Tugas Individu Pekan7-8_Muhammad Lutfi Ramadhan_23030630021

Nama : Muhammad Lutfi Ramadhan
Kelas : Matematika B 2023
NIM : 23030630021

Menggambar Plot 3D dengan EMT

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

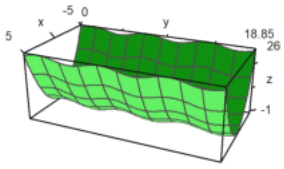
Euler menggambar fungsi seperti itu menggunakan algoritma pengurutan untuk menyembunyikan bagian yang ada di latar belakang. Secara umum, Euler menggunakan proyeksi sentral. Default-nya adalah dari kuadran positif x-y menuju ke asal x=y=z=0, tetapi sudut=0° melihat dari arah sumbu y. Sudut pandang dan ketinggian dapat diubah.

Euler dapat memplot:

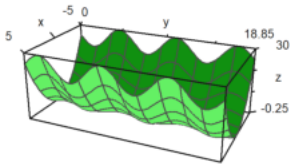
- permukaan dengan bayangan dan garis kontur atau rentang kontur,
- awan titik,
- kurva parametrik,
- permukaan implisit.

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

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




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



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

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

```
Closing bracket missing in function call!
Error in:
aspect(1.5); plot3d("x^2+sin(y)", r=pi: ...
                                       ^
```

Fungsi Dua Variabel

Untuk grafik sebuah fungsi, gunakan:

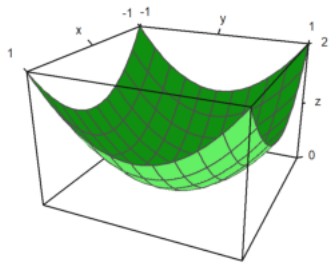
- ekspresi sederhana dalam x dan y,
- nama fungsi dari dua variabel,
- atau matriks data.

Default-nya adalah grid kawat berisi dengan warna berbeda di kedua sisinya. Perhatikan bahwa jumlah interval grid default adalah 10, tetapi plot menggunakan jumlah 40x40 persegi panjang untuk membangun permukaan. Ini dapat diubah.

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

Kita menggunakan default n=40 dan grid=10.

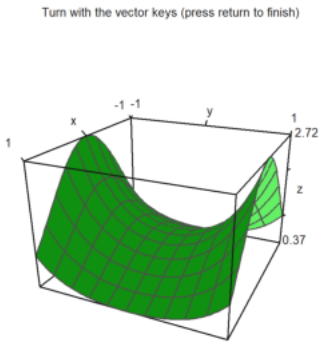
```
>plot3d("x^2+y^2"):
```



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

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

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



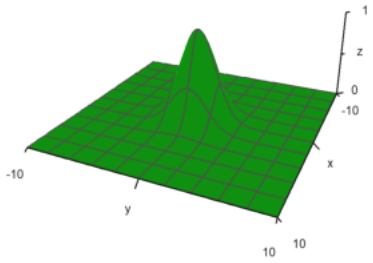
Rentang plot untuk fungsi dapat ditentukan dengan:

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

Ada beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik:

fscale: menskalakan nilai fungsi (default adalah <fscale>).
cale: angka atau vektor 1x2 untuk menskalakan ke arah x dan y.
rame: tipe bingkai (default 1).

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



Pandangan bisa diubah dalam banyak cara:

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

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

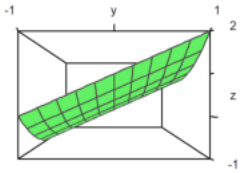
```
>view
```

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

Jarak yang lebih dekat memerlukan pembesaran yang lebih sedikit. Efeknya lebih mirip dengan lensa sudut lebar.

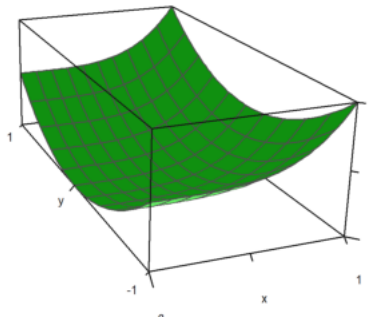
Dalam contoh berikut, angle=0 dan height=0 melihat dari sumbu y negatif. Label sumbu untuk y disembunyikan dalam kasus ini.

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



Plot selalu melihat ke pusat kubus plot. Kamu dapat memindahkan pusat dengan parameter center.

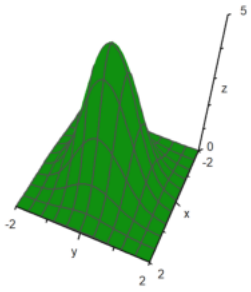
```
>plot3d("x^4+y^2",a=0,b=1,c=-1,d=1,angle=-20°,height=20°, ...
   center=[0.4,0,0],zoom=5):
```



Plot diskalakan agar sesuai dengan kubus unit untuk tampilan. Jadi tidak perlu mengubah jarak atau zoom tergantung pada ukuran plot. Namun, label tetap merujuk pada ukuran yang sebenarnya.

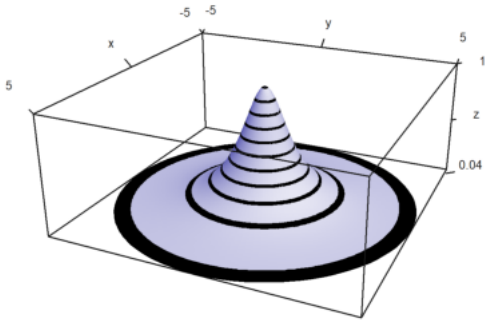
Jika kamu mematikan ini dengan scale=false, kamu harus memastikan bahwa plot tetap sesuai dengan jendela pemetaan, dengan mengubah jarak pandang atau zoom, serta menggerakkan pusat.

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

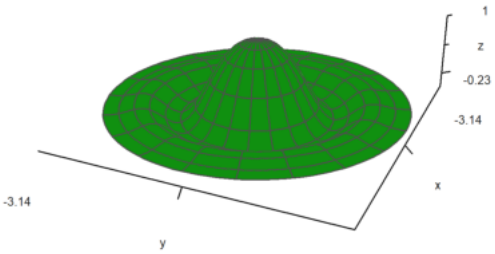


Plot polar juga tersedia. Parameter polar=true menggambar plot polar. Fungsi tetap harus berupa fungsi dari x dan y. Parameter 'fscale' menskalakan fungsi dengan skala sendiri. Jika tidak, fungsi akan diskalakan agar sesuai dengan sebuah kubus.

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



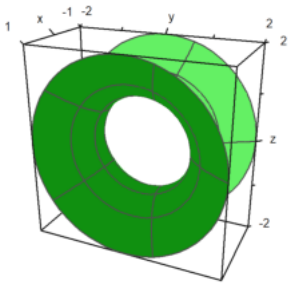

```
>function f(r) := exp(-r/2)*cos(r); ...
 plot3d("f(x^2+y^2)",>polar,scale=[1,1,0.4],r=pi,frame=3,zoom=4):
```



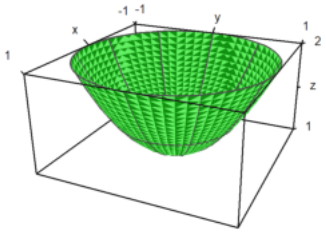
Parameter rotate memutar sebuah fungsi dalam x di sekitar sumbu x.

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

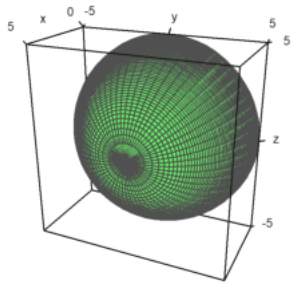
```
>plot3d("x^2+1",a=-1,b=1,rotate=true,grid=5):
```



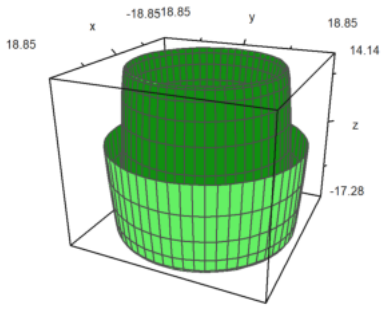

```
>plot3d("x^2+1",a=-1,b=1,rotate=2,grid=5):
```




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

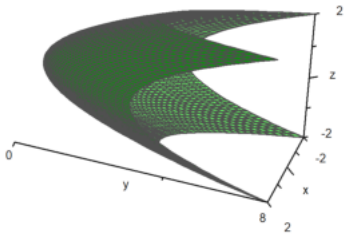



```
>plot3d("x*sin(x)",a=0,b=6pi,rotate=2):
```



Ini adalah fungsi dengan 3 variabel.

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



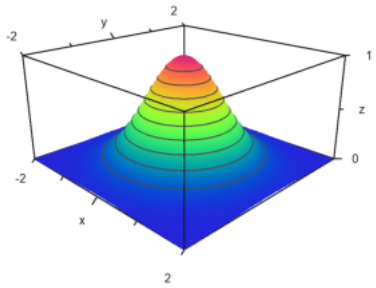
Plot Pola Kontur

Untuk plot, Euler menambahkan garis grid. Sebagai gantinya, bisa menggunakan garis level dan satu warna atau spektrum berwarna. Euler bisa menggambar ketinggian fungsi pada sebuah plot dengan bayangan. Dalam semua plot 3D, Euler bisa menghasilkan anaglyph merah/sian.

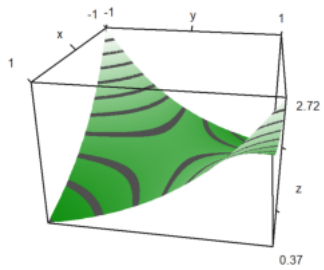
>hue: Mengaktifkan bayangan cahaya alih-alih kawat,
>contour: Memplot garis kontur otomatis pada plot,
level=... (atau levels): Sebuah vektor nilai untuk garis kontur.

Default-nya adalah level="auto", yang secara otomatis menghitung beberapa garis level. Seperti yang kamu lihat dalam plot, level-level ini sebenarnya adalah rentang level. Gaya default bisa diubah. Untuk plot kontur berikut, kita menggunakan grid yang lebih halus yaitu 100x100 titik, menskalakan fungsi dan plot, serta 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):
```

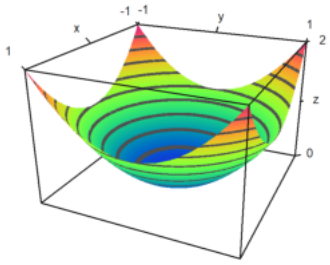


Bayangan default menggunakan warna abu-abu, tetapi skema warna spektral juga tersedia.

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

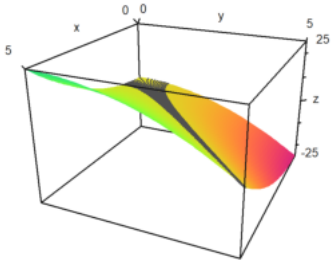
Untuk plot berikut, kita menggunakan skema spektral default dan meningkatkan jumlah titik untuk mendapatkan tampilan yang sangat halus.

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



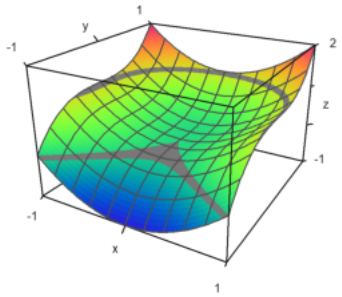
Alih-alih garis level otomatis, kita juga bisa menetapkan nilai-nilai garis level. Ini akan menghasilkan garis level tipis alih-alih rentang level.

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



Dalam plot berikut, kita 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. Kita juga menambahkan 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):
```

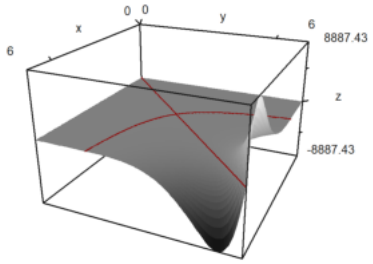


Dalam plot berikut, kita memplot set di mana:

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

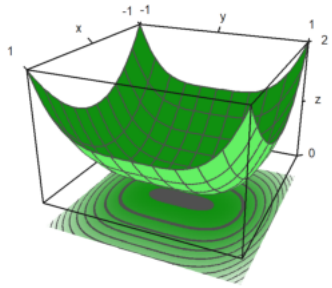
Kita menggunakan satu garis tipis untuk garis level.

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



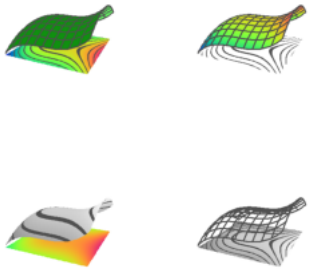
Dimungkinkan untuk menampilkan bidang kontur di bawah plot. Warna dan jarak dari plot dapat ditentukan.

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



Berikut adalah beberapa gaya lainnya. Kita selalu mematikan bingkai, dan menggunakan berbagai skema warna untuk plot dan grid.

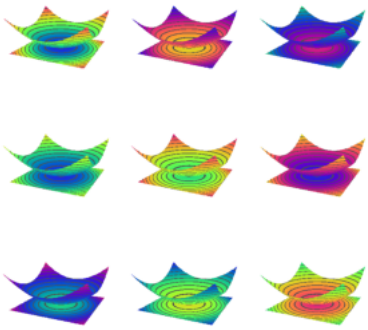
```
>figure(2,2); ...
 expr="y^3-x^2"; ...
 figure(1);  ...
   plot3d(expr,<frame,>cp,cpcolor=spectral); ...
 figure(2);  ...
   plot3d(expr,<frame,>spectral,grid=10,cp=2); ...
 figure(3);  ...
   plot3d(expr,<frame,>contour,color=gray,nc=5,cp=3,cpcolor=greenred); ...
 figure(4);  ...
   plot3d(expr,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



Ada beberapa skema spektral lainnya, bernomor dari 1 hingga 9. Tetapi kamu juga dapat menggunakan color=value, di mana value adalah:

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

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

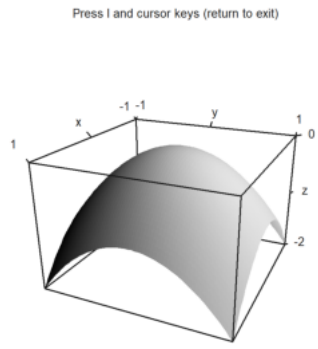


Sumber cahaya dapat diubah dengan menekan l dan menggunakan tombol panah selama interaksi pengguna. Parameter sumber cahaya dapat diatur dengan:

- light: arah cahaya,
- amb: cahaya ambient antara 0 dan 1.

Perhatikan bahwa program ini tidak membedakan antara sisi-sisi plot. Tidak ada bayangan. Untuk ini, kamu 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)"):
```

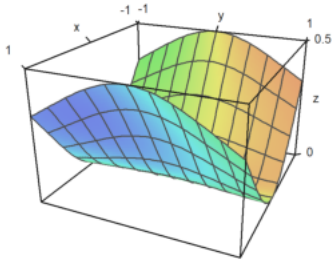


Parameter color mengubah warna permukaan. Warna garis level juga dapat diubah.

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

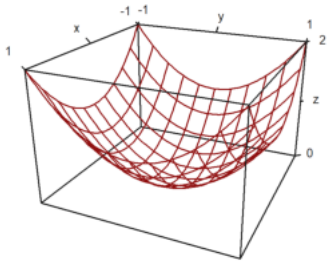
Warna 0 memberikan efek pelangi khusus.

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



Permukaan juga bisa transparan.

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



Plot Implisit

Ada juga plot implisit dalam tiga dimensi. Euler menghasilkan potongan melalui objek. Fitur plot3d termasuk plot implisit. Plot ini menunjukkan set nol dari sebuah fungsi dalam tiga variabel.

Solusi dari:

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

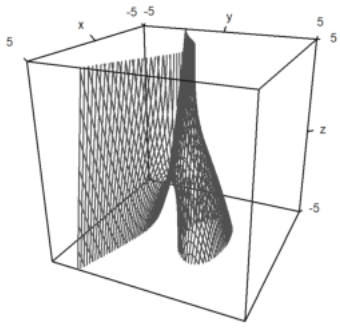
dapat divisualisasikan dalam potongan-potongan yang sejajar dengan bidang x-y, x-z, dan y-z.

- implicit=1: potongan sejajar dengan bidang y-z,
- implicit=2: potongan sejajar dengan bidang x-z,
- implicit=4: potongan sejajar dengan bidang x-y.

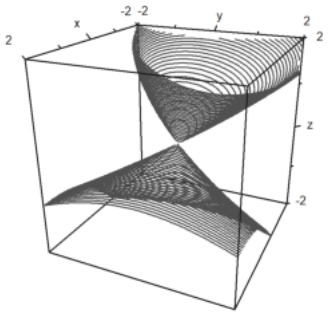
Tambahkan nilai-nilai ini jika kamu suka. Dalam contoh berikut, kita memplot:

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

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



Plot Data 3D

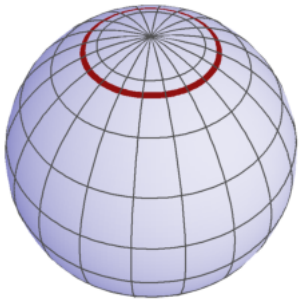
Sama seperti plot2d, plot3d menerima data. Untuk objek 3D, kamu perlu menyediakan matriks nilai x, y, dan z, atau tiga fungsi atau ekspresi fx(x,y), fy(x,y), fz(x,y).

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

Karena x, y, z adalah matriks, kita mengasumsikan bahwa (t, s) berjalan melalui grid bujur sangkar. Hasilnya, kamu bisa memplot gambar persegi panjang di ruang angkasa. Kamu bisa menggunakan bahasa matriks Euler untuk menghasilkan koordinat secara efektif.

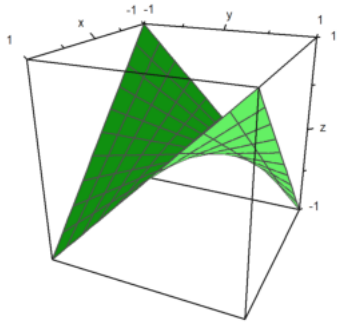
Dalam contoh berikut, kita menggunakan vektor nilai t dan vektor kolom nilai s untuk memparametrikan permukaan bola. Dalam gambar tersebut, kita dapat menandai wilayah, dalam kasus ini wilayah kutub.

```
>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 adalah contoh, yang merupakan grafik dari sebuah fungsi.

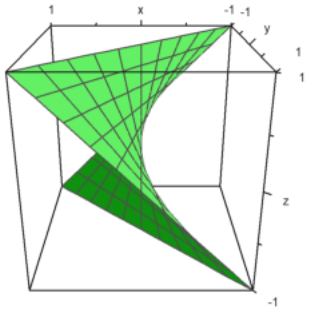
```
>t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,t*s,grid=10):
```



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

$$x = y\,z$$

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



Dengan lebih banyak usaha, kita bisa menghasilkan banyak permukaan.

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

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

dengan:

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

Kita mengubahnya dengan faktor:

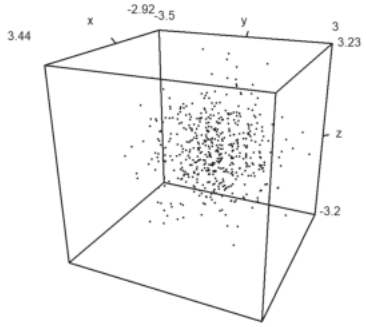
$$d(t,s) = \frac{\cos(4t)+\cos(8s)}{4}.$$

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

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

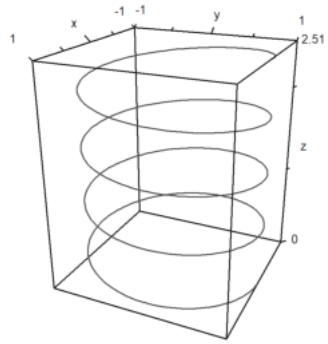
Gaya visualisasi adalah sama seperti di plot2d dengan points=true.

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

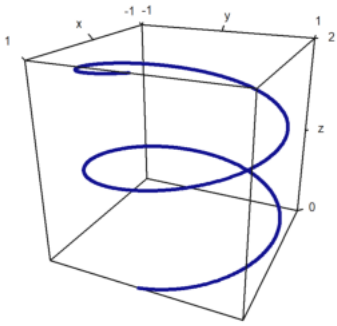


Dimungkinkan juga untuk memplot kurva dalam 3D. Dalam kasus ini, lebih mudah untuk menghitung titik-titik kurva sebelumnya. Untuk kurva dalam bidang, kita menggunakan urutan koordinat dan parameter wire=true.

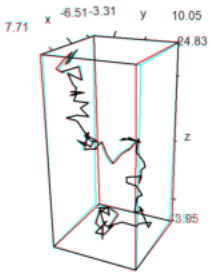
```
>t=linspace(0,8pi,500); ...
 plot3d(sin(t),cos(t),t/10,>wire,zoom=3):
```




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

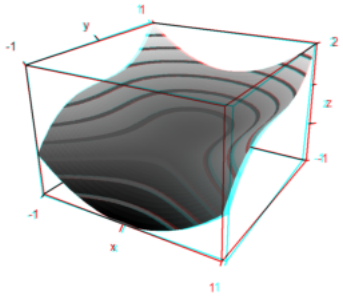



```
>X=cumsum(normal(3,100)); ...
  plot3d(X[1],X[2],X[3],>anaglyph,>wire):
```



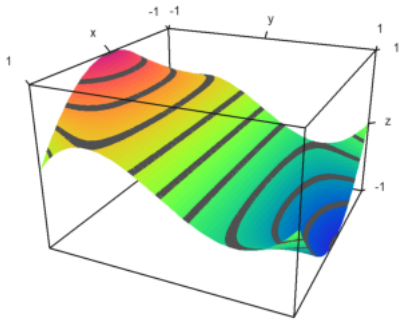
EMT juga dapat membuat plot dalam mode anaglyph. Untuk melihat plot seperti itu, Anda memerlukan kacamata merah/sian.

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



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

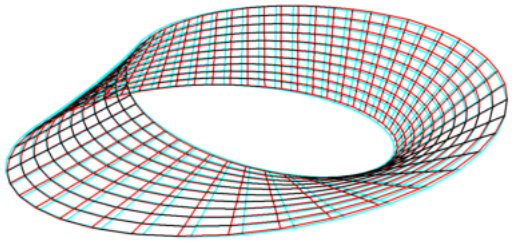
```
>plot3d("x^2*y^3-y",>spectral,>contour,zoom=3.2):
```



Euler juga dapat membuat plot permukaan parameterisasi, di mana parameter-parameter tersebut adalah nilai x, y, dan z dari gambar grid persegi panjang di ruang.

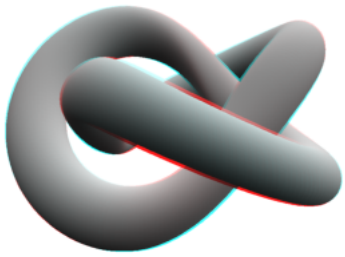
Untuk demo berikut, kita menetapkan parameter u dan v, dan menghasilkan koordinat ruang dari parameter-parameter ini.

```
>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 terlihat 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

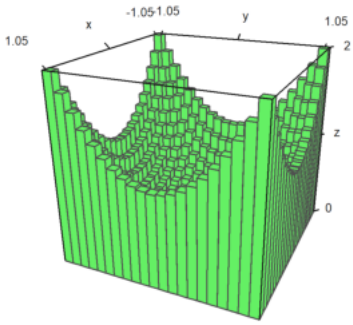
Plot batang juga dimungkinkan. Untuk ini, kita harus menyediakan:

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

z bisa lebih besar, tetapi hanya nilai nxn yang akan digunakan. Dalam contoh ini, kita pertama-tama menghitung nilai-nilainya.

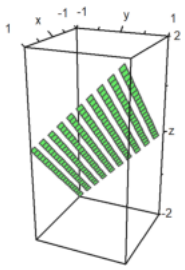
Lalu kita menyesuaikan x dan y, sehingga vektor-vektor tersebut terpusat pada nilai yang digunakan.

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



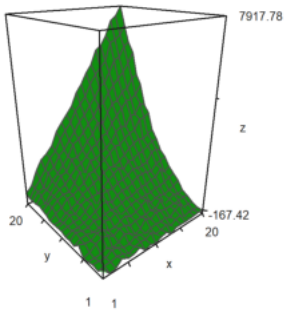
Dimungkinkan untuk membagi plot permukaan menjadi dua atau lebih bagian.

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

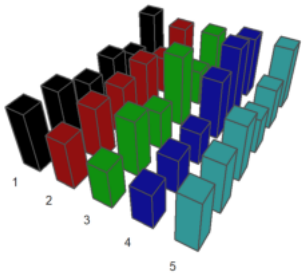


Jika kamu memuat atau menghasilkan matriks data M dari file dan perlu memplotnya dalam 3D, kamu bisa menyesuaikan matriks tersebut menjadi [-1,1] dengan scale(M), atau menyesuaikan matriks dengan >zscale. Ini bisa dikombinasikan dengan faktor skala individu yang diterapkan secara tambahan.

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

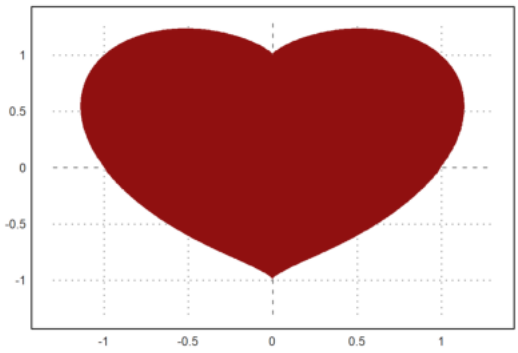



```
>Z=intrandom(5,100,6); v=zeros(5,6); ...
 loop 1 to 5; v[#]=getmultiplicities(1:6,Z[#]); end; ...
 columnsplot3d(v',scols=1:5,ccols=[1:5]):
```



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
```

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

Ekspresi ini menghasilkan bentuk hati tiga dimensi, yang kemudian kita putar di sekitar sumbu y.
Ini adalah ekspresi yang menggambarkan hati:

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

Selanjutnya kita set:

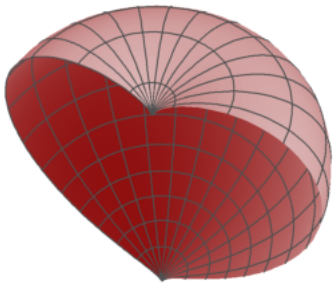
$$x=r.cos(a),\quad y=r.sin(a).$$

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

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

Ini memungkinkan kita untuk mendefinisikan fungsi numerik yang menyelesaikan r, jika a diberikan.
Dengan fungsi ini, kita dapat memplot bentuk hati yang diputar sebagai permukaan parametrik.

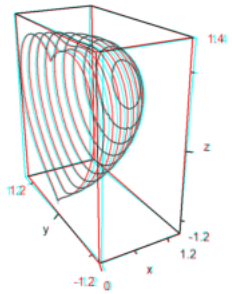
```
>function map f(a) := bisect("fr",0,2;a); ...
 t=linspace(-pi/2,pi/2,100); r=f(t);  ...
 s=linspace(pi,2pi,100)'; ...
 plot3d(r*cos(t)*sin(s),r*cos(t)*cos(s),r*sin(t), ...
 >hue,<frame,color=red,zoom=4,amb=0,max=0.7,grid=12,height=50°):
```



Berikut adalah plot 3D dari gambar di atas yang diputar di sekitar sumbu z. Kita mendefinisikan fungsi yang menggambarkan objek tersebut.

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

```
>plot3d("f(x,y,z)", ...
 xmin=0,xmax=1.2,ymin=-1.2,ymax=1.2,zmin=-1.2,zmax=1.4, ...
 implicit=1,angle=-30°,zoom=2.5,n=[10,100,60],>anaglyph):
```



Plot 3D Khusus

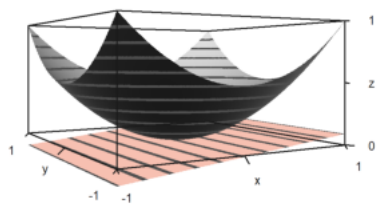
Fungsi plot3d memang berguna, tetapi tidak memuaskan semua kebutuhan. Selain dari rutinitas dasar, dimungkinkan untuk mendapatkan plot yang dibingkai dari objek apa pun yang kamu inginkan.

Meskipun Euler bukan program 3D, program ini dapat menggabungkan beberapa objek dasar. Kita 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-bingkai, dan mengatur pandangan.

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

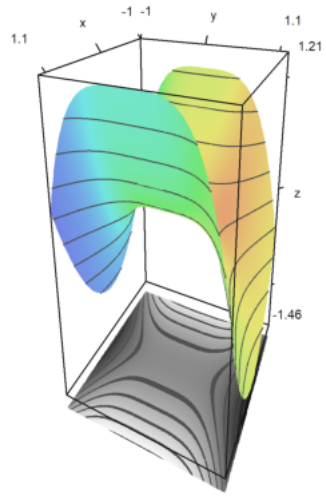


Dengan cara yang sama, kamu dapat memplot bidang kontur secara manual. Perhatikan bahwa plot3d() mengatur jendela menjadi fullwindow() secara default, tetapi plotcontourplane() mengasumsikan itu.

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

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

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



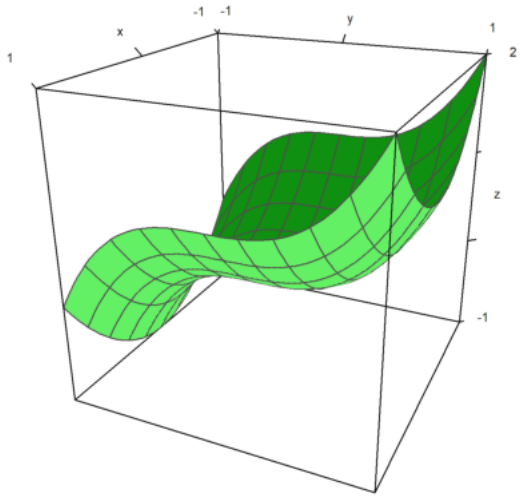
Animasi

Euler dapat menggunakan bingkai untuk menghitung animasi terlebih dahulu.

Salah satu fungsi yang menggunakan teknik ini adalah rotate. Fungsi ini dapat mengubah sudut pandang dan menggambar ulang plot 3D. Fungsi ini memanggil addpage() untuk setiap plot baru. Akhirnya, animasi dari plot tersebut ditampilkan.

Silakan pelajari sumber dari rotate untuk melihat lebih detail.

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



Menggambar Povray

Dengan bantuan file Euler povray.e, Euler dapat menghasilkan file Povray. Hasilnya sangat bagus untuk dilihat. Kamu perlu menginstal Povray (32bit atau 64bit) dari http://www.povray.org, dan menambahkan sub-direktori "bin" dari Povray ke jalur lingkungan, atau mengatur variabel defaultpovray dengan jalur penuh yang menunjuk ke "pvengine.exe".

Antarmuka Povray dari Euler menghasilkan file-file Povray di direktori rumah pengguna, dan memanggil Povray untuk mengurai file-file tersebut. 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 serupa dengan plot3d. Fungsi ini dapat menghasilkan grafik sebuah fungsi f(x,y), atau permukaan dengan koordinat X, Y, Z dalam matriks, termasuk garis level opsional. Fungsi ini secara otomatis memulai raytracer, dan memuat pemandangan ke dalam notebook Euler.

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

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

erlu diperhatikan bahwa alam semesta Povray memiliki sistem koordinat yang berbeda. Antarmuka ini menerjemahkan semua koordinat ke sistem Povray. Jadi kamu bisa terus berpikir dalam sistem koordinat Euler dengan z yang mengarah vertikal ke atas, serta sumbu x, y, z mengikuti aturan tangan kanan. Kamu perlu memuat file povray.

```
>load povray;
```

Pastikan, direktori bin Povray ada di jalur. Jika tidak, edit variabel berikut sehingga berisi jalur ke executable Povray.

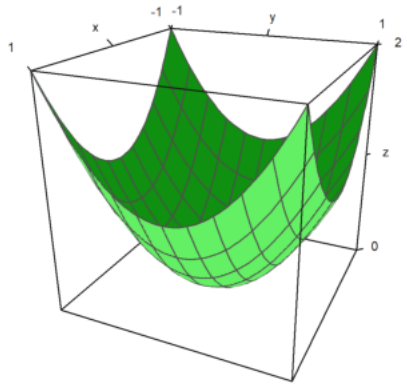
```
>defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

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

Untuk kesan pertama, kita memplot sebuah fungsi sederhana. Perintah berikut menghasilkan sebuah file povray di direktori pengguna, dan menjalankan Povray untuk melakukan pelacakan sinar pada file ini.

Jika kamu memulai perintah berikut, GUI Povray akan terbuka, menjalankan file tersebut, dan menutup secara otomatis. Karena alasan keamanan, kamu akan diminta apakah ingin mengizinkan file exe untuk dijalankan. Kamu bisa menekan cancel untuk menghentikan pertanyaan lebih lanjut. Mungkin kamu harus menekan OK di jendela Povray untuk mengakui dialog start-up Povray.

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




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

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

Kadang-kadang, perlu untuk mencegah penskalaan fungsi, dan menskalakannya secara manual.

Kita memplot set titik-titik dalam bidang kompleks, di mana 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);
```

Plot dengan Koordinat

Alih-alih fungsi, kita bisa memplot dengan koordinat. Seperti pada plot3d, kita memerlukan tiga matriks untuk mendefinisikan objek. Dalam contoh ini, kita memutar sebuah fungsi di sekitar sumbu z.

```
>load povray;
>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]):
```

```
Command was not allowed!
exec:
    return _exec(program,param,dir,print,hidden,wait);
```

```
povray:
    exec(program,params,defaulthome);
Try "trace errors" to inspect local variables after errors.
pov3d:
    if povray then povray(currentfile,w,h,w/h); endif;
```

Dalam contoh berikut, kita memplot gelombang teredam. Kita menghasilkan gelombang dengan bahasa matriks Euler. Kita juga menunjukkan bagaimana sebuah objek tambahan bisa ditambahkan ke pemandangan pov3d. Untuk menghasilkan objek, lihat contoh-contoh berikut ini. Perhatikan bahwa plot3d menskalakan plot, sehingga sesuai dengan kubus unit.

```
>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);
```

```
Function povlook not found.
Try list ... to find functions!
Error in:
... =30°,add=povsphere([0.5,0,0.25],0.15,povlook(red)),   w=500,h= ...
                                                          ^
```

Dengan metode shading canggih dari Povray, sangat sedikit titik yang bisa menghasilkan permukaan yang sangat halus. Hanya di batas dan bayangan trik ini mungkin menjadi jelas. Untuk ini, kita perlu menambahkan vektor normal di setiap titik matriks.

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

$$x^2 y^3$$

Persamaan permukaan adalah [x,y,Z]. Kita menghitung dua turunan terhadap x dan y dari ini dan mengambil hasil silang sebagai normal.

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

Kita mendefinisikan normal sebagai hasil silang dari kedua turunan tersebut, dan mendefinisikan fungsi koordinat.

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

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

Kita hanya menggunakan 25 titik.

```
>x=-1:0.5:1; y=x'
```

```
  -1
-0.5
   0
 0.5
   1
```

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

```
Function pov3d not found.
Try list ... to find functions!
Error in:
... =10°,    xv=NX(x,y),yv=NY(x,y),zv=NZ(x,y),<shadow): ...
                                                        ^
```

Persamaan simpul Trefoil dikerjakan oleh A. Busser dalam Povray. Ada versi yang lebih baik dari ini di contoh-contoh. Simak di:

Trefoil Knot

Untuk hasil yang baik dengan tidak terlalu banyak titik, kita menambahkan vektor normal di sini. Kita menggunakan Maxima untuk menghitung normal bagi kita. Pertama, tiga fungsi untuk koordinat sebagai ekspresi simbolik:

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

Kemudian vektor turunan terhadap x dan y.

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

Sekarang normal, yang merupakan hasil silang dari kedua turunan tersebut.

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

Sekarang kita menghitung semuanya secara numerik.

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

Vektor normal adalah hasil evaluasi dari ekspresi simbolik dn[i] untuk i=1,2,3. Sintaks untuk ini adalah &"expression"(parameters). Ini merupakan alternatif dari metode di 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));
```

```
Function povlook not found.
Try list ... to find functions!
Error in:
... ,zoom=5,w=450,h=350,   <shadow,look=povlook(blue),   xv=&"dn[1 ...
                                                    ^
```

Kita juga bisa 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();
```

```
Function povstart not found.
Try list ... to find functions!
Error in:
povstart(zoom=4); x=-1:0.5:1; r=1-(x+1)^2/6; t=(0°:30°:360°)'; ...
                 ^
```

Dengan povgrid(), kurva-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();
```

```
Function povstart not found.
Try list ... to find functions!
Error in:
povstart(center=[0,0,1],zoom=3.6); t=linspace(0,2,1000); r=exp ...
                                  ^
```

Objek Povray

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

```
>load povray;
>//defaultpovray="pvengine.exe"list
>povstart(zoom=4):
```

Pertama kita mendefinisikan tiga silinder, dan menyimpannya sebagai string di Euler. Fungsi povx() dan seterusnya, hanya mengembalikan vektor [1,0,0], yang juga bisa digunakan.

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

String-string tersebut berisi kode Povray, yang tidak perlu dipahami pada titik ini.

```
>c1
```

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

Seperti yang kamu lihat, kita menambahkan tekstur pada objek dengan tiga warna berbeda. Ini dilakukan oleh povlook(), yang mengembalikan string dengan kode Povray yang relevan. Kita bisa menggunakan warna Euler default, atau mendefinisikan warna sendiri. Kita juga bisa menambahkan transparansi, atau mengubah cahaya ambient.

```
>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 dari tiga silinder sulit divisualisasikan jika kamu belum pernah melihatnya sebelumnya.

```
>povend();
```

```
 Command was not allowed!
 exec:
     return _exec(program,param,dir,print,hidden,wait);
 povray:
     exec(program,params,defaulthome);
 Try "trace errors" to inspect local variables after errors.
 povend:
     povray(file,w,h,aspect,exit);
```

Fungsi-fungsi berikut menghasilkan fraktal secara rekursif. Fungsi pertama menunjukkan bagaimana Euler menangani objek Povray sederhana. Fungsi povbox() mengembalikan string, 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();
```

```
 Command was not allowed!
 exec:
     return _exec(program,param,dir,print,hidden,wait);
 povray:
     exec(program,params,defaulthome);
 Try "trace errors" to inspect local variables after errors.
 povend:
     povray(file,w,h,aspect,exit);
```

Objek bisa dikurangi dengan menggunakan perbedaan. Seperti persimpangan, ini adalah bagian dari objek CSG dari Povray.

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

Untuk demonstrasi ini, kita mendefinisikan objek di Povray, bukan menggunakan string di Euler. Definisi ditulis langsung ke file.

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

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

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

Kita menghasilkan kubus kedua, memutarnya sedikit, dan menskalakannya.

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

Kemudian kita mengambil perbedaan dari kedua objek.

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

Sekarang kita 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();
```

```
    union {
      cylinder { <-1.3,0,0>,<1.3,0,0>,0.02 }
      cone {
        <1.42,0,0>,0
        <1.3,0,0>,0.08
      }
      texture { pigment { color rgb <0.470588,0.470588,0.470588> } }
    }
    union {
      cylinder { <-1.3,0,0>,<1.3,0,0>,0.02 }
      cone {
        <1.42,0,0>,0
        <1.3,0,0>,0.08
      }
      rotate 90*z
      texture { pigment { color rgb <0.470588,0.470588,0.470588> } }
    }
    union {
      cylinder { <-1.3,0,0>,<1.3,0,0>,0.02 }
      cone {
        <1.42,0,0>,0
        <1.3,0,0>,0.08
      }
      rotate -90*y
      texture { pigment { color rgb <0.470588,0.470588,0.470588> } }
    }
    Command was not allowed!
    exec:
        return _exec(program,param,dir,print,hidden,wait);
    povray:
        exec(program,params,defaulthome);
    Try "trace errors" to inspect local variables after errors.
    povend:
        povray(file,w,h,aspect,exit);
```

Fungsi Implisit

Povray bisa memplot set di mana f(x,y,z)=0, sama seperti parameter implisit di plot3d. Hasilnya tampak jauh lebih baik, bagaimanapun. Sintaks untuk fungsi ini sedikit berbeda. Kamu tidak bisa menggunakan output dari 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)+
 writeAxes();...
 povend(exit);
```

```
    Variable or function exit not found.
    Error in:
    povend(exit); ...
              ^
```

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

```
    Command was not allowed!
    exec:
        return _exec(program,param,dir,print,hidden,wait);
    povray:
        exec(program,params,defaulthome);
    Try "trace errors" to inspect local variables after errors.
    povend:
        povray(file,w,h,aspect,exit);
```

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

Create the implicit surface. Note the different syntax in the expression.

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

```
object {
isosurface {
function { pow(x,2)*y-pow(y,3)-pow(z,2) }
max_gradient 5
open
contained_by { box { <-1,-1,-1>, <1,1,1>
 } }
 texture { pigment { color rgb <0.0627451,0.564706,0.0627451> }  }
 finish { ambient 0.2 }
}}
union {
  cylinder { <-1.1,0,0>,<1.1,0,0>,0.02 }
  cone {
    <1.22,0,0>,0
    <1.1,0,0>,0.08
  }
  texture { pigment { color rgb <0.470588,0.470588,0.470588> } }
}
union {
  cylinder { <-1.1,0,0>,<1.1,0,0>,0.02 }
  cone {
    <1.22,0,0>,0
    <1.1,0,0>,0.08
  }
  rotate 90*z
  texture { pigment { color rgb <0.470588,0.470588,0.470588> } }
}
union {
  cylinder { <-1.1,0,0>,<1.1,0,0>,0.02 }
  cone {
    <1.22,0,0>,0
    <1.1,0,0>,0.08
  }
  rotate -90*y
  texture { pigment { color rgb <0.470588,0.470588,0.470588> } }
}
Variable or function exit not found.
Error in:
... (z,2)",povlook(green))); writeAxes(); povend(exit); ...
                                                    ^
```

Objek Mesh

Dalam contoh ini, kita menunjukkan bagaimana cara membuat objek mesh, dan menggambarnya dengan informasi tambahan. Kita ingin memaksimalkan xy di bawah kondisi x+y=1 dan menunjukkan sentuhan tangensial dari garis level.

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

Kita tidak dapat menyimpan objek ini sebagai string seperti sebelumnya, karena terlalu besar. Jadi kita mendefinisikan objek di file Povray menggunakan declare. Fungsi povtriangle() melakukan ini secara otomatis. Fungsi ini bisa menerima vektor normal seperti pov3d(). Berikut definisi objek mesh, dan langsung menulisnya ke 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);
```

Kita sekarang mendefinisikan dua cakram yang akan dipotong dengan permukaan.

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

Tulis permukaan minus dua cakram ini.

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

Tulis dua persimpangan ini.

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

Tambahkan sebuah titik di maksimum.

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

Tambahkan sumbu dan selesaikan.

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

```
union {
  cylinder { <-0.1,0,0>,<1.1,0,0>,0.015 }
  cone {
    <1.19,0,0>,0
    <1.1,0,0>,0.06
  }
  texture { pigment { color rgb <0.470588,0.470588,0.470588> } }
}
union {
  cylinder { <-0.1,0,0>,<1.1,0,0>,0.015 }
  cone {
    <1.19,0,0>,0
    <1.1,0,0>,0.06
  }
```

```
    rotate 90*z
    texture { pigment { color rgb <0.470588,0.470588,0.470588> } }
  }
  union {
    cylinder { <-0.1,0,0>,<1.1,0,0>,0.015 }
    cone {
      <1.19,0,0>,0
      <1.1,0,0>,0.06
    }
    rotate -90*y
    texture { pigment { color rgb <0.470588,0.470588,0.470588> } }
  }
  Variable or function exit not found.
  Error in:
  writeAxes(0,1,0,1,0,1,d=0.015); povend(exit); ...
                                                ^
```

Anaglyphs di Povray

Untuk menghasilkan anaglyph untuk dilihat dengan kacamata merah/sian, 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, kamu memerlukan kacamata merah/sian untuk melihat contoh berikut dengan benar. Fungsi pov3d() memiliki sakelar sederhana untuk menghasilkan anaglyph.

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

```
  Command was not allowed!
  exec:
      return _exec(program,param,dir,print,hidden,wait);
  povray:
      exec(program,params,defaulthome);
  Try "trace errors" to inspect local variables after errors.
  pov3d:
      if povray then povray(currentfile,w,h,w/h); endif;
```

Jika kamu membuat sebuah pemandangan dengan objek-objek, kamu perlu menempatkan pembuatan pemandangan tersebut ke dalam fungsi, dan menjalankannya dua kali dengan nilai yang berbeda untuk parameter anaglyph.

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

Fungsi povanaglyph() melakukan semua ini secara otomatis. Parameter-parameternya mirip dengan gabungan antara povstart() dan povend()

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

```
  Command was not allowed!
  exec:
      return _exec(program,param,dir,print,hidden,wait);
  povray:
      exec(program,params,defaulthome);
  Try "trace errors" to inspect local variables after errors.
  povanaglyph:
      povray(currentfile,w,h,aspect,exit);
```

Mendefinisikan Objek Sendiri

Antarmuka Povray di Euler berisi banyak objek. Namun kamu tidak terbatas pada objek-objek ini. Kamu dapat membuat objek sendiri, yang menggabungkan objek-objek lain, atau objek baru sepenuhnya. Berikut adalah contoh pembuatan torus. Perintah Povray untuk ini adalah "torus". Jadi kita mengembalikan string dengan perintah ini dan parameternya. Perhatikan bahwa torus selalu berada di tengah titik asal.

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

Berikut adalah torus pertama kita.

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

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

Sekarang kita menggunakan 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 ini ke dalam sebuah pemandangan. Untuk tampilannya, kita 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();
```

Povray memanggil program tersebut. Namun, jika terjadi kesalahan, program tidak akan menampilkan error-nya. Kamu harus menggunakan perintah berikut:

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

Jika ada yang tidak berfungsi. Ini akan membuat jendela Povray tetap terbuka.

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

```
Function povstart not found.
Try list ... to find functions!
Error in:
... rt(center=[0.4,0,0],angle=0°,zoom=3.8,aspect=1.5); writeln(pov ...
                                                        ^
```

Berikut adalah contoh yang lebih rumit. Kita menyelesaikan persamaan:

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

dan menunjukkan titik-titik yang layak serta optimum 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, kita cek apakah contoh ini memiliki solusi.

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

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

Ya, memiliki solusi. Selanjutnya kita mendefinisikan dua objek. Yang pertama adalah bidang:

$$a \cdot x \le b$$

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

Kemudian kita mendefinisikan persimpangan semua setengah ruang dan sebuah kubus. Berikut adalah kode di dalam fungsi:

```
>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
```

```
>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); ...
```

Sekarang kita bisa memplot pemandangan.

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

```
Function povstart not found.
Try list ... to find functions!
Error in:
... ovstart(angle=120°,center=[0.5,0.5,0.5],zoom=3.5); writeln(adm ...
                                                        ^
```

Dan sebuah panah ke arah optimum.

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

```
Function povlook not found.
Try list ... to find functions!
Error in:
writeln(povarrow(x,c*0.5,povlook(red))); ...
                                  ^
```

Kita tambahkan teks ke layar. Teks adalah 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();
```

```
Function povtext not found.
Try list ... to find functions!
Error in:
... "Linear Problem",[0,0.2,1.3],size=0.05,rotate=5°)); povend(); ...
                                                    ^
```

More Examples

You can find some more examples for Povray in Euler in the following files.

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

Soal-soal

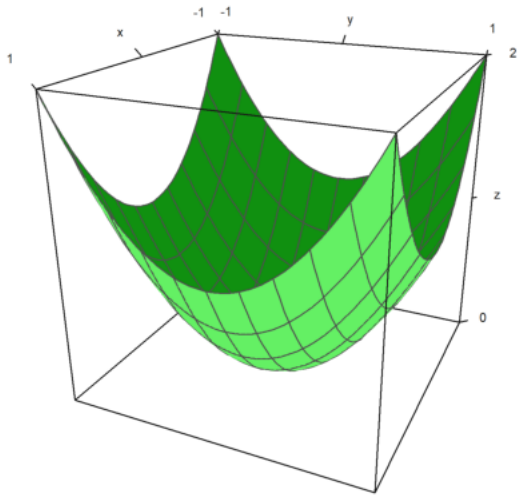
Nama : Muhammad Lutfi Ramadhan
Kelas : Matematika B 2023
NIM : 23030630021

1. Gambarkan permukaan paraboloid yang diberikan oleh fungsi berikut.

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

Penyelesaian:

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

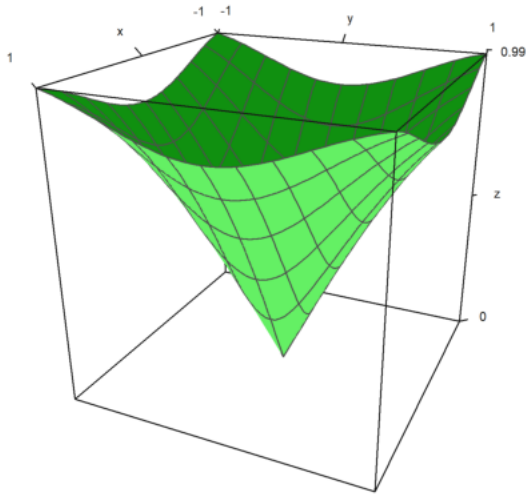


2. Gambarkan kurva heliks yang didefinisikan oleh parameter berikut

$$z = sin\sqrt{x^2 + y^2}$$

Penyelesaian:

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

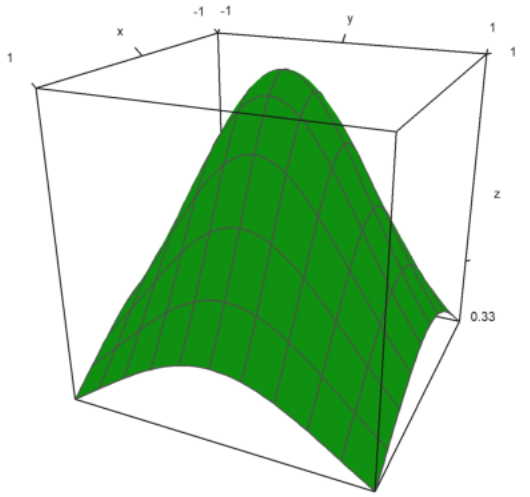


3. Gambarkan permukaan hyperboloid yang didefinisikan oleh fungsi.

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

Penyelesaian:

```
>plot3d("1/(x^2 + y^2 + 1)"):
```

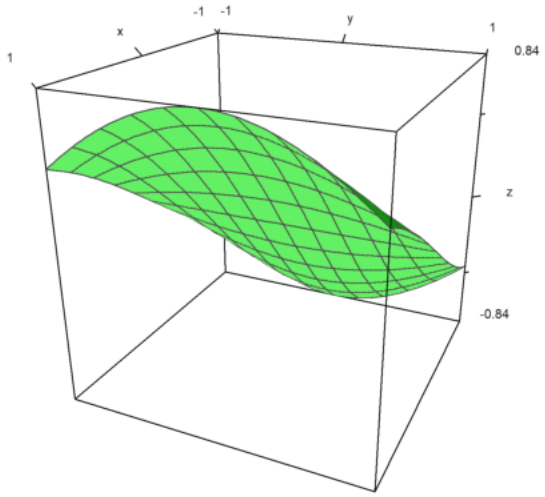


4. Gambarkan permukaan yang didefinisikan oleh fungsi berikut.

$$z = sin(x).cos(y)$$

Penyelesaian:

```
>plot3d("sin(x) * cos(y)"):
```



5. Gambarkan permukaan paraboloid terbalik yang didefinisikan oleh fungsi berikut.

$$z = -x^2 - y^2$$

Penyelesaian:

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