar, tetapi hanya nilai nxn yang akan digunakan.
Dalam contoh ini, pertama-tama kita menghitung nilainya. Kemudian kita sesuaikan x dan y, sehingga vektor-vektornya berpusat pada nilai yang digunakan.

```
>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):
```



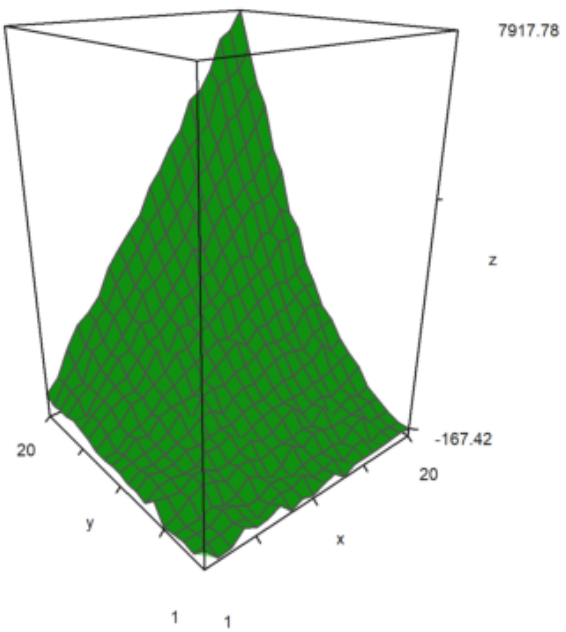
Dimungkinkan untuk membagi plot suatu permukaan menjadi dua bagian atau lebih.

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

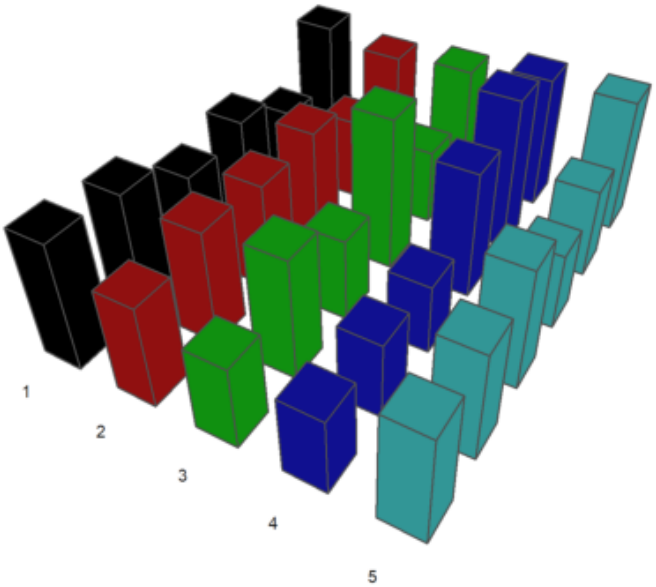


Jika memuat atau menghasilkan matriks data M dari sebuah file dan perlu memplotnya 3D Anda dapat menskalakan matriks ke [-1,1] dengan skala(M), atau skala matriks dengan >zscale. Hal ini dapat dikombinasikan dengan penskalaan individual faktor yang diterapkan tambahan.

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

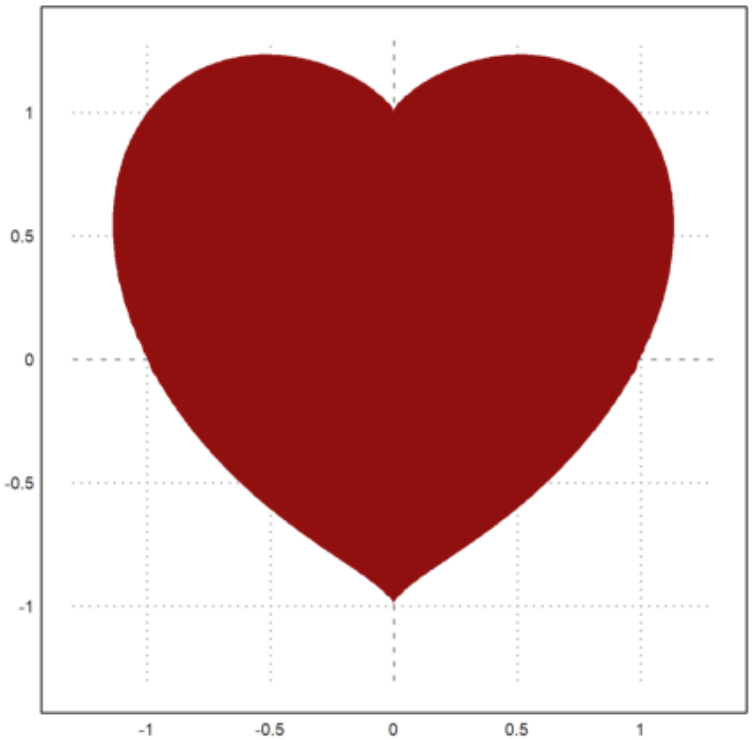



```
>Z=intrandom(5,100,6); v=zeros(5,6); ...
>loop 1 to 5; v[#]=getmultiplicities(1:6,Z[#]); end; ...
>columnsplot3d(v',scols=1:5,ccols=[1:5]):
```



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 hati di sekitar sumbu y. Inilah ungkapannya, yang mana mendefinisikan hati:

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

Selanjutnya kita atur

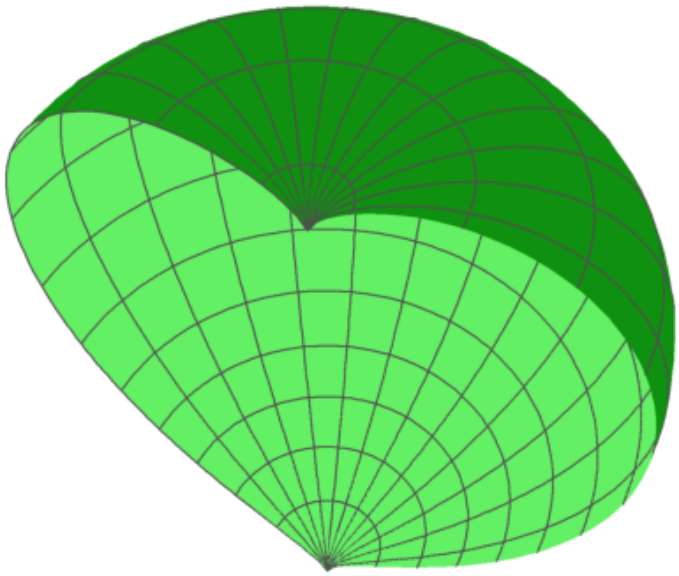
$$x=r.cos(a),\quad y=r.sin(a).$$

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

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

Hal ini memungkinkan untuk mendefinisikan fungsi numerik, yang menyelesaikan 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), ...
>>color,<frame,color=red,zoom=4,amb=0,max=0.7,grid=12,height=50°):
```

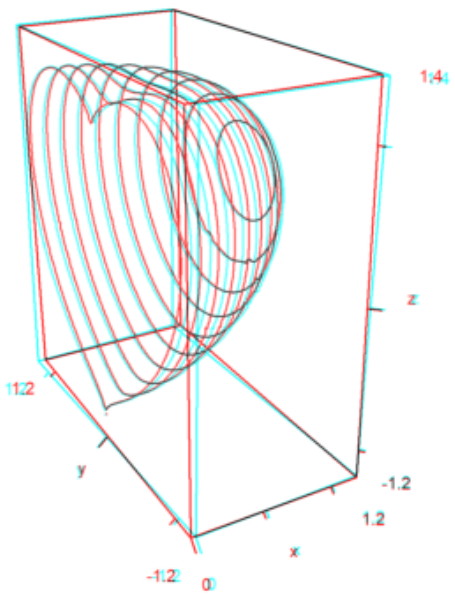


Berikut ini adalah plot 3D dari gambar di atas yang diputar mengelilingi sumbu z. Kami mendefinisikan fungsi, yang mendeskripsikan objek.

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



Plot 3D Khusus

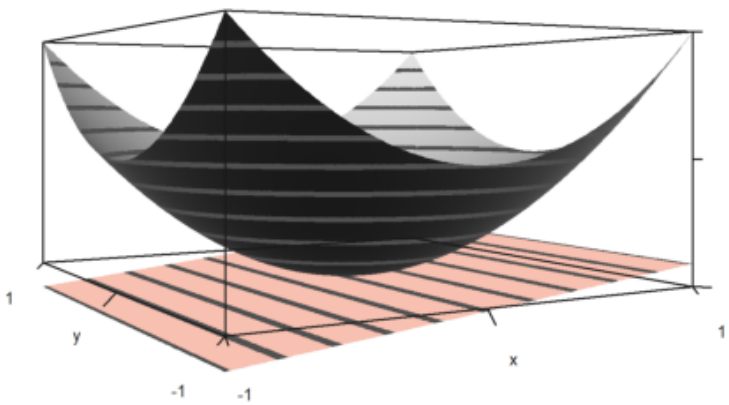
Fungsi plot3d bagus untuk dimiliki, tetapi tidak memenuhi semua kebutuhan. Selain rutinitas yang lebih mendasar, dimungkinkan untuk mendapatkan a plot berbingkai dari objek apa pun yang Anda suka.
Meskipun Euler bukan program 3D, ia dapat menggabungkan beberapa objek dasar. Kami mencoba memvisualisasikan paraboloid dan garis singgungnya.

```
>function myplot...
```

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

Sekarang framedplot() menyediakan frame, dan mengatur tampilan.

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

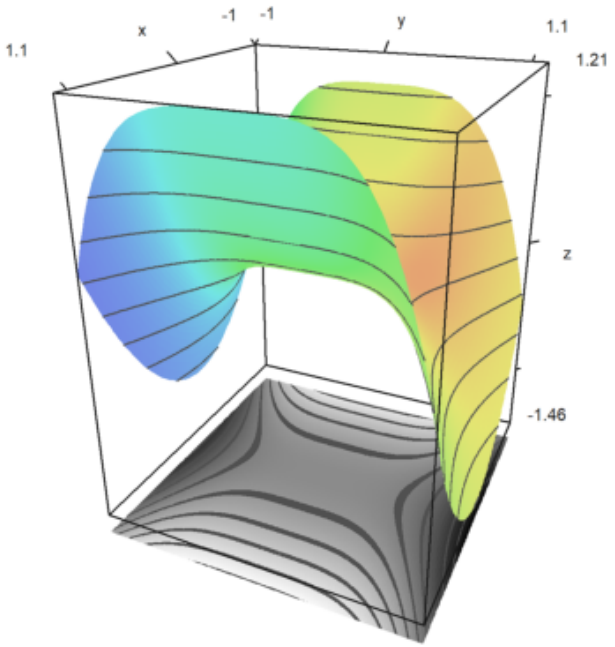


Dengan cara yang sama, Anda dapat memplot bidang kontur secara manual. Perhatikan bahwa plot3d() menyetel jendela ke fullwindow() secara default, namun plotcontourplane() berasumsi demikian.

```
>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
```

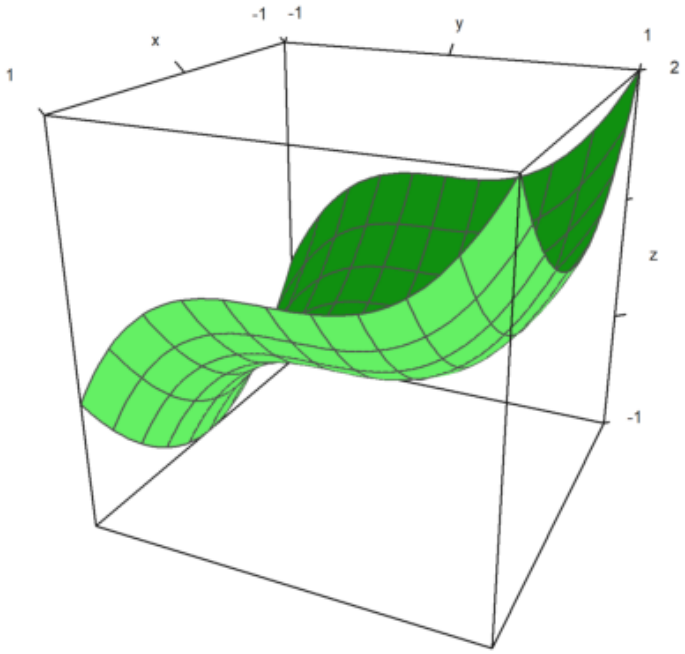
```
>myplot(x,y,z):
```



Animasi

Euler dapat menggunakan frame untuk melakukan pra-komputasi animasi.
Salah satu fungsi yang menggunakan teknik ini adalah memutar. Bisa mengubah sudut pandang dan menggambar ulang plot 3D. Fungsi tersebut memanggil addpage() untuk setiap plot baru. Akhirnya ia menganimasikan plotnya.
Silakan pelajari sumber putaran untuk melihat lebih detail.

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



Menggambar Povray

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

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

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

Fungsi pov3d memiliki semangat yang sama dengan plot3d. Itu 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 buku catatan 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 buku catatan Euler.

Fungsi objek memiliki parameter yang disebut "tampilan", yang memerlukan string dengan kode Povray untuk tekstur dan penyelesaiannya dari objek. Fungsi povlook() dapat digunakan untuk menghasilkan string ini. Ini memiliki parameter untuk warna, itu transparansi, Phong Shading dll.

Perhatikan bahwa alam semesta Povray memiliki sistem koordinat lain. Antarmuka ini menerjemahkan semua koordinat ke Povray sistem. Jadi Anda dapat terus berpikir dalam sistem koordinat Euler dengan z menunjuk vertikal ke atas, dan sumbu x,y,z di akal tangan kanan.

Anda perlu memuat file povray.

```
> load povray;
```

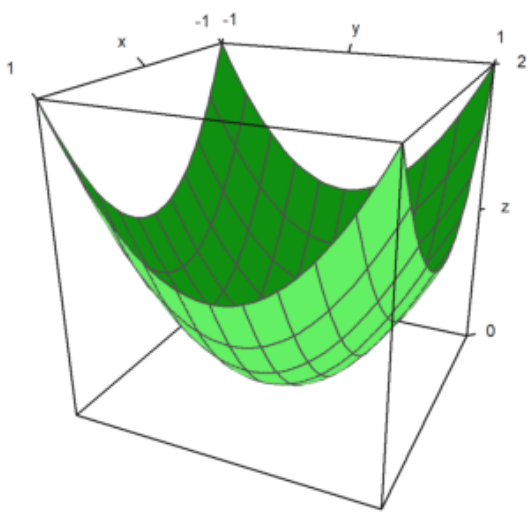
Pastikan, direktori Povray bin ada di jalurnya. Jika tidak, edit variabel berikut agar berisi path ke povray yang dapat dieksekusi.

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

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

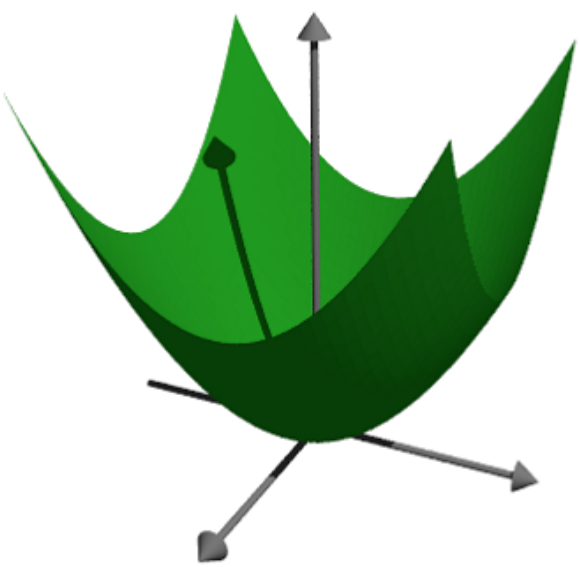
Untuk kesan pertama, kami memplot fungsi sederhana. Perintah berikut menghasilkan file povray di direktori pengguna Anda, dan menjalankan Povray untuk ray tracing file ini.
Jika Anda menjalankan perintah berikut, GUI Povray akan terbuka, menjalankan file, dan menutup secara otomatis. Karena keamanan alasan, Anda akan ditanya, apakah Anda ingin mengizinkan file exe dijalankan. Anda dapat menekan batal untuk menghentikan pertanyaan lebih lanjut. Anda mungkin harus menekan OK di jendela Povray untuk mengonfirmasi dialog pengaktifan Povray.

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



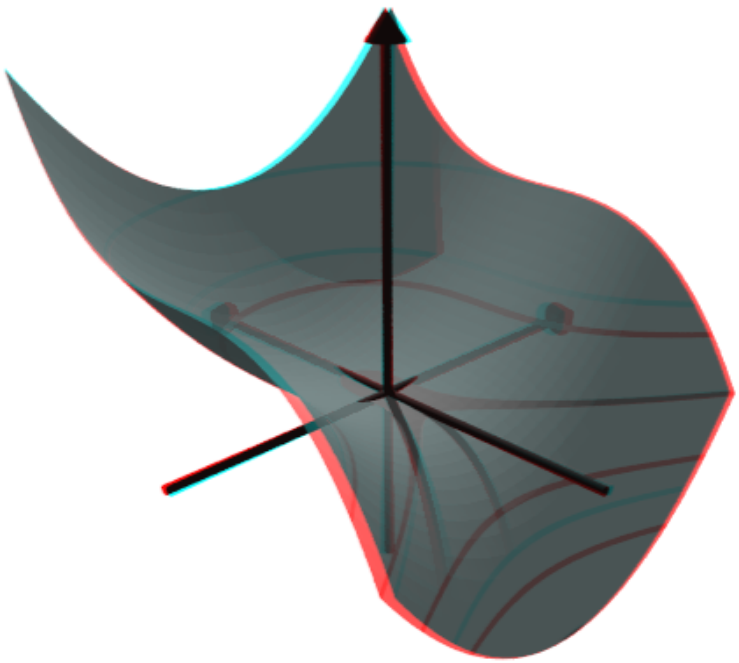

```
>pov3d("sin(x^2+y^2)")
```

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



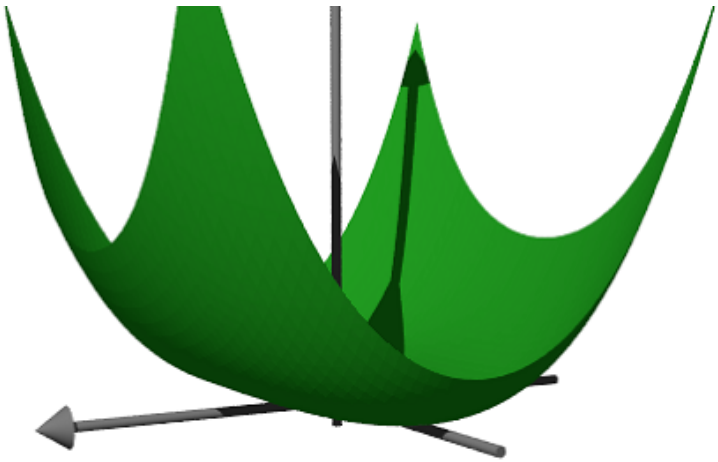
Kita dapat membuat fungsinya transparan dan menambahkan penyelesaian lainnya. Kita dapat juga menambahkan garis level ke plot fungsi.

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



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

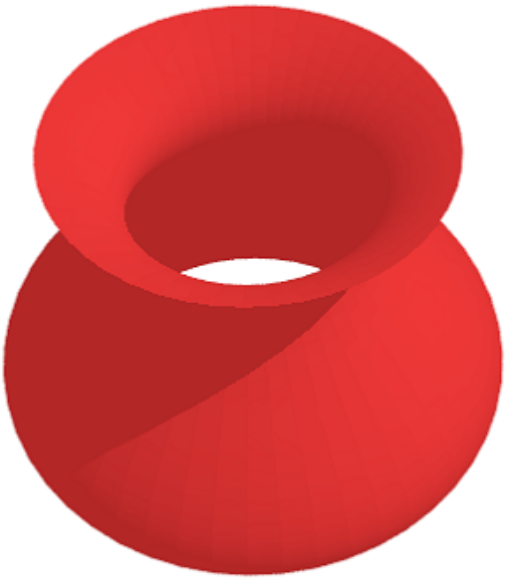
```
>pov3d("((x-1)^2+y^2)*((x+1)^2+y^2)/40",r=2, ...
> angle=-120°,level=1/40,dlevel=0.005,light=[-1,1,1],height=10°,n=50, ...
> <fscale,zoom=3.8);
```



Merencanakan dengan Koordinat

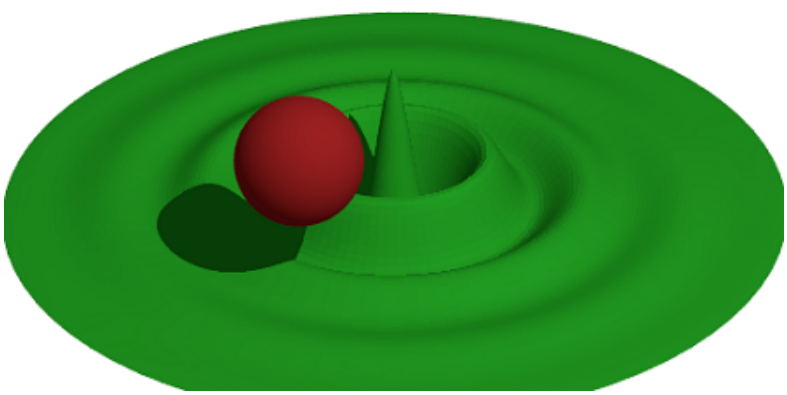
Daripada menggunakan fungsi, kita dapat memplot dengan koordinat. Seperti di plot3d, kita memerlukan tiga matriks untuk mendefinisikan objek.
Dalam contoh ini kita memutar fungsi di sekitar sumbu z.

```
>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]);
```



Dalam contoh berikut, kita 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 berikut ini contoh. 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=6,axis=0,height=30°,add=povsphere([0.5,0,0.25],0.15,povlook(red)), ...
> w=500,h=300);
```



Dengan metode arsiran canggih Povray, hanya sedikit poin yang dapat melakukannya menghasilkan permukaan yang sangat halus. Hanya di perbatasan dan dalam bayangan triknya mungkin menjadi jelas.
Untuk ini, kita perlu menjumlahkan vektor normal di setiap titik matriks.

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

$$x^2 y^3$$

Persamaan permukaannya adalah [x,y,Z]. Kami menghitung dua turunannya ke x dan y dari ini dan ambil perkalian silang sebagai normal.

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

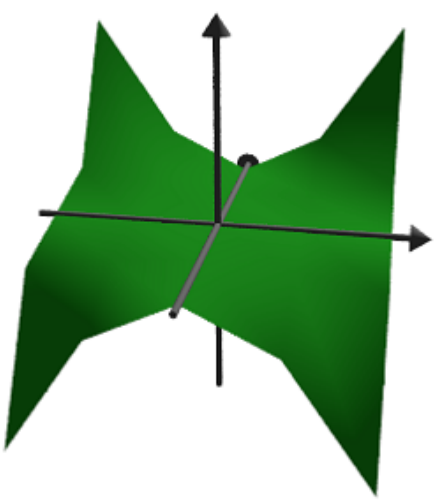
Kami mendefinisikan normal sebagai produk silang dari turunan ini, dan mendefinisikan fungsi koordinat.

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

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

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. Di sana adalah versi perbaikan dari contoh ini.
Lihat: Contoh\Trefoil Knot | Simpul Trefoil
Untuk tampilan yang bagus dengan poin yang tidak terlalu banyak, kami menambahkan vektor normal di sini. Kami menggunakan Maxima untuk menghitung normalnya bagi kami. Pertama, ketiganya berfungsi 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);
```

Maka dua turunan vektor ke x dan y.

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

Sekarang normal, yang merupakan perkalian silang kedua turunannya.

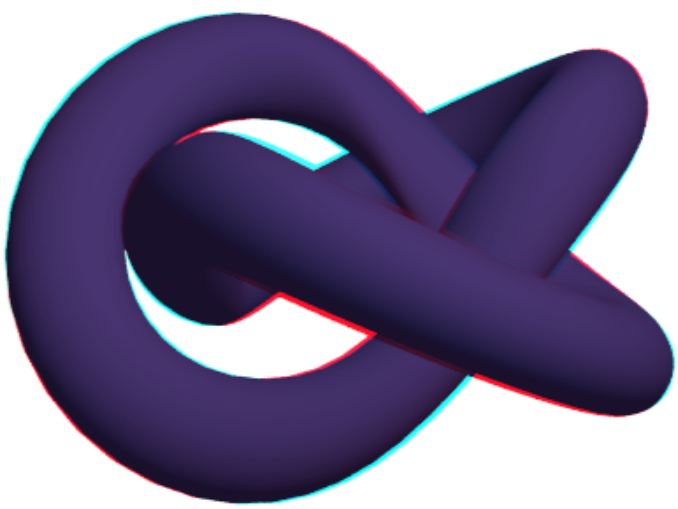
```
>dn &= crossproduct(dx,dy);
```

Kami sekarang mengevaluasi semua ini secara numerik.

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

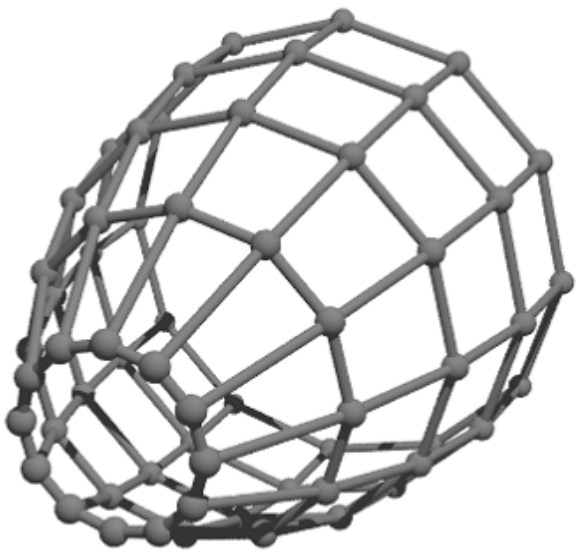
Vektor normal adalah evaluasi ekspresi simbolik dn[i] untuk saya=1,2,3. Sintaksnya adalah &"ekspresi"(parameter). Ini alternatif dari metode pada contoh sebelumnya, di mana kita mendefinisikannya ekspresi simbolik NX, NY, NZ terlebih dahulu.

```
>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));
```



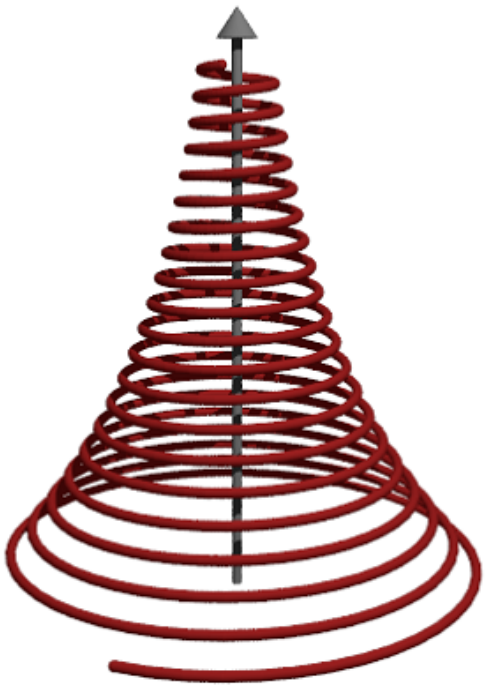
Kita juga dapat membuat grid dalam 3D.

```
>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 dapat dilakukan.

```
>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. Benda-benda ini adalah disimpan sebagai string di Euler, dan perlu ditulis ke file Povray.
Kita memulai keluaran 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.

```
>rc1=povcylinder(-povx,povx,1,povlook(red)); ...
>c2=povcylinder(-povy,povy,1,povlook(yellow)); ...
>c3=povcylinder(-povz,povz,1,povlook(blue)); ...
```

Stringnya berisi kode Povray, yang tidak perlu kita pahami saat itu poin.

```
>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 Anda lihat, kami menambahkan tekstur pada objek dalam tiga warna berbeda.
Hal ini dilakukan oleh povlook(), yang mengembalikan string dengan relevan Kode Povray. Kita dapat menggunakan warna default Euler, atau menentukan warna kita sendiri warna. Kita juga dapat menambahkan transparansi, atau mengubah cahaya sekitar.

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

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

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

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

```
  Variable c1 not found!
  Error in:
  writeln(povintersection([c1,c2,c3])); ...
                                ^
```

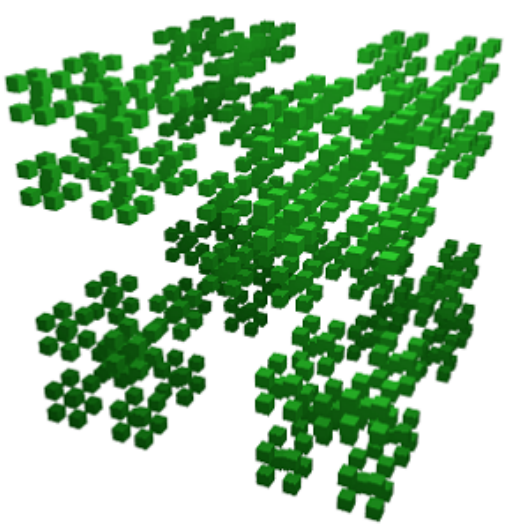
Persimpangan tiga silinder sulit untuk divisualisasikan, jika Anda tidak pernah melakukannya pernah melihatnya sebelumnya.

```
>povend;
```

Fungsi berikut menghasilkan fraktal secara rekursif.
Fungsi pertama menunjukkan, bagaimana Euler menangani objek Povray sederhana. Itu 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 pemotongan satu objek dari objek lainnya. Menyukai persimpangan, ada bagian dari objek CSG Povray.

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

Untuk demonstrasi ini, kami mendefinisikan objek di Povray, bukan menggunakan string di Euler. Definisi ditulis ke file langsung.
Koordinat kotak -1 berarti [-1,-1,-1].

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

Kita bisa menggunakan objek ini di povobject(), yang mengembalikan string sebagai biasa.

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

Kita membuat kubus kedua, lalu memutar dan menskalakannya sedikit.

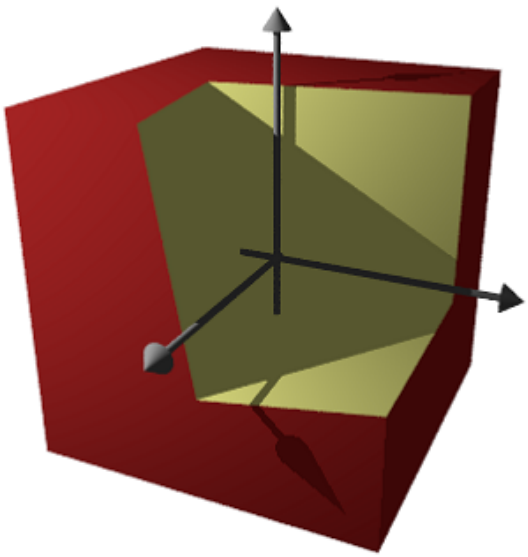
```
>c2=povobject("mycube",povlook(yellow),translate=[1,1,1], ...
> rotate=xrotate(10°)+yrotate(10°), scale=1.2);
```

Lalu kita ambil selisih kedua benda tersebut.

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

Sekarang tambahkan tiga sumbu.

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



Fungsi Implisit

Povray dapat memplot himpunan di mana f(x,y,z)=0, seperti parameter implisit di plot3d. Hasilnya terlihat jauh lebih baik, Namun.
Sintaks untuk fungsinya sedikit berbeda. Anda tidak dapat menggunakan keluaran ekspresi Maxima atau Euler.

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

```
>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)+
>povend();
```

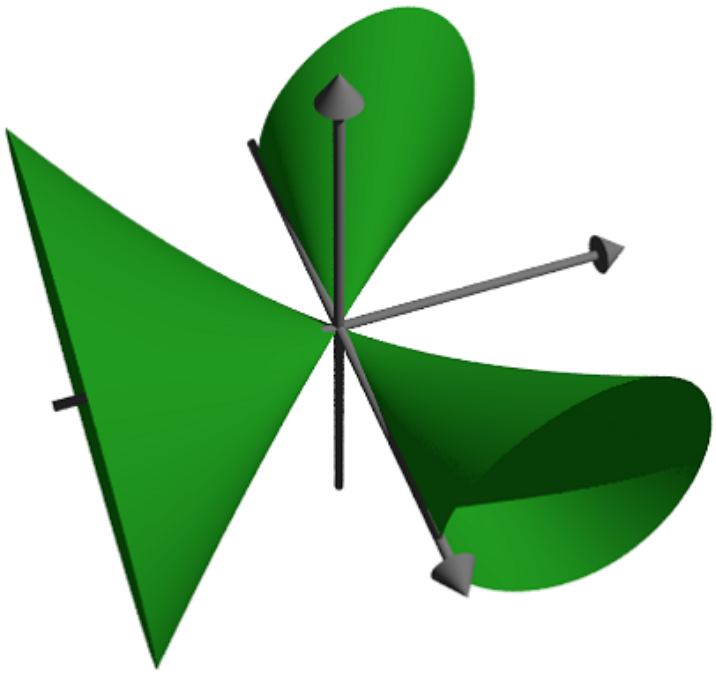
```
  Error : Povray error!

  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);
```

Buat permukaan implisit. Perhatikan sintaks yang berbeda di ekspresi.

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



Objek Jaring

Dalam contoh ini, kami menunjukkan cara membuat objek mesh, dan menggambarnya dengan informasi tambahan.
Kita ingin memaksimalkan xy pada kondisi x+y=1 dan mendemonstrasikan sentuhan tangensial pada garis datar.

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

Kita tidak dapat menyimpan objek dalam string seperti sebelumnya, karena terlalu besar. Jadi kita mendefinisikan objek dalam file Povray menggunakan menyatakan. Fungsi povtriangle() melakukan ini secara otomatis. Ia dapat menerima vektor normal seperti pov3d().
Berikut ini mendefinisikan objek mesh, dan segera 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 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);
```

Tulis permukaannya dikurangi kedua cakram.

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

Tuliskan kedua perpotongannya.

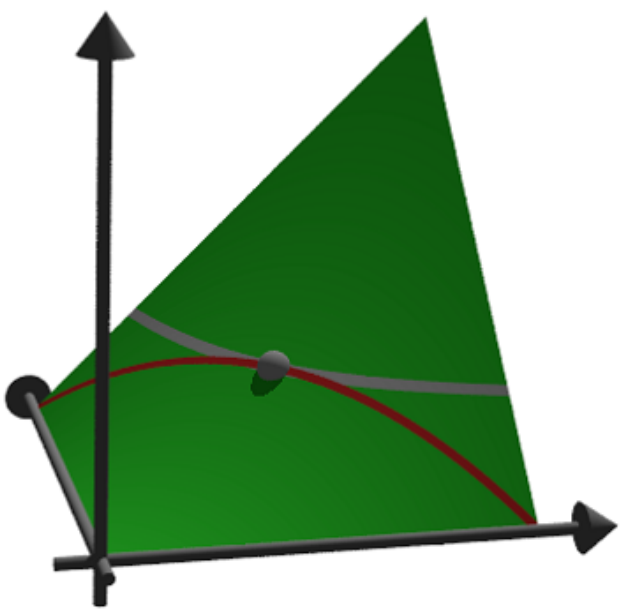
```
>writeln(povintersection([mesh,cl],povlook(red)); ...
>writeln(povintersection([mesh,ll],povlook(gray)));
```

Tulis poin maksimal.

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

Tambahkan sumbu dan selesaikan.

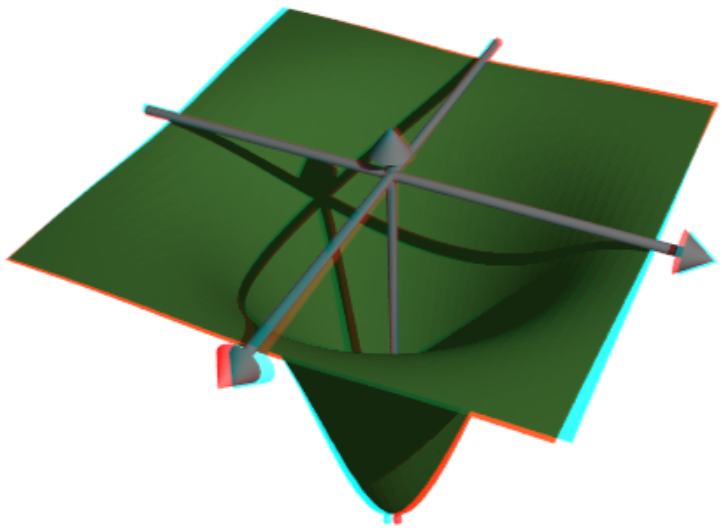
```
>writeAxes(0,1,0,1,0,1,d=0.015); ...
>povend();
```



Anaglyph di Povray

Untuk menghasilkan anaglyph untuk kacamata merah/cyan, Povray harus dijalankan dua kali dari posisi kamera yang berbeda. Ini menghasilkan dua file Povray dan dua file PNG, yang dimuat dengan fungsi loadanaglyph().
Tentu saja, Anda memerlukan kacamata merah/sian untuk melihat contoh berikut dengan baik.
Fungsi pov3d() memiliki saklar sederhana untuk menghasilkan anaglyph.

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



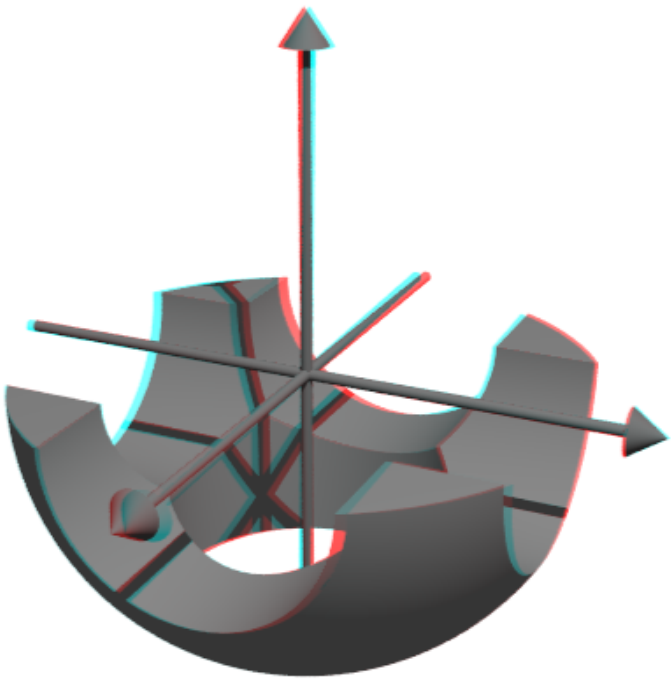
Jika Anda membuat adegan dengan objek, Anda perlu memasukkan generasinya adegan menjadi suatu fungsi, dan jalankan dua kali dengan nilai yang berbeda parameter anaglif.

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

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

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

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



Mendefinisikan Objek sendiri

Antarmuka povray Euler berisi banyak objek. Tapi kamu memang begitu tidak terbatas pada ini. Anda dapat membuat objek sendiri, yang digabungkan objek lain, atau merupakan objek yang benar-benar baru.
Kami mendemonstrasikan torus. Perintah Povray untuk ini adalah "torus". Jadi kita mengembalikan string dengan perintah ini dan parameternya. Perhatikan bahwa torus selalu berpusat pada titik asal.

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

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

Ini torus pertama kita.

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

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

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

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

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

Sekarang kita menempatkan objek-objek ini ke dalam sebuah adegan. Untuk tampilannya kami menggunakan Phong Bayangan.

```
>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, hal ini tidak terjadi menampilkan kesalahan. Oleh karena itu Anda harus menggunakan
>povend(<keluar);
jika ada yang tidak berhasil. Ini akan membiarkan jendela Povray terbuka.

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

Berikut adalah contoh yang lebih rumit. Kami memecahkannya
lateks: Ax \le b, \quad x \ge 0, \quad c.x \to \text{Maks.}
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 mempunyai solusi.

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

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

Ya, sudah.
Selanjutnya kita mendefinisikan dua objek. Yang pertama adalah pesawat
lateks: a \cdot x \le b

```
>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
```

Lalu kita tentukan perpotongan semua setengah ruang dan kubus.

```
>
```

```
  ol=[];
   loop 1 ke baris(A); ol=ol|satu pesawat(A[#],b[#]); akhir;
  ol=ol|povbox([0,0,0],[r,r,r]);
   kembalikan titik temu(ol,lihat);
  fungsi akhir
```

Sekarang kita dapat merencanakan adegannya.

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

Berikut ini adalah lingkaran di sekitar optimal.

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

```
  Function oneplane not found.
  Try list ... to find functions!
  Try "trace errors" to inspect local variables after errors.
  adm:
      loop 1 to rows(A); ol=ol|oneplane(A[#],b[#]); end;
```

Dan kesalahan ke arah optimal.

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

Kami menambahkan teks ke layar. Teks hanyalah objek 3D. Kita perlu menempatkannya dan putar sesuai pandangan kita.

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

Contoh Lainnya

Anda dapat menemukan beberapa contoh Povray di Euler berikut ini file.
Lihat: Contoh/Bola Dandelin
Lihat: Contoh/Donat Matematika
Lihat: Contoh/Simpul Trefoil
Lihat: Contoh/Optimasi dengan Affine Scaling