

EMT4Plot3D_Huwaida Nisa Awanis_23030130039

Nama : Huwaida Nisa Awanis
Prodi: Pendidikan Matematika C 2023
NIM : 23030130039

Menggambar Plot 3D dengan EMT

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

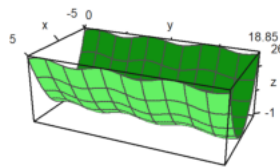
Euler menggambar fungsi tersebut menggunakan algoritma pengurutan untuk menyembunyikan bagian di latar belakang. Secara umum Euler menggunakan proyeksi sentral. Standarnya adalah dari kuadran x-y positif menuju titik asal $x=y=z=0$, tetapi sudut= 0° dilihat dari arah sumbu y. Sudut pandang dan ketinggian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

Euler bisa merencanakan

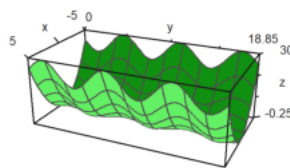
- permukaan dengan garis penetasan dan level atau rentang level,
- awan titik,
- kurva parametrik,
- permukaan implisit.

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

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



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



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

Fungsi dari dua Variabel

Untuk grafik sebuah fungsi, gunakan

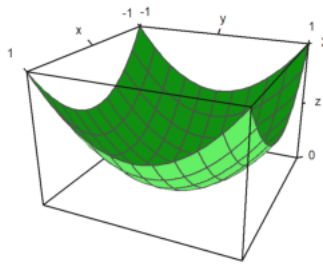
- ekspresi sederhana dalam x dan y,
- nama fungsi dari dua variabel
- atau matriks data.

Standarnya adalah kisi-kisi kawat yang terisi dengan warna yang berbeda di kedua sisi. 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.

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

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

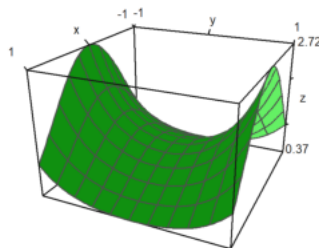


Interaksi pengguna dapat dilakukan dengan >user parameter. Pengguna dapat menekan tombol-tombol berikut:

- kiri, kanan, atas, bawah: memutar sudut pandang
- +, -: memperbesar atau memperkecil
- a: menghasilkan anaglyph (lihat di bawah)
- 1: beralih memutar sumber cahaya (lihat di bawah)
- spasi: mengembalikan ke pengaturan awal
- enter: mengakhiri interaksi

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

Turn with the vector keys (press return to finish)



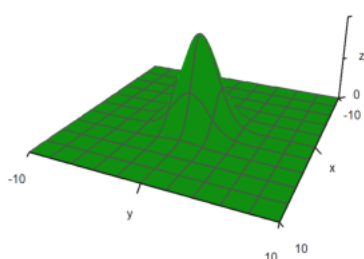
Rentang plot untuk fungsi dapat ditentukan dengan:

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

ada beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.

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

- jarak: 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.

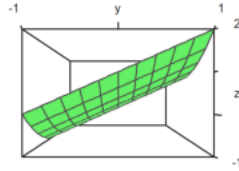
```
>view
```

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

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

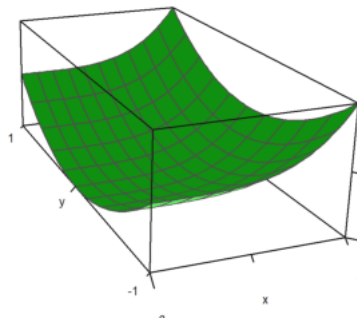
Dalam contoh berikut ini, sudut = 0 dan tinggi = 0 terlihat dari sumbu y negatif. Label sumbu untuk y disembunyikan dalam kasus ini.

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



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

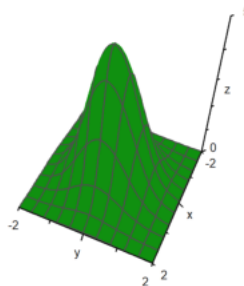
```
>plot3d("x^4+y^2", a=0, b=1, c=-1, d=1, angle=-20°, height=20°, ...  
center=[0.4, 0, 0], zoom=5) :
```



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

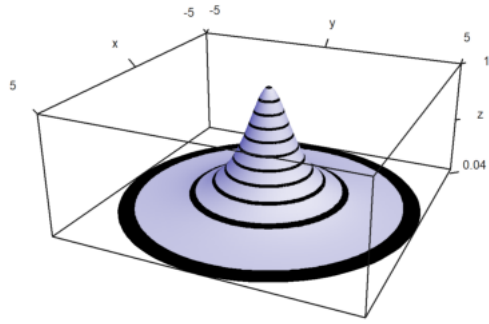
Jika Anda menonaktifkannya dengan scale=false, Anda harus berhati-hati agar plot tetap muat di dalam jendela plotting, dengan mengubah jarak tampilan atau zoom, dan memindahkan bagian tengahnya.

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

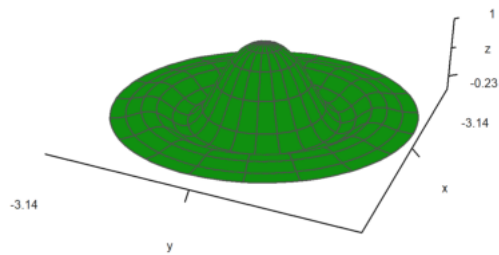


Plot polar juga tersedia. Parameter polar=true menggambar plot polar. Fungsi harus tetap merupakan fungsi dari x dan y. Parameter "fscale" menskalakan fungsi dengan skala sendiri. Jika tidak, fungsi akan diskalakan agar sesuai dengan kubus.

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



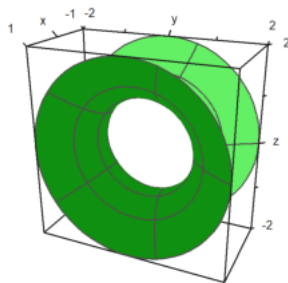
```
>function f(r) := exp(-r/2)*cos(r); ...
plot3d("f(x^2+y^2)",>polar,scale=[1,1,0.4],r=pi,frame=3,zoom=4):
```



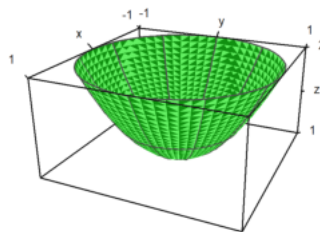
Parameter rotate memutar fungsi dalam x di sekitar sumbu x.

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

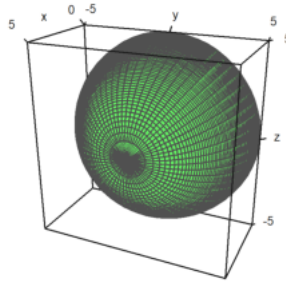
```
>plot3d("x^2+1",a=-1,b=1,rotate=true,grid=5):
```



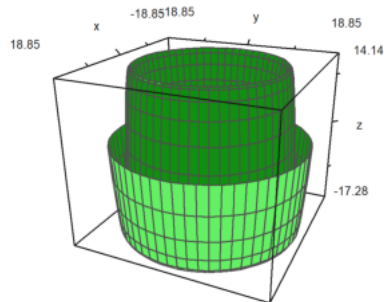
```
>plot3d("x^2+1",a=-1,b=1,rotate=2,grid=5):
```



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

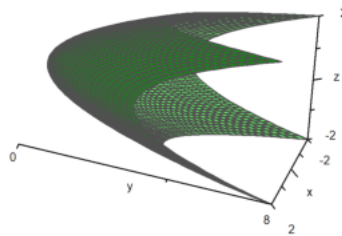


```
>plot3d("x*sin(x)", a=0, b=6pi, rotate=2) :
```



Berikut ini adalah plot dengan tiga fungsi.

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



Plot Kontur

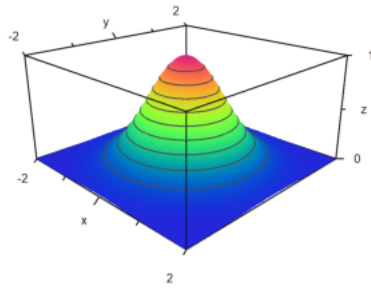
Untuk plot, Euler menambahkan garis kisi-kisi. Sebagai gantinya, dimungkinkan untuk menggunakan garis level dan rona satu warna atau rona berwarna spektral. Euler dapat menggambar ketinggian fungsi pada plot dengan bayangan. Pada semua plot 3D, Euler dapat menghasilkan anaglyph merah/cyan.

- Rona: Mengaktifkan bayangan cahaya, bukan kabel.
- >contour: Memplot garis kontur otomatis pada plot.
- level=... (atau level): Vektor nilai untuk garis kontur.

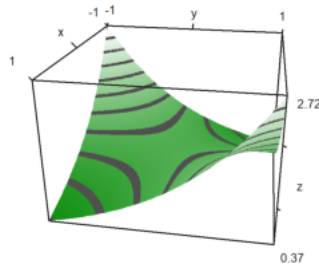
Defaultnya adalah level="auto", yang menghitung beberapa garis level secara otomatis. Seperti yang Anda lihat di plot, level sebenarnya adalah rentang level.

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

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



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

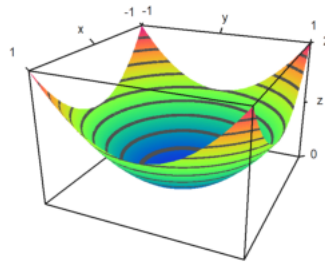


Bayangan default menggunakan warna abu-abu. Tetapi, kisaran warna spektral juga tersedia.

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

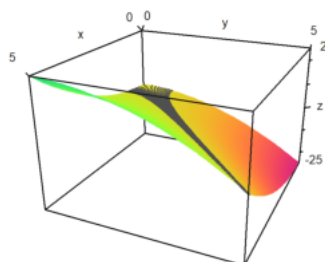
Untuk plot berikut ini, kami menggunakan skema spektral default dan menambah jumlah titik untuk mendapatkan tampilan yang sangat mulus.

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



Alih-alih garis level otomatis, kita juga dapat menetapkan nilai garis level. Hal ini akan menghasilkan garis level yang tipis, alih-alih rentang level.

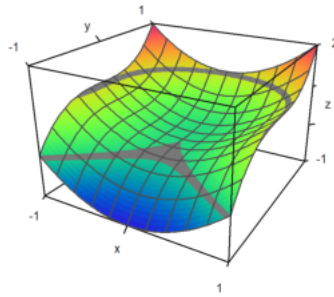
```
>plot3d("x^2-y^2", 0, 5, 0, 5, level=-1:0.1:1, color=redgreen):
```



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

Selain itu, kami menghamparkan 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):
```

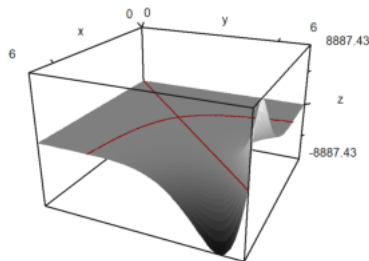


Pada contoh berikut, kami memplot himpunan, di mana

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

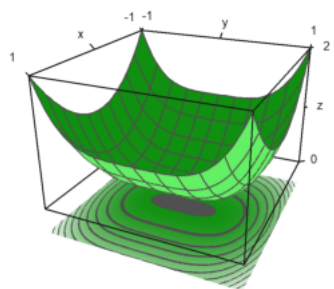
Kita menggunakan satu garis tipis untuk garis level.

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



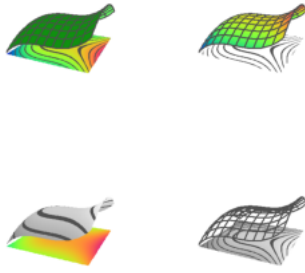
Dimungkinkan untuk menampilkan bidang kontur di bawah plot. Warna dan jarak ke plot dapat ditentukan.

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



Berikut ini beberapa gaya lainnya. Kami selalu mematikan bingkai, dan menggunakan berbagai skema warna untuk plot dan kisi-kisi.

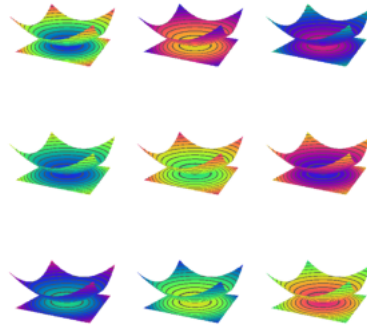
```
>figure(2,2); ...
expr="y^3-x^2"; ...
figure(1); ...
plot3d(expr,<frame,>cp,cpcolor=spectral); ...
figure(2); ...
plot3d(expr,<frame,>spectral,grid=10,cp=2); ...
figure(3); ...
plot3d(expr,<frame,>contour,color=gray,nc=5,cp=3,cpcolor=greenred); ...
figure(4); ...
plot3d(expr,<frame,>hue,grid=10,>transparent,>cp,cpcolor=gray);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



Ada beberapa skema spektral lainnya, yang diberi nomor dari 1 hingga 9. Tetapi Anda juga dapat menggunakan `color=value`, di mana `value`

- spektral: untuk rentang dari biru ke merah
- putih: untuk rentang yang lebih redup
- kuningbiru, ungu-hijau, biru-kuning, hijau-merah
- biru-kuning, hijau-ungu, kuning-biru, merah-hijau

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



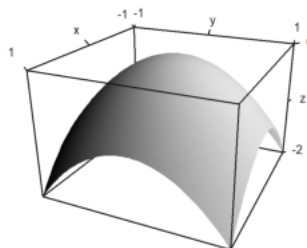
Sumber cahaya dapat diubah dengan `l` dan tombol cursor selama interaksi pengguna. Itu juga dapat diatur dengan parameter.

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

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

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

Press l and cursor keys (return to exit)



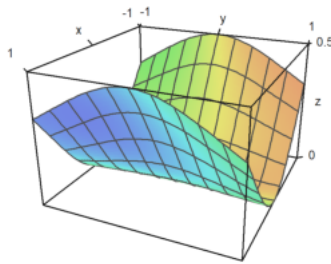
Parameter warna mengubah warna permukaan. Warna garis level juga dapat diubah.

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



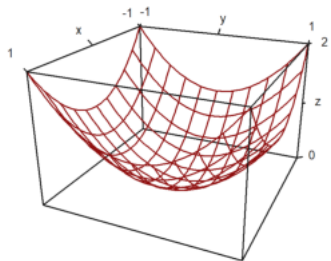

Warna 0 memberikan efek pelangi yang istimewa.

```
>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 mencakup plot implisit. Plot ini menunjukkan himpunan nol suatu fungsi dalam tiga variabel. Solusi dari

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

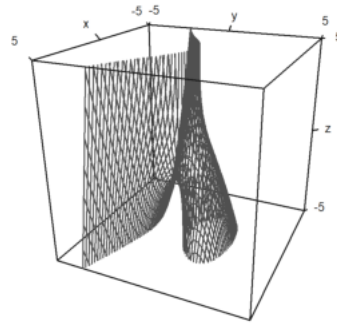
dapat divisualisasikan dalam potongan yang sejajar dengan bidang x-y-, x-z- dan y-z.

- implisit=1: dipotong sejajar bidang y-z
- implisit=2: dipotong sejajar dengan bidang x-z
- implisit=4: dipotong sejajar bidang x-y

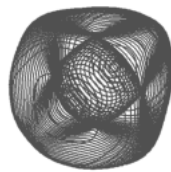
Tambahkan nilai-nilai ini, jika Anda mau. Dalam contoh kita memplot

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

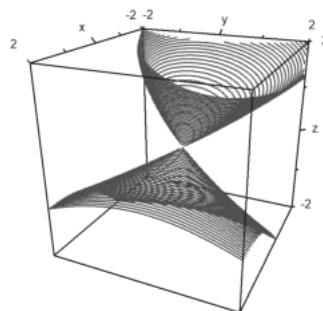
```
>plot3d("x^2+y^3+z*y-1",r=5,implicit=3):
```



```
>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)^2)-d",r=2,<frame,>im
```



```
>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 $fx(x,y)$, $fy(x,y)$, $fz(x,y)$.

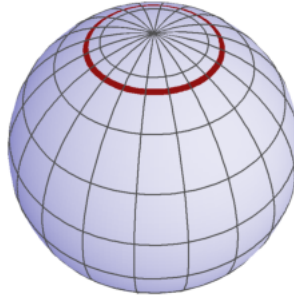
$$\gamma(t, s) = (x(t, s), y(t, s), z(t, s))$$

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

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

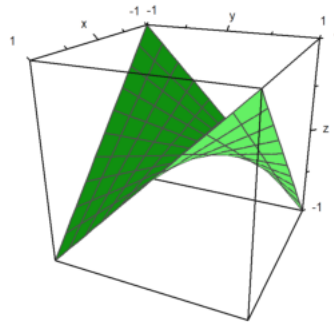
Dalam contoh berikut, kita menggunakan vektor nilai t dan vektor kolom nilai s untuk membuat parameter permukaan bola. Dalam gambar kita dapat menandai wilayah, dalam kasus kita wilayah kutub.

```
>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.

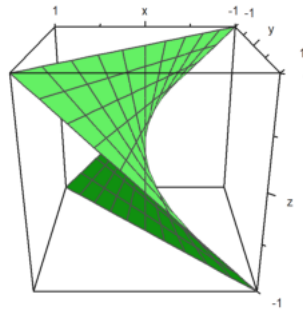
```
>t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,t*s,grid=10):
```



Namun, kita bisa membuat berbagai macam permukaan. Berikut adalah permukaan yang sama sebagai suatu fungsi

$$x = yz$$

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



Dengan lebih banyak usaha, kita dapat menghasilkan banyak permukaan.

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

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

dengan

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

Kami mendistorsi ini dengan sebuah faktor

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

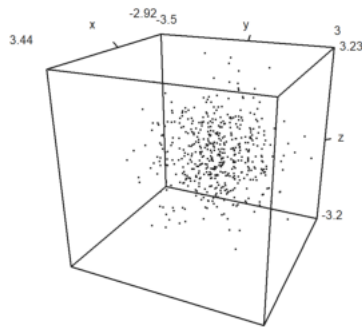
```
>t=linspace(0,2pi,320); s=linspace(-pi/2,pi/2,160)'; ...
d=1+0.2*(cos(4*t)+cos(8*s)); ...
plot3d(cos(t)*cos(s)*d,sin(t)*cos(s)*d,sin(s)*d,hue=1, ...
light=[1,0,1],frame=0,zoom=5):
```



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

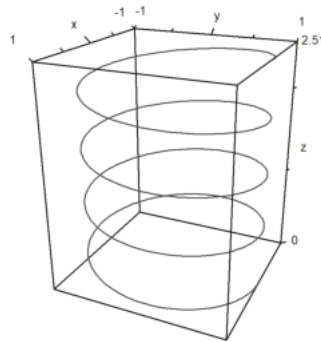
Gayanya sama seperti di plot2d dengan `points=true`;

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

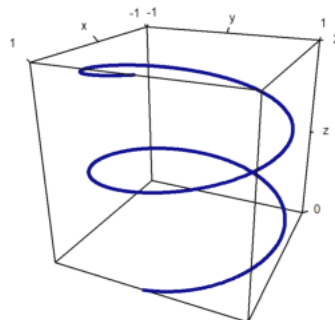


Dimungkinkan juga untuk memplot kurva dalam 3D. Dalam hal ini, lebih mudah untuk menghitung terlebih dahulu titik-titik kurva. Untuk kurva pada bidang kita menggunakan barisan koordinat dan parameter `wire=true`.

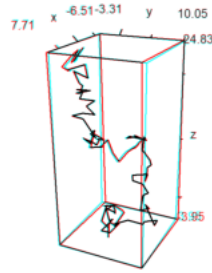
```
>t=linspace(0,8pi,500); ...
plot3d(sin(t),cos(t),t/10,>wire,zoom=3):
```



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

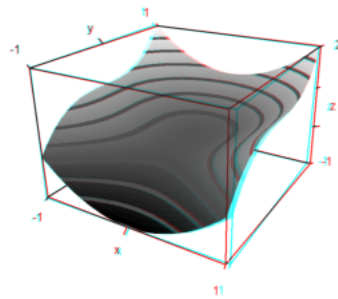


```
>X=cumsum(normal(3,100)); ...
plot3d(X[1],X[2],X[3],>anaglyph,>wire):
```



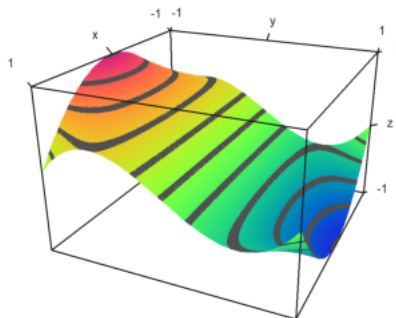
EMT juga dapat membuat plot dalam mode anaglyph. Untuk melihat plot seperti itu, Anda memerlukan kacamata berwarna merah/sian.

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



Seringkali skema warna spektral digunakan untuk plot. Ini menekankan ketinggian fungsinya.

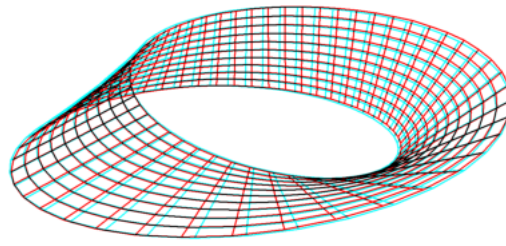
```
>plot3d("x^2+y^3-y",>spectral,>contour,zoom=3.2):
```



Euler juga dapat memplot permukaan yang diparameterisasi, jika parameternya adalah nilai x, y, dan z dari gambar kotak persegi panjang di ruang tersebut.

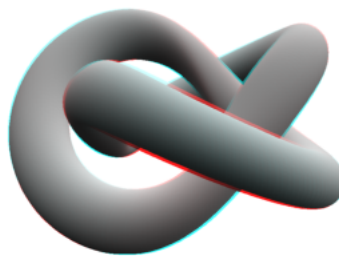
Untuk demo berikut, kami menyiapkan parameter u- dan v-, dan menghasilkan koordinat ruang dari parameter tersebut.

```
>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

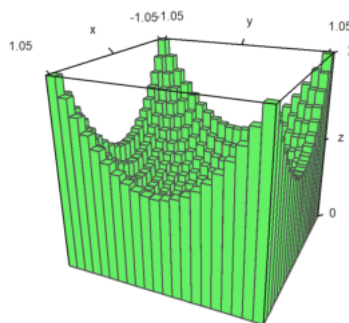
Plot batang juga dimungkinkan. Untuk itu, kita harus menyediakannya

- 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.

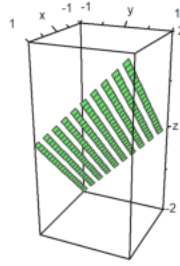
Dalam contoh ini, pertama-tama kita menghitung nilainya. Kemudian kita sesuaikan x dan y, sehingga vektor-vektornya berpusat pada nilai yang digunakan.

```
>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):
```



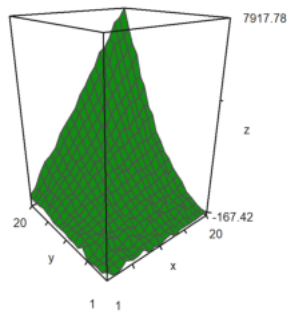
Dimungkinkan untuk membagi plot suatu permukaan menjadi dua bagian atau lebih.

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

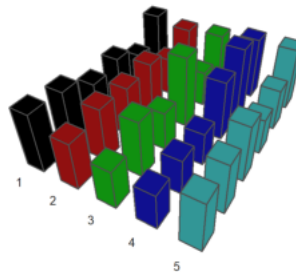


Jika memuat atau menghasilkan matriks data M dari file dan perlu memplotnya dalam 3D. Anda dapat menskalakan matriks ke [-1,1] dengan skala(M), atau menskalakan matriks dengan >zscale. Hal ini dapat dikombinasikan dengan faktor penskalaan individual yang diterapkan sebagai tambahan.

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

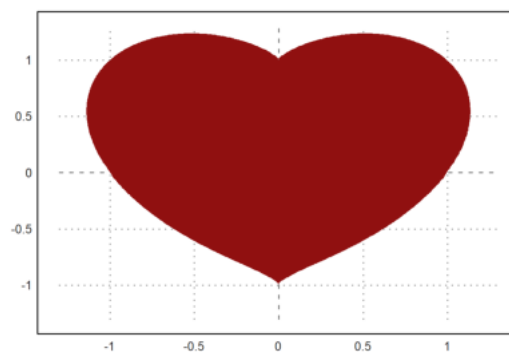


```
>Z=intrandom(5,100,6); v=zeros(5,6); ...
loop 1 to 5; v[#]=getmultiplicities(1:6,Z[#]); end; ...
columnplot3d(v',scols=1:5,ccols=[1:5]):
```



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
```

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

Kami ingin memutar kurva jantung di sekitar sumbu y. Berikut ini adalah ekspresi yang mendefinisikan jantung:

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

Selanjutnya kita atur

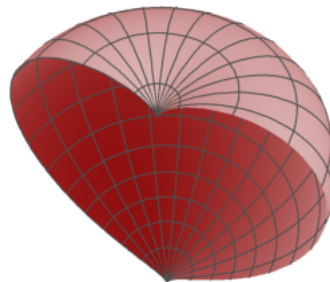
$$x = r \cdot \cos(a), \quad y = r \cdot \sin(a).$$

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

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

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

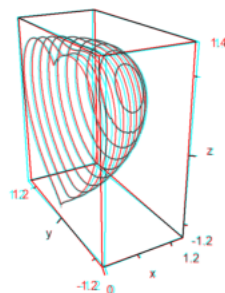
```
>function map f(a) := bisect("fr",0,2;a); ...
t=linspace(-pi/2,pi/2,100); r=f(t); ...
s=linspace(pi,2pi,100)'; ...
plot3d(r*cos(t)*sin(s),r*cos(t)*cos(s),r*sin(t), ...
>hue,<frame,color=red, zoom=4, amb=0, max=0.7, grid=12, height=50°):
```



Berikut ini adalah plot 3D dari gambar di atas yang diputar 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

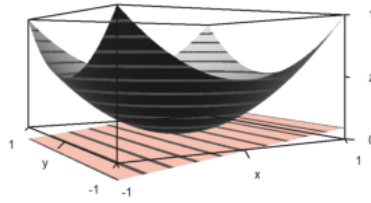
Fungsi plot3d bagus untuk dimiliki, tetapi tidak memenuhi semua kebutuhan. Selain rutinitas yang lebih mendasar, dimungkinkan untuk mendapatkan plot berbingkai dari objek apa pun yang Anda sukai.

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

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

Sekarang framedplot() menyediakan bingkai, dan mengatur tampilan.


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

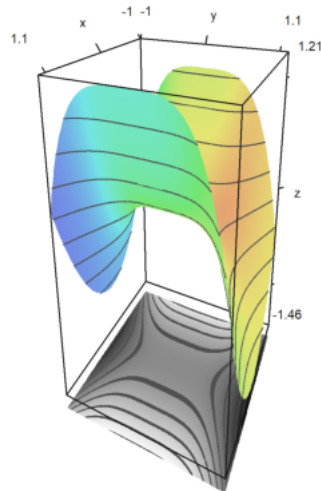


Dengan cara yang sama, Anda dapat memplot bidang kontur secara manual. Perhatikan bahwa `plot3d()` menyetelel jendela ke `fullwindow()` secara default, tetapi `plotcontourplane()` berasumsi demikian.

```
>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):
```



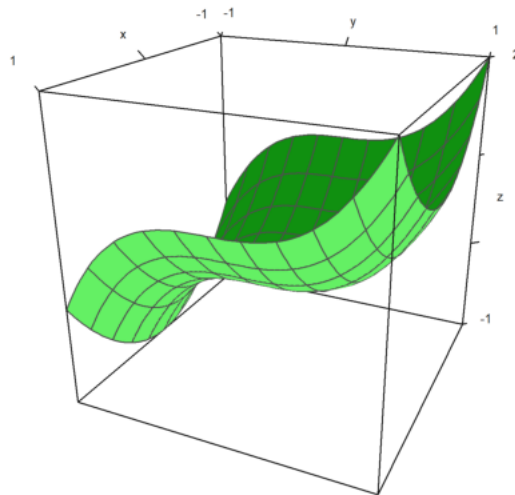
Animasi

Euler dapat menggunakan frame untuk melakukan pra-komputasi animasi.

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

Silakan pelajari sumber rotasi untuk melihat lebih detail.

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



Menggambar Povray

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

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

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

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

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

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

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

```
>load povray;
```

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

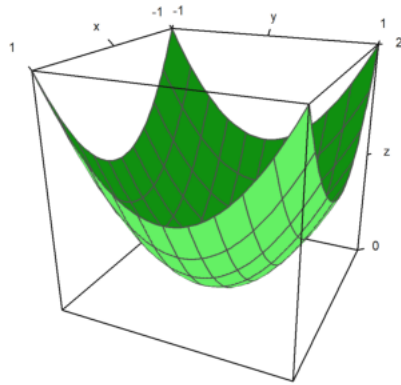
```
>defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

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

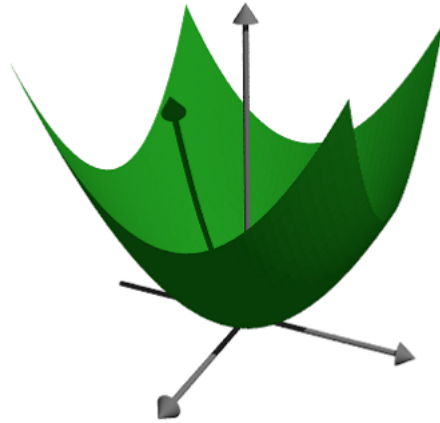
Untuk kesan pertama, kami memplot fungsi sederhana. Perintah berikut menghasilkan file povray di direktori pengguna Anda, dan menjalankan Povray untuk penelusuran sinar file ini.

Jika Anda memulai perintah berikut, GUI Povray akan terbuka, menjalankan file, dan menutup secara otomatis. Karena alasan keamanan, Anda akan ditanya apakah Anda ingin mengizinkan file exe 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) :
```

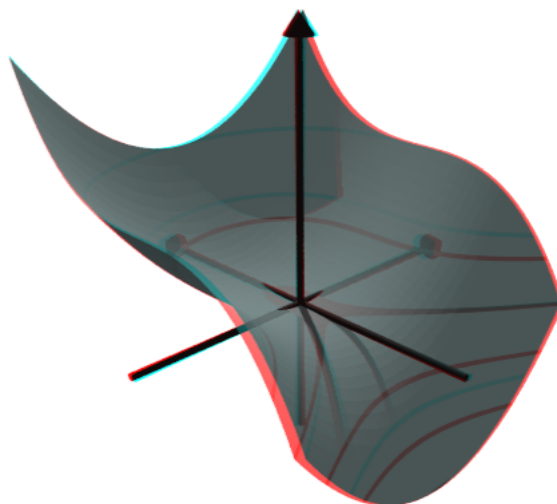


```
>pov3d("x^2+y^2", zoom=3);
```



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

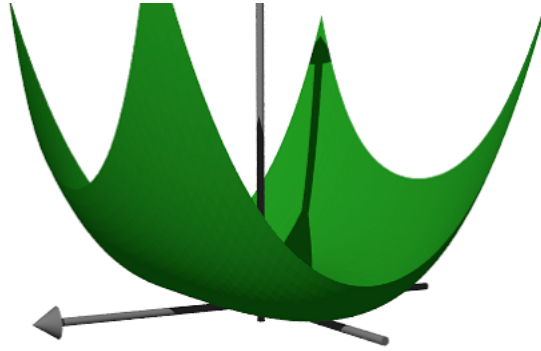
```
>pov3d("x^2+y^3", axiscolor=red, angle=-45°, >anaglyph, ...
  look=povlook(cyan, 0.2), level=-1:0.5:1, zoom=3.8);
```



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);
```

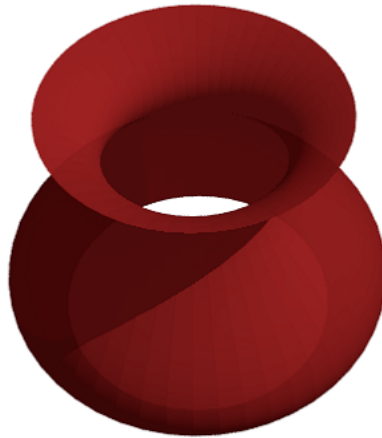


Merencanakan dengan Koordinat

Alih-alih fungsi, kita bisa memplot dengan koordinat. Seperti di plot3d, kita memerlukan tiga matriks untuk mendefinisikan objek.

Dalam contoh ini kita memutar suatu fungsi di sekitar sumbu z.

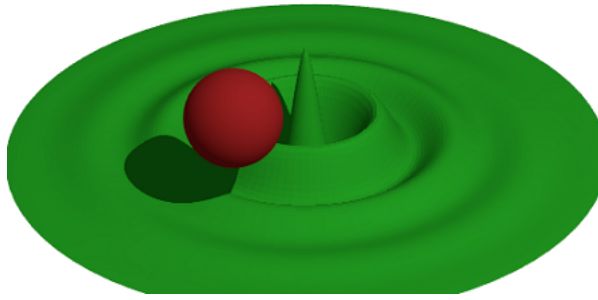
```
>function f(x) := x^3-x+1; ...  
x=-1:0.01:1; t=linspace(0,2pi,50)'; ...  
Z=x; X=cos(t)*f(x); Y=sin(t)*f(x); ...  
pov3d(X,Y,Z,angle=40°,look=povlook(red,0.1),height=50°,axis=0, zoom=4, light=[10,5,15]);
```



Pada 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 contoh berikut. Perhatikan bahwa plot3d menskalakan plot, sehingga cocok dengan kubus satuan.

```
>r=linspace(0,1,80); phi=linspace(0,2pi,80)'; ...  
x=r*cos(phi); y=r*sin(phi); z=exp(-5*r)*cos(8*pi*r)/3; ...  
pov3d(x,y,z, zoom=6, axis=0, height=30°, add=povsphere([0.5,0,0.25],0.15,povlook(red)), ...  
w=500,h=300);
```



Dengan metode peneduh canggih Povray, sangat sedikit titik yang dapat menghasilkan permukaan yang sangat halus. Hanya pada batas-batas dan dalam bayangan, triknya mungkin terlihat jelas.

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

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

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

Persamaan permukaannya adalah $[x,y,Z]$. Kami menghitung dua turunan dari x dan y dan mengambil perkalian silangnya sebagai normal.

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

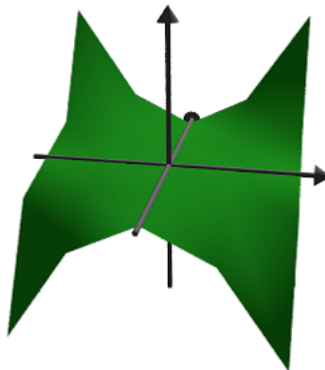
Kami mendefinisikan normal sebagai produk silang dari turunan ini, dan mendefinisikan fungsi koordinat.

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

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

Kami hanya menggunakan 25 poin.

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



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

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, tiga fungsi koordinat sebagai ekspresi simbolik.

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

Kemudian kedua vektor turunan ke x dan y .

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

Sekarang normalnya, yaitu perkalian silang kedua turunannya.

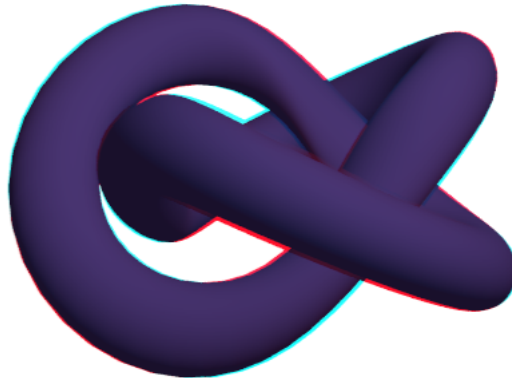
```
>dn &= crossproduct(dx,dy);
```

Kami sekarang mengevaluasi semua ini secara numerik.

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

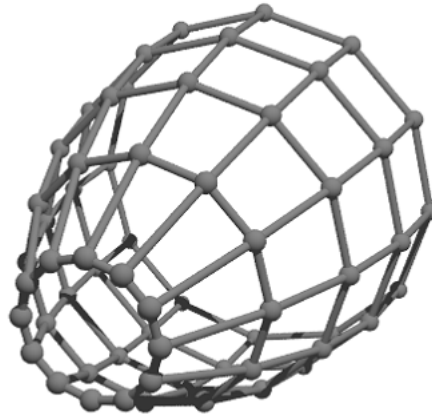
Vektor normal adalah evaluasi ekspresi simbolik dn[i] untuk i=1,2,3. Sintaksnya adalah "&ekspresi" (parameter). Ini adalah alternatif dari metode pada contoh sebelumnya, di mana kita mendefinisikan ekspresi simbolik NX, NY, NZ terlebih dahulu.

```
>pov3d(X(x,y),Y(x,y),Z(x,y),>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));
```



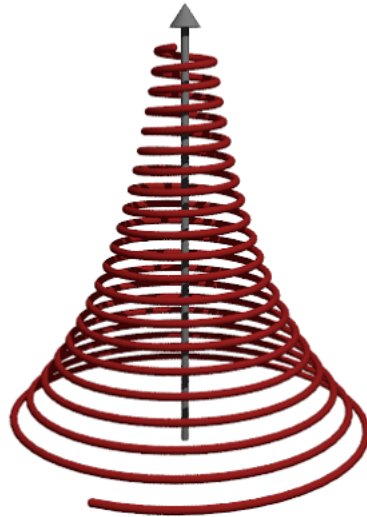
Kami juga dapat menghasilkan grid dalam 3D.

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



Dengan povgrid(), kurva dimungkinkan.

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



Objek Povray

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

Kami memulai output dengan povstart().

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

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

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

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

String tersebut berisi 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 Anda lihat, kami menambahkan tekstur pada objek dalam tiga warna berbeda.

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

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

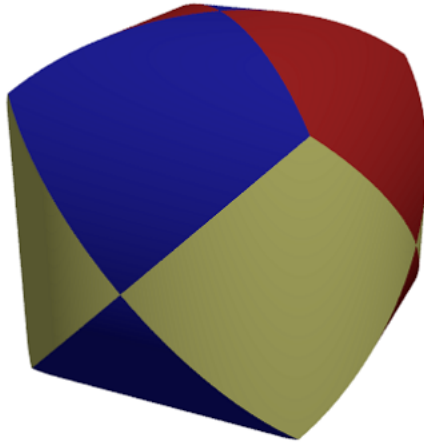
```
texture { pigment { color rgbf <0.101961,0.2,0.301961,0.1> } }  
finish { ambient 0.5 }
```

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

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

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

```
>povend;
```



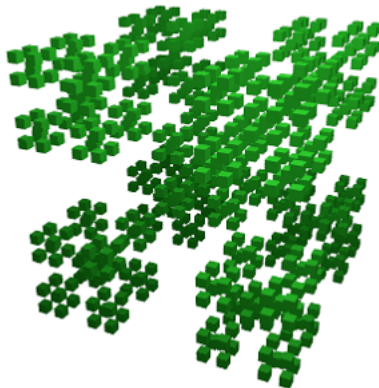
Fungsi berikut menghasilkan fraktal secara rekursif.

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

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

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

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



Perbedaan memungkinkan pemisahan satu objek dari objek lainnya. Seperti persimpangan, ada bagian dari objek CSG di Povray.

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

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

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


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

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

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

Kami membuat kubus kedua, dan memutar serta menskalakannya sedikit.

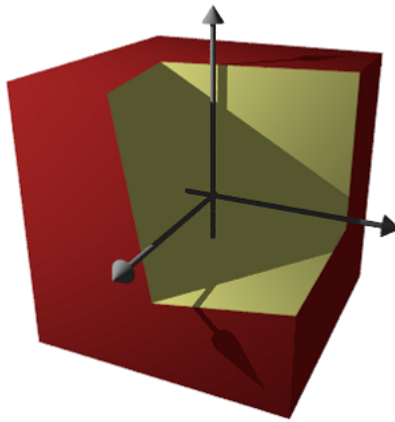
```
>c2=povobject("mycube",povlook(yellow),translate=[1,1,1], ...  
rotate=xrotate(10°)+yrotate(10°), scale=1.2);
```

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. Namun hasilnya terlihat jauh lebih baik.

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)+pow (z,2)-pow (c,2),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);
```

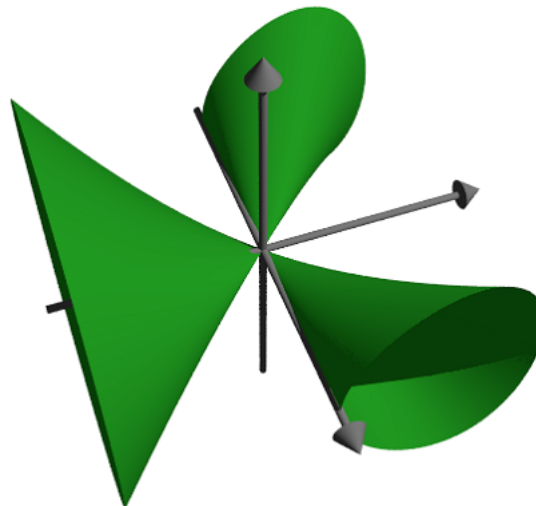
```
>povstart (angle=25°,height=10°);  
>writeln(povsurface ("pow (x,2)+pow (y,2)*pow (z,2)-1",povlook (blue),povbox (-2,2,"")));  
>povend();
```



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

Buat permukaan implisit. Perhatikan sintaksis yang berbeda dalam 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 menggambarinya dengan informasi tambahan.

Kita ingin memaksimalkan xy pada kondisi $x+y=1$ dan mendemonstrasikan sentuhan tangensial 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 #declare. 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 mendefinisikan dua cakram, yang akan berpotongan dengan permukaan.

```
>c1=povdisc([0.5,0.5,0],[1,1,0],2); ...
l1=povdisc([0,0,1/4],[0,0,1],2);
```

Tulis permukaannya dikurangi kedua cakram.

```
>writeln(povdifference(mesh,povunion([c1,l1]),povlook(green)));
```

Tuliskan kedua perpotongan tersebut.

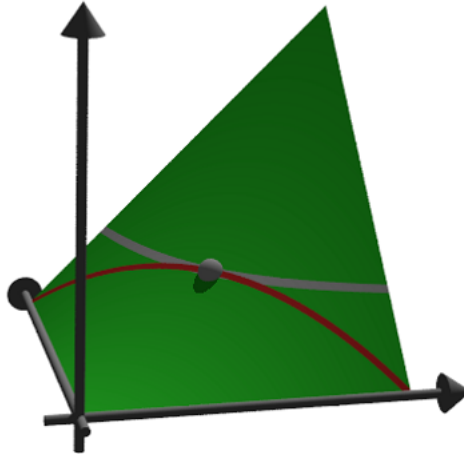
```
>writeln (povintersection ([mesh,c1],povlook (red))); ...  
writeln (povintersection ([mesh,l1],povlook (gray)));
```

Tulis poin maksimal.

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

Tambahkan sumbu dan selesai.

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



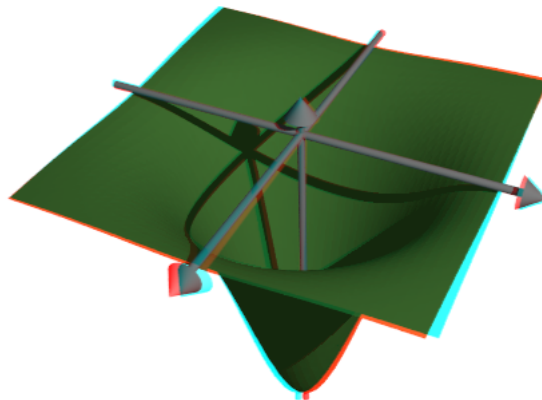
Anaglyph di Povray

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

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

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 pembuatan adegan ke dalam fungsi, dan menjalankannya dua kali dengan nilai 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]);
```

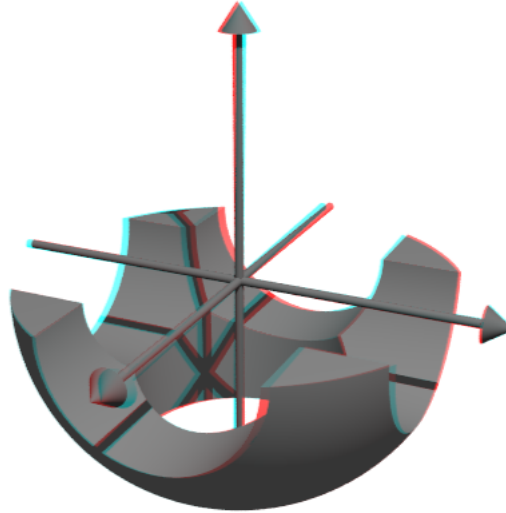
```

obj=povdifference(s,un,povlook(red));
writeln(obj);
writeAxes();
endfunction

```

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

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



Mendefinisikan Objek sendiri

Antarmuka povray Euler berisi banyak objek. Namun Anda tidak dibatasi pada hal ini. Anda dapat membuat objek sendiri, yang menggabungkan objek lain, atau merupakan objek yang benar-benar baru.

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

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

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

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

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

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

Sekarang kita tempatkan objek-objek tersebut ke dalam sebuah adegan. Untuk tampilannya kami menggunakan Phong Shading.

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

```

```
>povend();
```

memanggil program Povray. Namun, jika terjadi kesalahan, kesalahan tersebut tidak ditampilkan. 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 menyelesaikan

$$Ax \leq b, \quad x \geq 0, \quad c \cdot x \rightarrow \text{Max.}$$

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

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

Pertama, mari kita periksa, apakah contoh ini memiliki solusi atau tidak.

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

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

Ya, benar.

Selanjutnya kita mendefinisikan dua objek. Yang pertama adalah pesawat

$$a \cdot x \leq b$$

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

Kemudian kita tentukan perpotongan 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 bisa merencanakan adegan tersebut.

```
>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 sekeliling optimal.

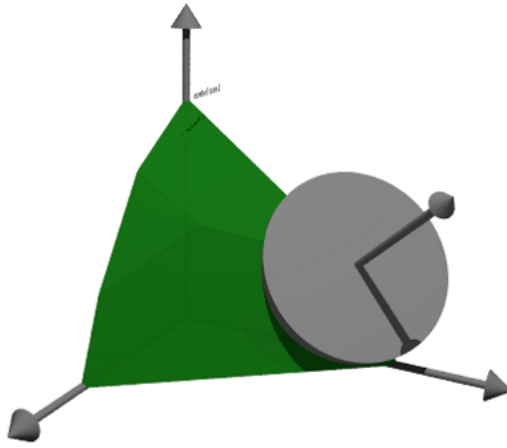
```
>writeln(povintersection([povsphere(x,0.5),povplane(c,c*x')], ...  
    povlook(red,0.9)));
```

Dan kesalahan pada arah yang optimal.

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

Kami 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();
```



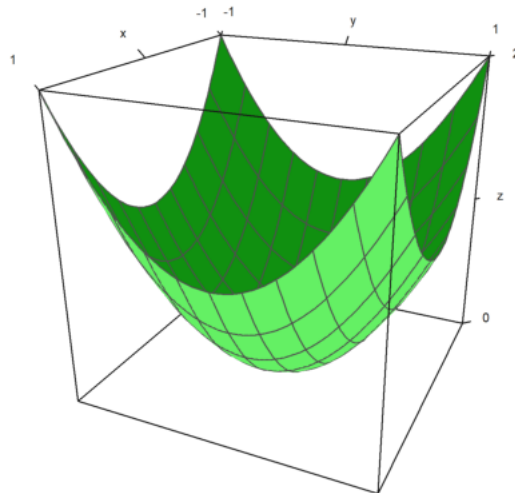
Contoh Lainnya

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

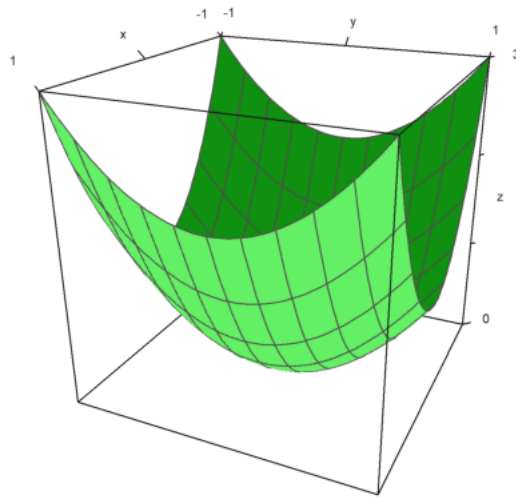
[Contoh/Contoh Bola Dandelin](#)
[Contoh/Contoh/Donat Matematika](#)
[Contoh/Simpul Trefoil](#)
[Contoh/Optimalisasi dengan Penskalaan Affine](#)

1. Menggambar Grafik Fungsi Dua Variabel dalam Bentuk Ekspresi Langsung

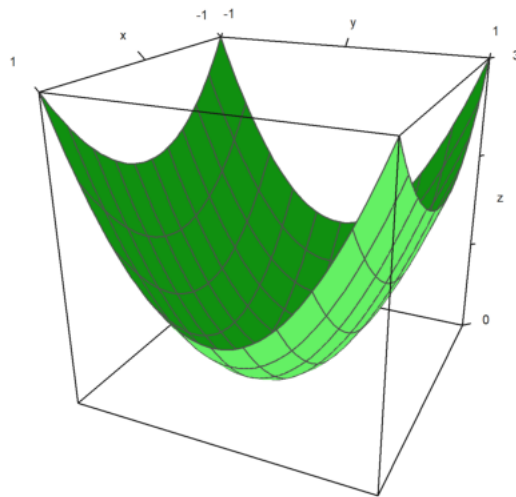
```
>plot3d("x^2+y^2") :
```



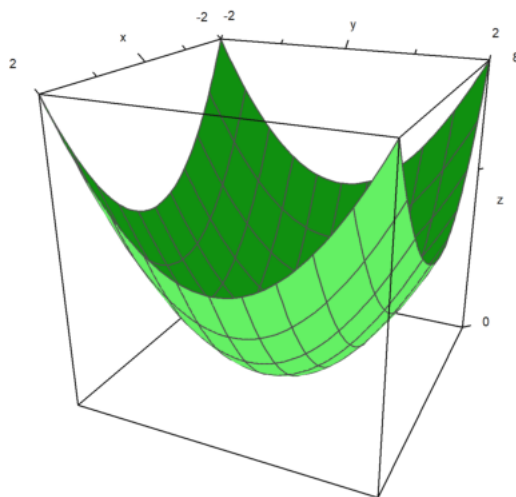
```
>plot3d("2*x^2+y^2") :
```



```
>plot3d("x^2+2*y^2");
```

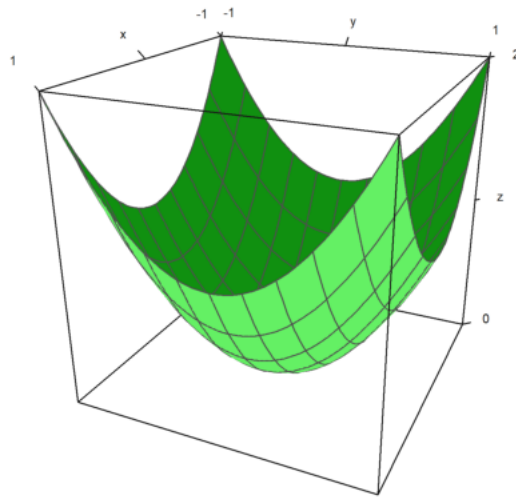


```
>plot3d("x^2+y^2", -2, 2, -2, 2):
```

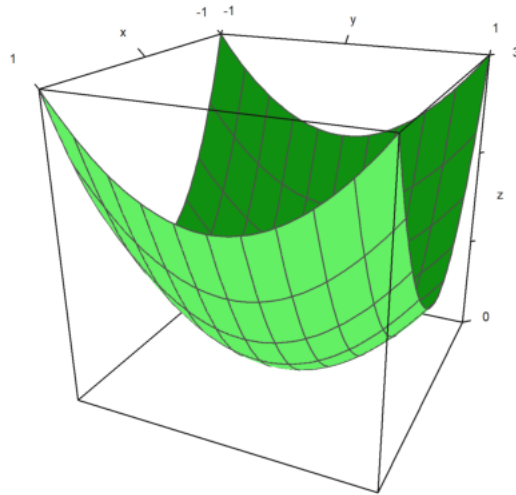


2. Menggambar Grafik Fungsi Dua Variabel yang Rumusnya Disimpan dalam Variabel Ekspresi

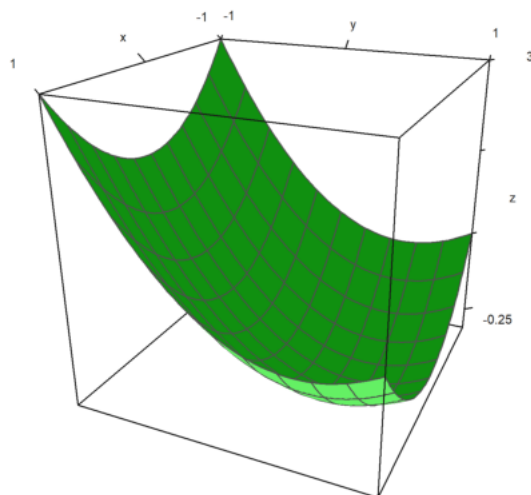
```
>expr := ("x^2 + y^2");...
plot3d (expr):
```



```
>expr := ("2*x^2 + y^2"); ...
plot3d (expr, >user): // >user digunakan untuk memberikan tampilan plot 3d yang bisa diputar kiri kanan atas
```



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



(Penjelasan singkat)

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.

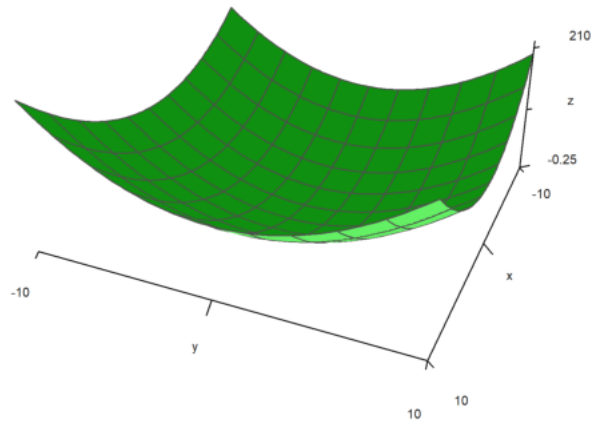
Terdapat beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.

fscale: skala untuk nilai fungsi (standarnya adalah <fscale).

scale: angka atau vektor 1x2 untuk menskalakan ke arah x dan y.

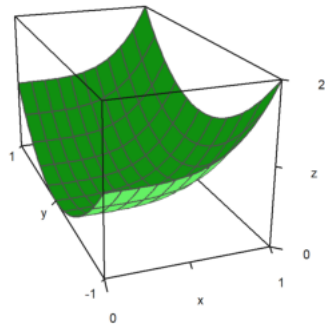
frame: jenis bingkai (default 1).

```
>// fscale: skala untuk nilai fungsi(membuat plot 3d terlihat lebih mendatar) ...  
// scale: membuat plot lebih terlihat besar/kecil...  
// frame: membuat garis-garis kubus tidak terlihat. Hanya menyisakan garis sumbu saja...  
expr := ("x^2 + y^2 - y");...  
plot3d (expr,r=10,n=80,fscale=4,scale=1.2,frame=3):
```

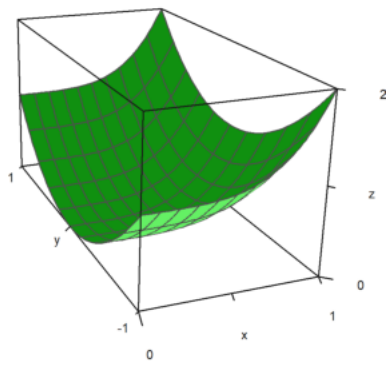


3. Menggambar Grafik Fungsi Dua Variabel yang Fungsinya Didefinisikan sebagai Fungsi Numerik

```
>plot3d("x^4+y^2",a=0,b=1,c=-1,d=1,angle=-20°,height=20°): // a,b =rentang x. c,d =rentang y
```

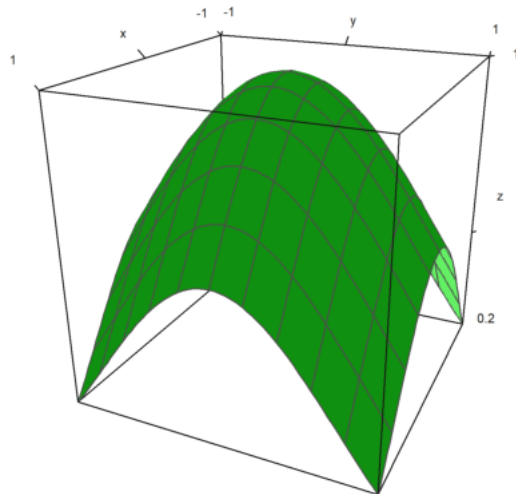


```
>plot3d("x^4+y^2",a=0,b=1,c=-1,d=1,angle=-20°,height=20°, ...  
center=[0.4,0,0],zoom=3):
```

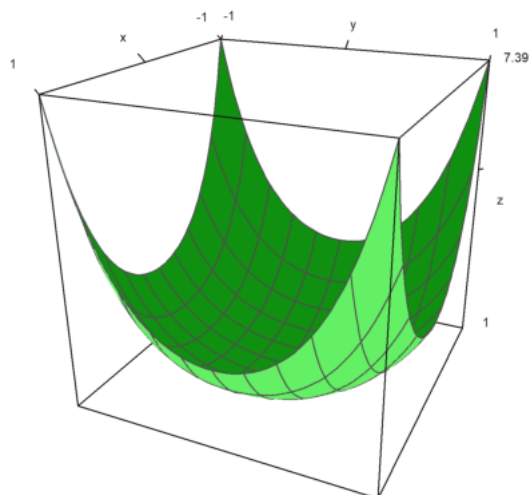


4. Menggambar Grafik Fungsi Dua Variabel yang Fungsinya Didefinisikan sebagai Fungsi Simbolik

```
>function f(x,y) :=exp(-(x^2+y^2)/2) * cos(x*y);
>plot3d ("f(x,y)");
```



```
>function f(x,y) :=exp(x^2 + y^2);
>plot3d ("f(x,y)");
```

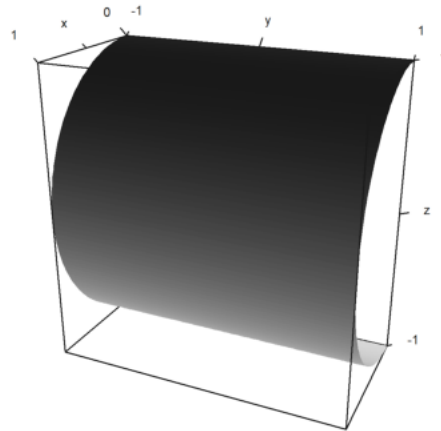


5. Menggambar Data x , y , z pada ruang Tiga Dimensi (3D)

```

>t=linspace(0,2pi,180);
>s=linspace(-pi/2,pi/2,90)';
>x=cos(s);
>y=cos(t);
>z=sin(s);
>plot3d(x,y,z,>hue):

```

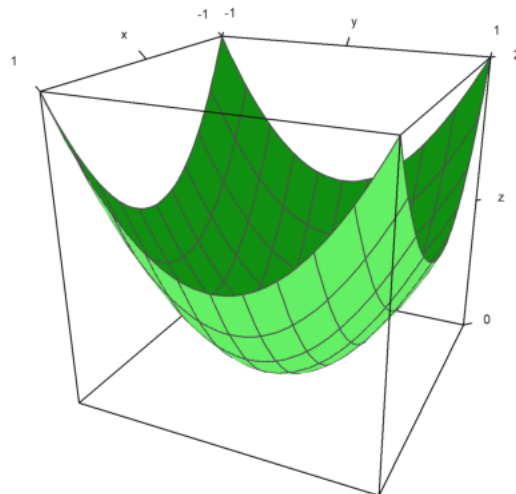


6. Membuat Gambar Grafik Tiga Dimensi (3D) yang Bersifat Interaktif dan animasi grafik 3D

```

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

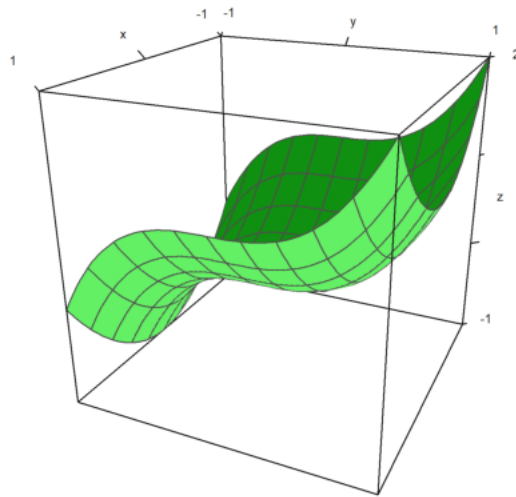
```



```

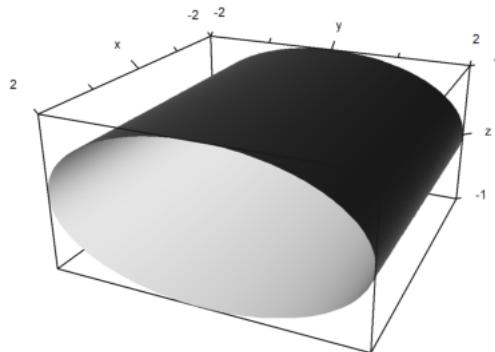
>function testplot () := plot3d("x^2+y^3"); ...
rotate("testplot"); testplot():

```



7. Menggambar Fungsi Parametrik Tiga Dimensi (3D)

```
>t=linspace(0,2pi,180);
>s=linspace(-pi/2,pi/2,90)';
>x= 2*sin(s);
>y= 2*sin(t);
>z=cos(t);
>plot3d(x,y,z,>hue):
```



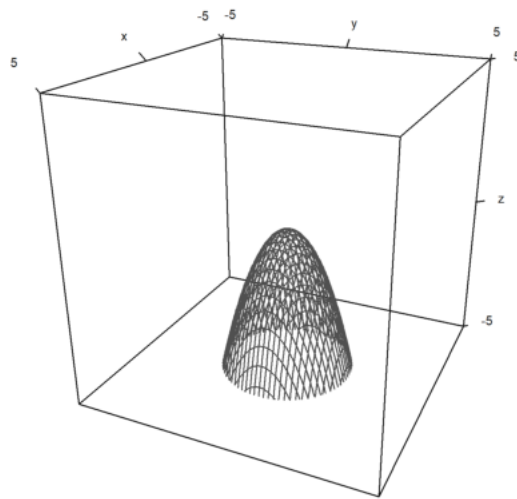
8. Menggambar Fungsi Implisit Tiga Dimensi (3D)

implicit = 1: memotong sejajar dengan bidang-y-z

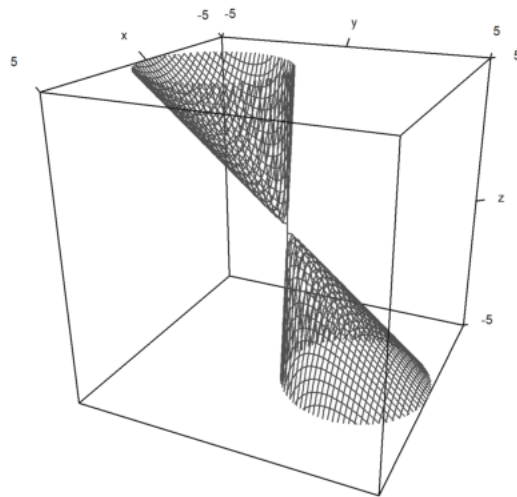
implicit = 2: memotong sejajar dengan bidang x-z

implicit = 4: memotong sejajar dengan bidang x-y

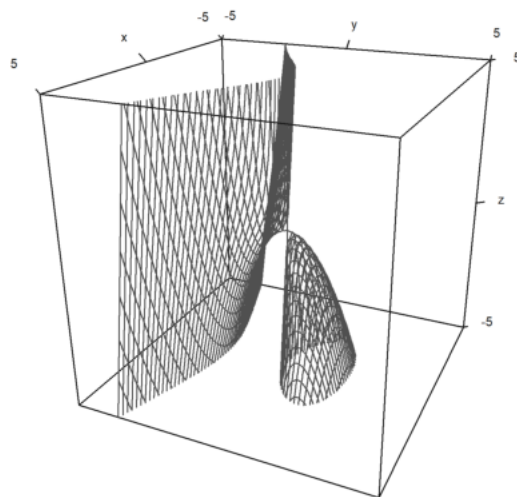
```
>plot3d("x^2+y^2+z",r=5,implicit=3):
```



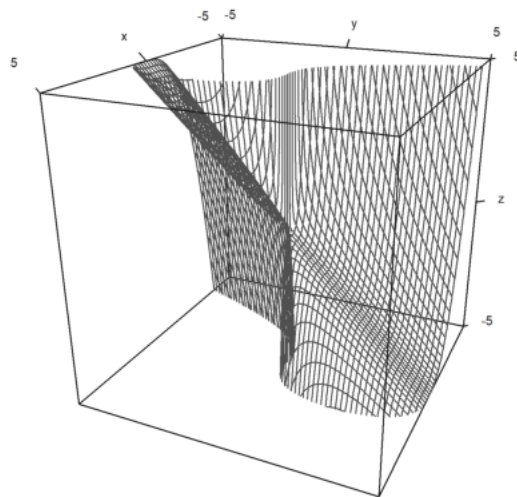
```
>plot3d("x^2+y^2+z*y", r=5, implicit=3):
```



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

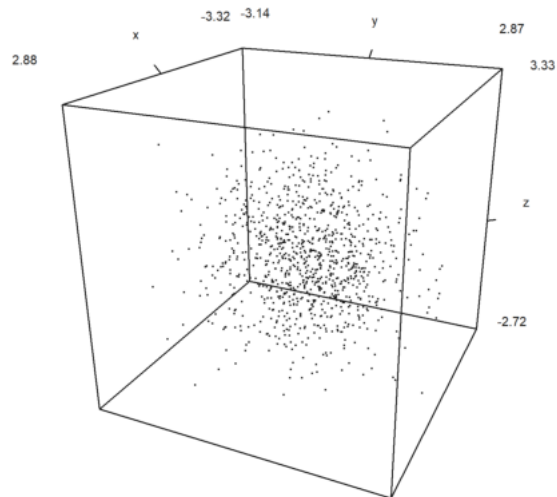


```
>plot3d("x^3+y^2+z*y", r=5, implicit=3):
```

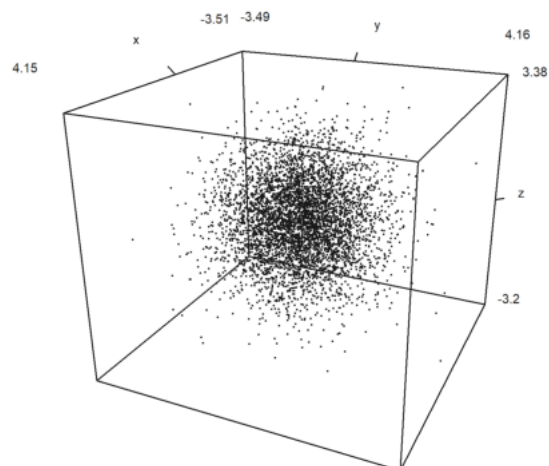


9. Menggambar Titik pada ruang Tiga Dimensi (3D)

```
>n=500;
>plot3d(normal(1,2*n),normal(1,2*n),normal(1,2*n),points=true,style=".") :
```

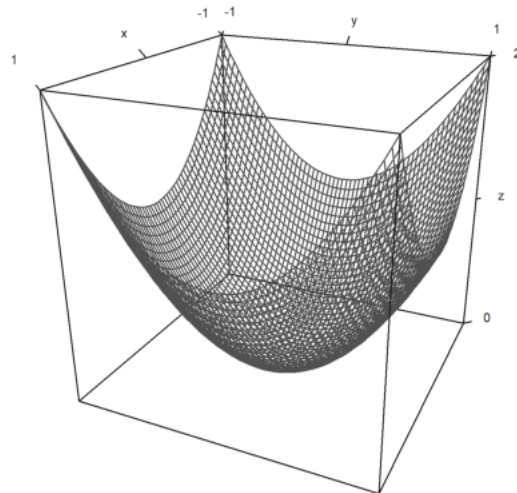


```
>n=500;
>plot3d(normal(10,n),normal(10,n),normal(10,n),points=true,style=".") :
```



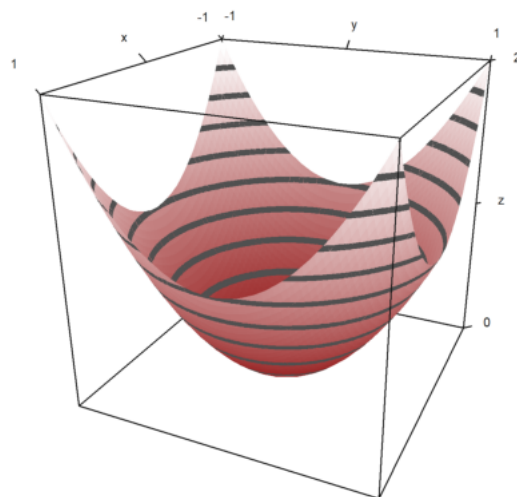
10. Mengatur tampilan, warna dan sudut pandang gambar permukaan Tiga Dimensi (3D)

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

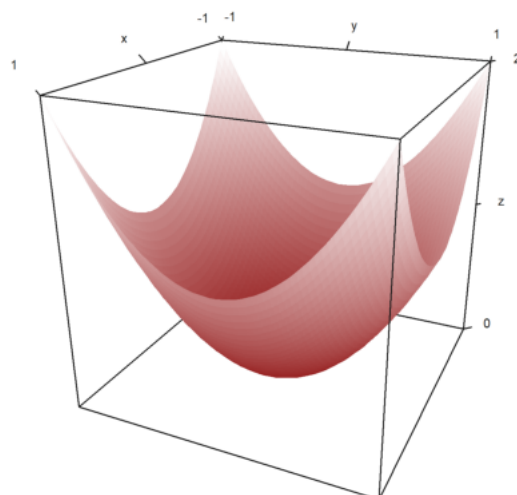


menggunakan `>contour` ; `>spectral` ; `>cp` ; `>hue`; untuk memberikan warna yang diinginkan

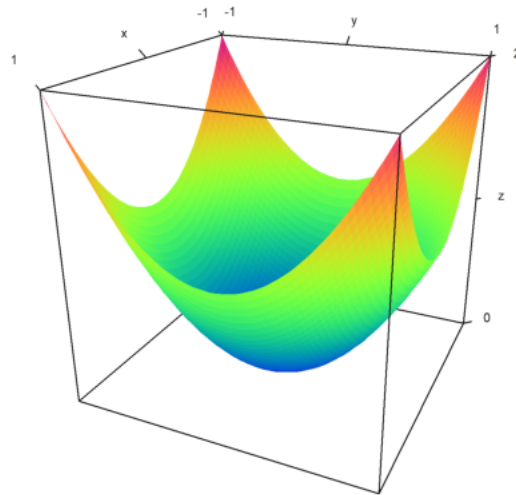
```
>plot3d("x^2+y^2", >contour, color=red): //>contour untuk memunculkan warna
```



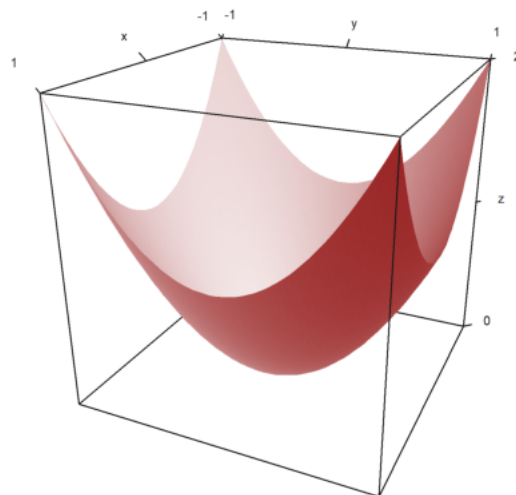
```
>plot3d("x^2+y^2", >spectral, color=red): //menghilangkan garis
```



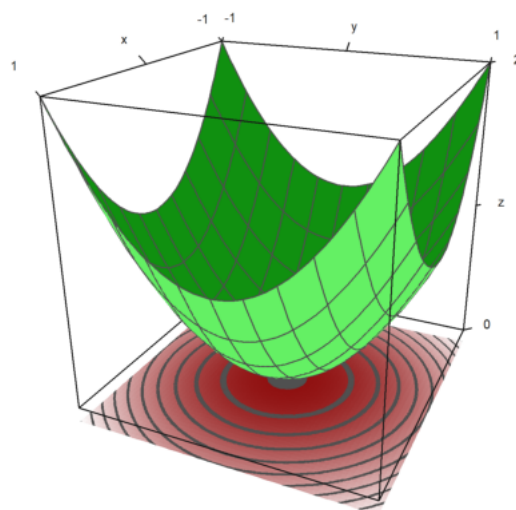
```
>plot3d("x^2+y^2", >spectral, contourcolor=red): // contourcolor untuk memberikan warna dari biru ke merah
```



```
>plot3d("x^2+y^2", >hue, color=red): // memberikan warna berdasarkan gradien
```

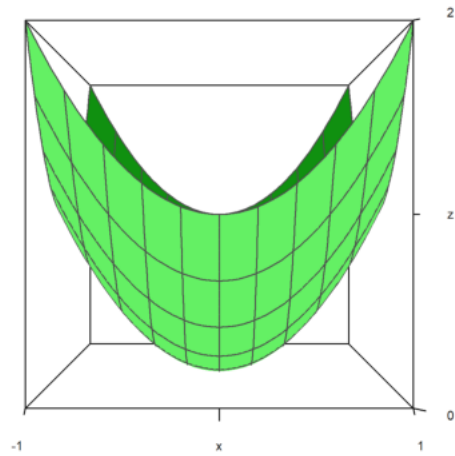


```
>plot3d("x^2+y^2", >cp, color=red): //>cp memberikn warna pada dasar gambar 3d
```

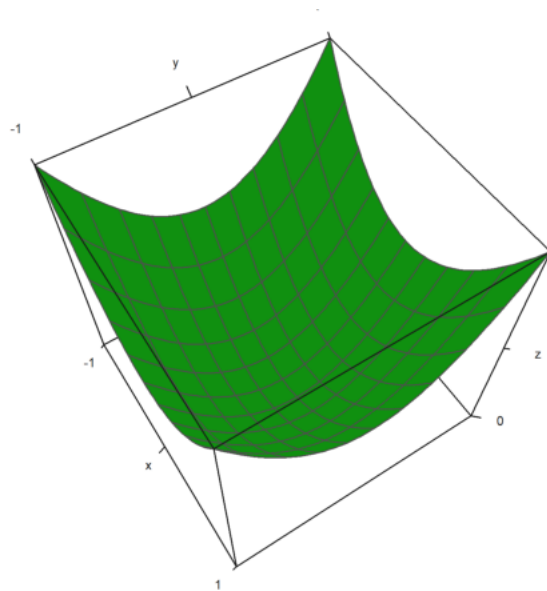


Menggunakan angle=... dan height=... untuk menentukan sudut pandangnya


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

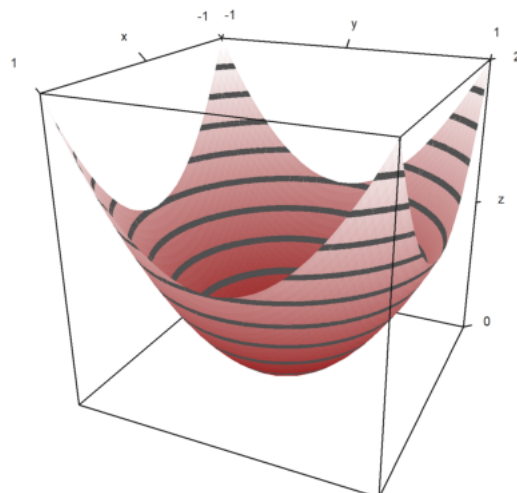


```
>plot3d("x^2+y^2", angle=1, height=1):
```

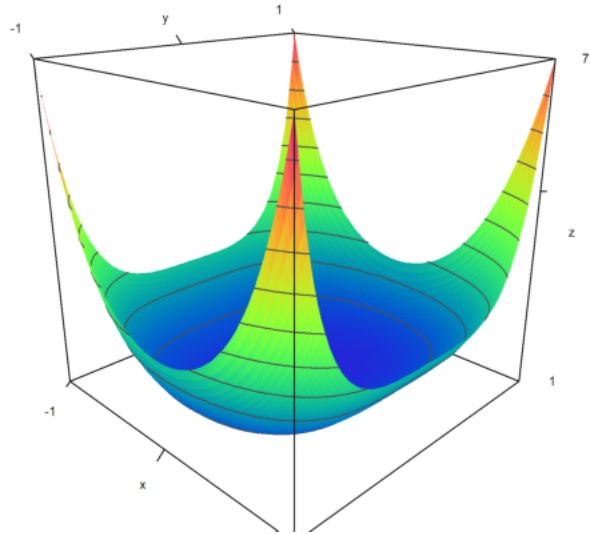


11. Menampilkan kontur dan bidang kontur permukaan Tiga Dimensi (3D)

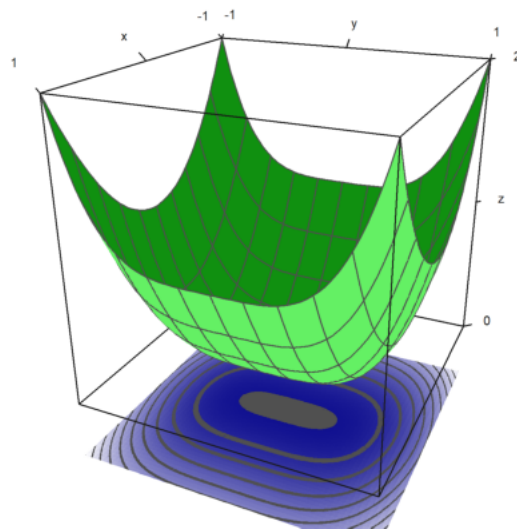
```
>plot3d("x^2+y^2", >contour, color=red):
```



```
>plot3d("exp(x^2+y^4)",r=1,n=100,level="thin", ...  
>contour,>spectral,fscale=1,scale=1.1,angle=45°,height=20°):
```

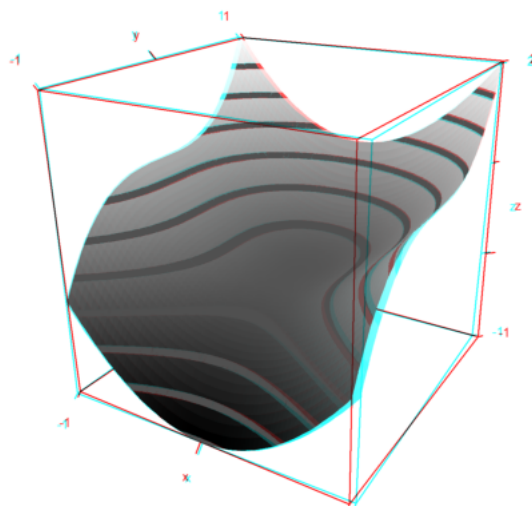


```
>plot3d("x^2+y^4",>cp, cpcolor=blue, cpdelta=0.2):
```

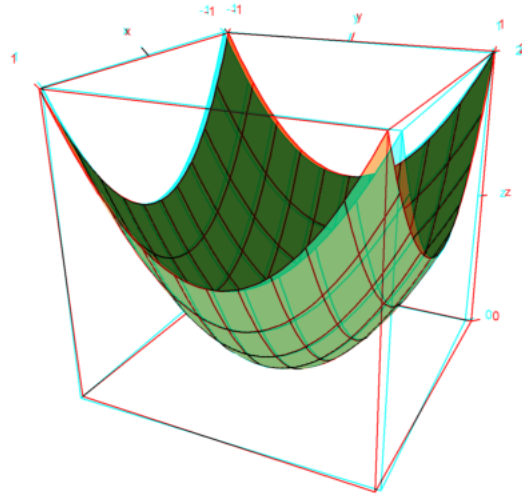


12. Menggambar Grafik Tiga Dimensi alam modus anaglif

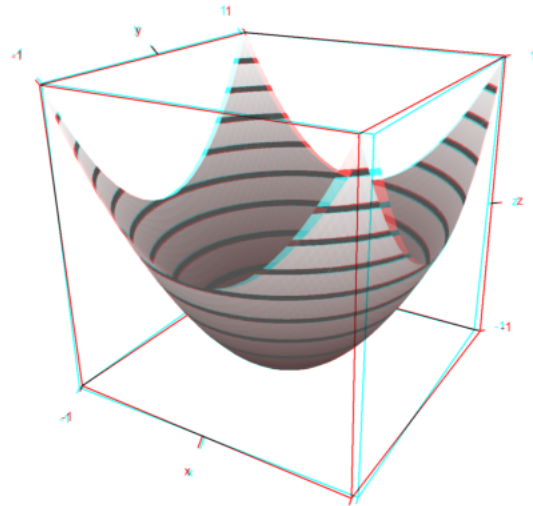
```
>plot3d("x^2+y^3",>anaglyph,>contour,angle=30°):
```



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

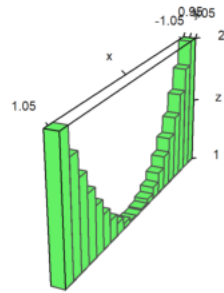


```
> plot3d("x^2+y^2-1",>anaglyph,>contour,color=red,angle=30°):
```



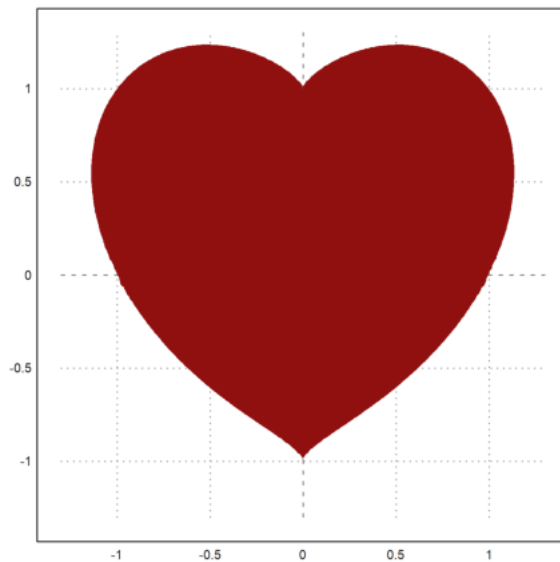
13. Menggambar Diagram Batang Tiga Dimensi

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



14. Menggambar 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):
```



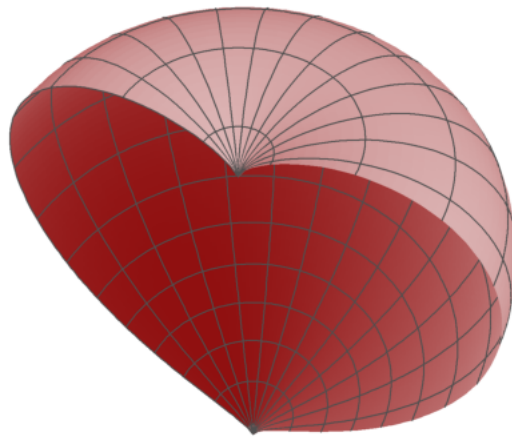
```
>ekspresi &= (x^2+y^2-1)^3-x^2*y^3; $ekspresi
```

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

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

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

```
>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°):
```

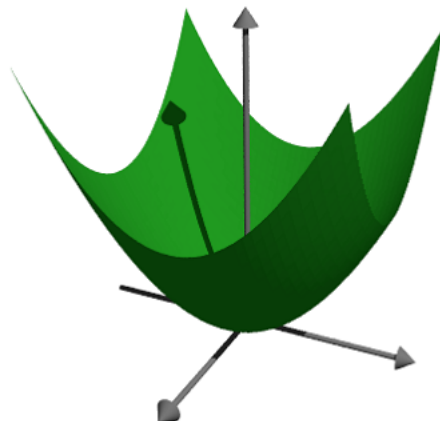


15. Menggambar Grafik 3D dengan Povray di EMT

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

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

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



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

