Nama : Robi'atul Adawiyah
NIM : 22301244039
Kelas: Pendidikan Matematika C 2022

Menggambar Plot 3D dengan EMT

Notebook ini berisi perkenalan plot 3D di dalam Euler. Kita butuh sebuah plot 3D untuk menvisualisasikan sebuah fungsi dua variabel.

Euler menggambar setiap fungsi menguggunakan sebuah Algoritma Pengurutan untuk menyembunyikan bagian-bagian yang lain di dala, background. Umumnya, Euler menggunakan proyeksi pusat. Grafik bawaan nya dari kuadran positif x-y terhadap titik asal x=y=z=0, namun pada sudut 0 derajat terlihat dari arah sumbu y. Pemandangan sudut dan tinggi dapat diubah.

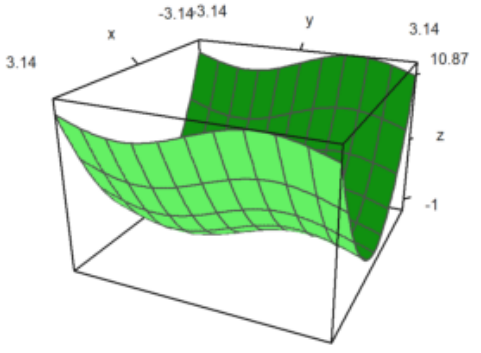
Euler dapat plot

- permukaan dengan arsiran dan level garis atau level range,

```
- kumpulan titik-titik,
- kurva parametrik,
- permukaan implisit.
```

Sebuah plot 3D untuk fungsi menggunakan "plot3d". Langkah terbaik dengan plot sebuah ekspresi di dalam x dan y. Parameter r mengatur range plot sekitar (0,0).

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


-3.14
3.14
x
3.14
y
3.14
10.87
z
-1

Fungsi Dua Variabel

Untuk grafik fungsi, gunakan

- ekspresi sederhana di x dan y,

```
- nama fungsi dua variabel
- atau data matriks
```

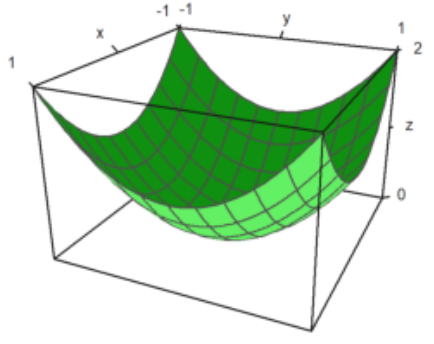
Bawaan EMT grid yang diarsir dengan warna yang berbeda pada kedua sisinya. Perhatikan bahwa banyaknya inteval grid bawaan adalah 10, namun plot menggunakan bawaan EMT persegi berukuran 40x40 yang membangun permukaan. Permukaan bawaan EMT dapat diubah.

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

```
- grid=10, grid=[10,10]: banyaknya garis grid di setiap arah
```

Kita menggunakan pengaturan bawaan EMT n=40 dan grid=10.

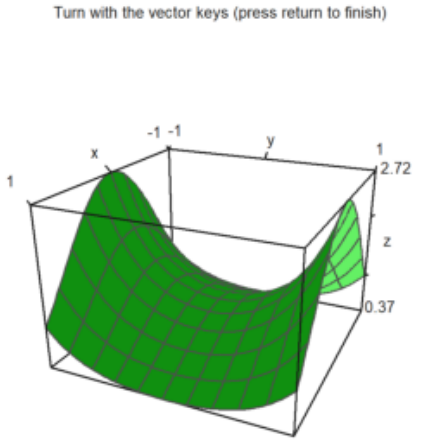
```
>plot3d("x^2+y^2"):
```



Interaksi Pengguna dapat menggunakan parameter >user. Pengguna dapat menekan key berikut.
-kiri, kanan, atas, bawah: melihat dari beberapa angle

```
- +,-: zoom in atau zoom out
- a: membuat "anaglyph" lihat di bawah nanti
- l: mengalihkan sumber cahaya, lihat di bawah nanti
- spasi: mengatur ulang ke pengaturan bawaan.
- enter: mengakhiri iterasi
```

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



Plot range untuk fungsi dapat dispesifikasikan dengan

- a,b: range x

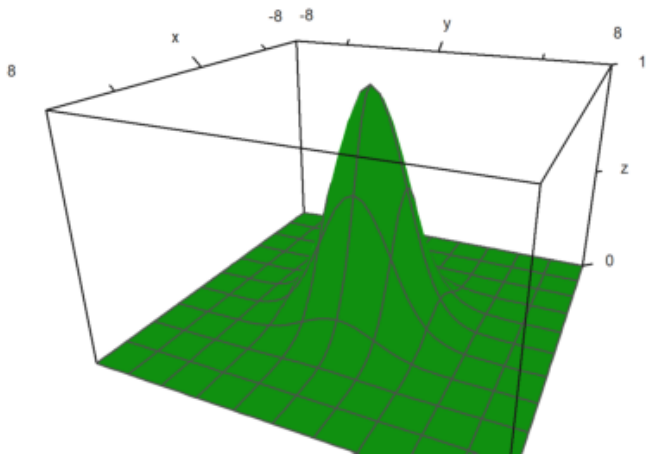
```
- c,d: range y
- r: persegi simetris sekitar (0,0)
- n: banyaknya sub-interval pada plot
```

Ada beberapa parameter untuk skala fungsi atau mengubah tampilan pada grafik.

fscale: skala pada nilai fungsi (bawaannya adalah <fscale)

```
scale: banyak atau vektor 1x2 ke skala pada direksi x dan y
frame: tipe frame (bawaannya 1)
```

```
>plot3d("exp(-(x^2+y^2)/5)",r=8,n=40,fscale=5,scale=1.5,frame=4):
```



Gambarannya dapat diubah dnegan banyak cara yang berbeda.

- distance: Jarak gambar ke plot

- zoom: nilai zoom.
- angle: angle pada sumbu y negatif dalam radian
- height: tinggi gambar dalam bentuk radian

Pengaturan bawaan nilai dapat diamati atau diubah dengan fungsi "view()". Fungsi ini dapat menampilkan parameter di dalam perintah di atas.

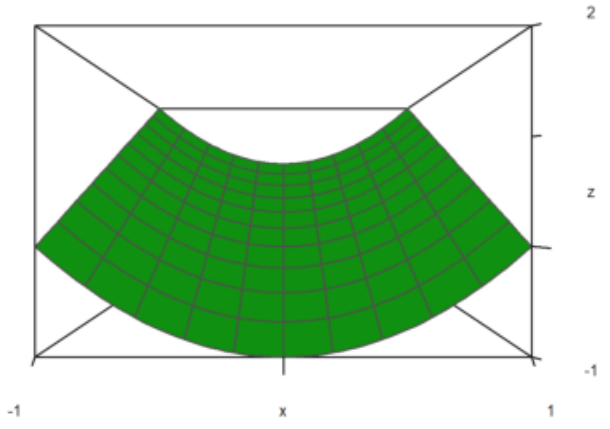
```
>view
```

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

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

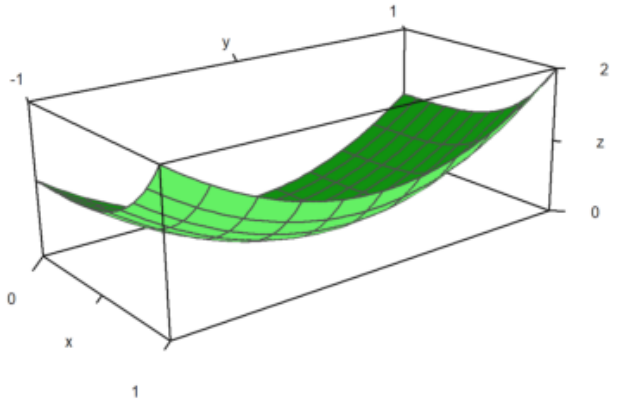
Dalam contoh berikut, angle=0 dan height=0 terlihat dari sumbu y negatif. Label sumbu untuk y tersembunyi dalam kasus ini.

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



Plotnya selalu terlihat pada titik tengah plot kubik. Kota dapat mengubahnya dengan parameter center.

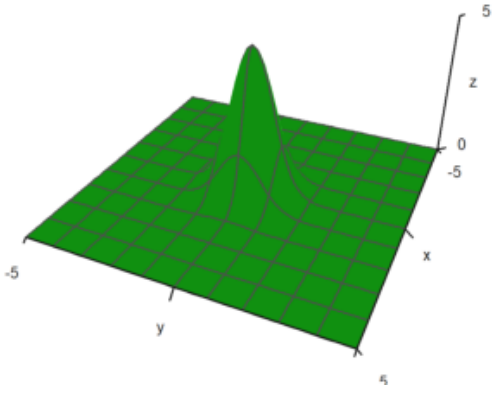
```
>plot3d("x^4+y^2",a=0,b=1,c=-1,d=1,angle=50°,height=20°, ...
>  center=[-0.4,0,0],zoom=5):
```



Plot diskalakan untuk disesuaikan ke unit kubik untuk penglihatan. Jadi, tidak pelu mengubah jarak atau zoom bergantung pada ukuran plot. Akan tetapi, labelnya merujuk kepada ukuran aslinya.

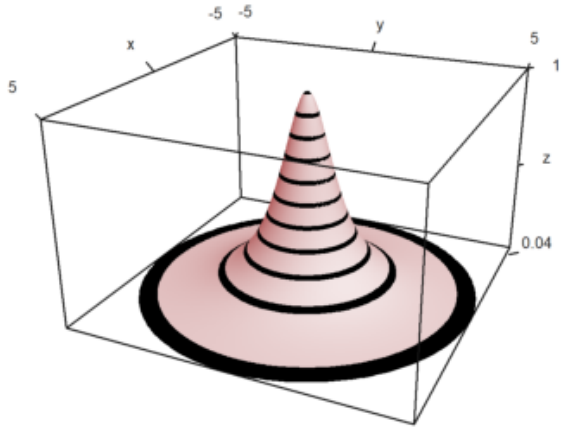
Jika kita memilih mematikan penyesuaian skala dengan "scale=off", kita perlu berhati-hati apakah plot masih sesuai denan jendela plot, dengan mengubah jarak pandangan atau zoom, dan memindahkan pusatnya.

```
>plot3d("5*exp(-x^2-y^2)",r=5,<fscale,<scale,distance=20,height=30°, ...
>  center=[0,0,-2],frame=3):
```

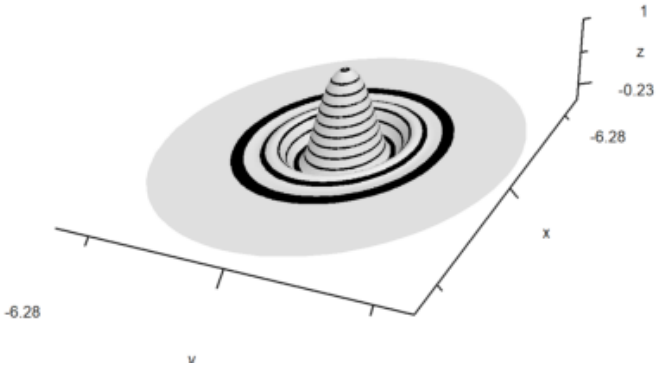


Polar plot juga ada. Prameter polar=true menggambar polar plot. Fungsi pasti harus tetap beruapa sebuah fungsi x dan y. Parameter "fscale" menyekalakan fungsi dengan sekala itu sendiri. Di sisi lain fungsi terskalakan untuk sesuai dengan plot kubik.

```
>plot3d("1/(x^2+y^2+1)",r=5, >polar, ...
>fscale=3,>hue,n=200,zoom=3.5, >contour, color=red):
```



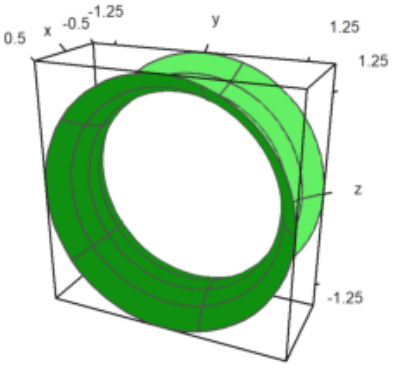

```
>function f(r) := exp(-r/2)*cos(r); ...
>plot3d("f(x^2+y^2)",n=100,>polar,scale=[1,0.7,0.4],r=2pi,frame=3,zoom=5,>hue,color=gray,fscale=3,>c
```



Parameter rotate merotasi fungsi x disekitaran sumbu c.
- rotate=1: menggunakan sumbu x

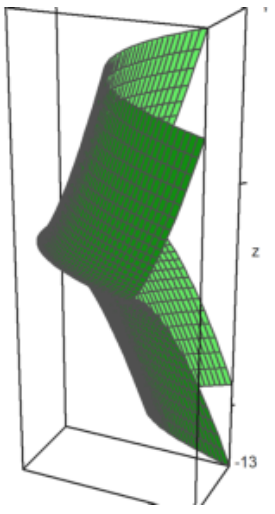
- rotate=2: menggunakan sumbu y

```
>plot3d("x^2+1",a=-0.5,b=0.5,rotate=true,grid=5):
```



Berikut plot dengan tiga fungsi.

```
>plot3d("x","x^2+y^2","x^3+y-3",r=2,zoom=5,frame=4):
```


z
-13

Plot kontur

Untuk Plot, Euler menambahkan grid garis. Disamping itu dapat juga menggunakan level garis dan corak one-color atau corak warna sepktral. Euler dapat menggambar tinggi fungsi pada plot dengan arsiran. Di dalam semua plot 3D Euler dapat menghasilkan anaglyph merah/biru muda.

- >hue: memperlihatkan arsiran

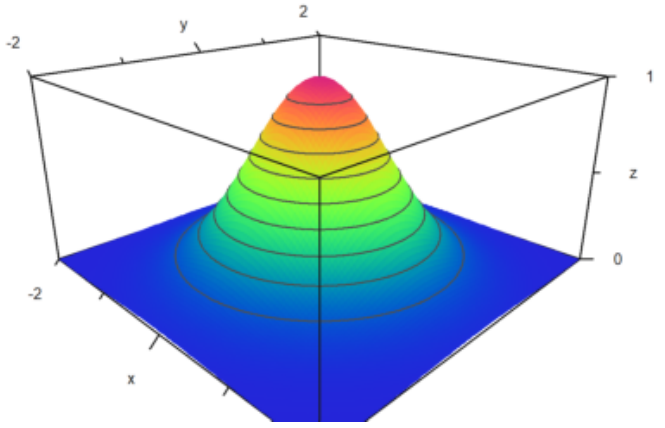
```
 - >contour: Plot garis kontur pada sebuah plot
 - >level: ... (atau levels): vektor yang berisi nilai untuk kontur
```

garis.

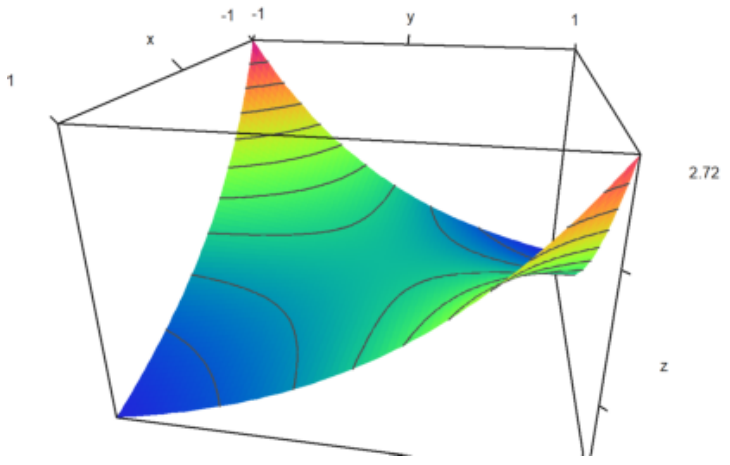
Pengaturan bawaannya adalah level=”auto”, yang mengkomputasi beberapa level garis secara otomatis. Seperti yang dilihat pada plot, level adalah range kelas.

Ragam bawaan dapat diubah. Untuk plot kontur berikut, kita menggunakan grid yang lebih halus 100x100 titik, 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=1.5,angle=45°,height=20°):
```




```
>plot3d("exp(x*y)",angle=100°,>contour,>spectral,level="thin",scale=1.5):
```

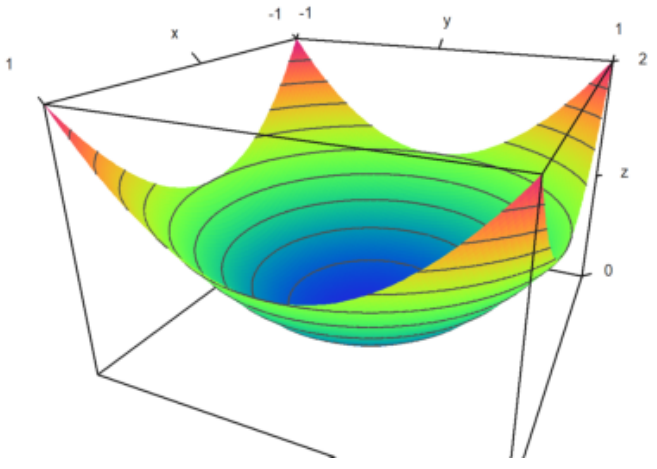


Arsiran bawaan menggunakan warna abu-abu. Akan tetapi, range spectral pada warna juga tersedia.
- >spectral: Menggunaan skema spektral bawaan

- `color=...: Menggunakan warna khusus atau skema-skema spektral`

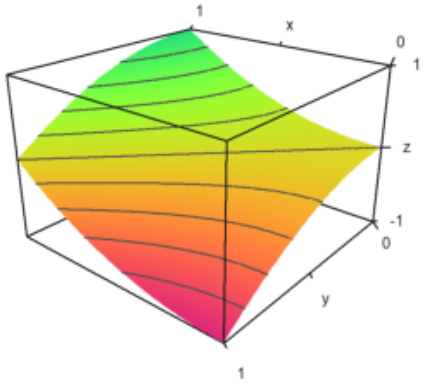
Untuk plot berikut kita menggunaan skema spektral biasa dan meningkatkan jumlah titik (n) untuk mendapatkan gamabar yang sangat halus.

```
>plot3d("x^2+y^2",>spectral,>contour,n=100,scale=1.5,level="thin"):
```



Disamping level garis yang otomatis, kita juga dapat mengatur nilai level garis. Kita akan menghasilkan level=”thin” dibanding range pada level.

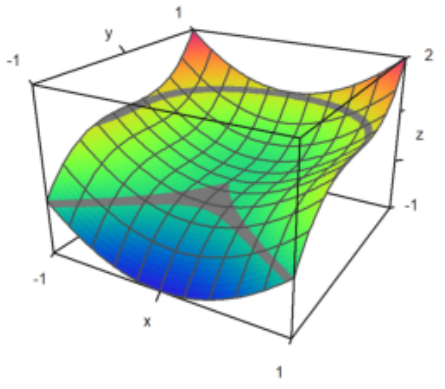
```
>plot3d("x^2-y^2",0,1,0,1,angle=220°,level=-1:0.2:1,color=redgreen):
```



Di dalam plot berikut, kita menggunakan tali level yang sangat luas dari -0.1 ke 1, dan dari 0.9 ke 1. Level ini dimasukkan sebagai sebuah matriks denga tali level sebanyak kolom.

Lebih lanjut, kita menghamparkan grid dengan interval 10 di dalam setiap direksi.

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

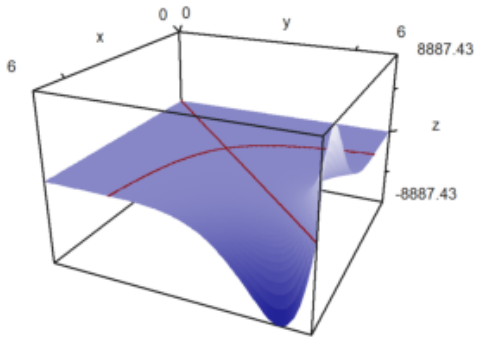


Di dalam contoh berikut, kita plot sebuah himpunan di mana

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

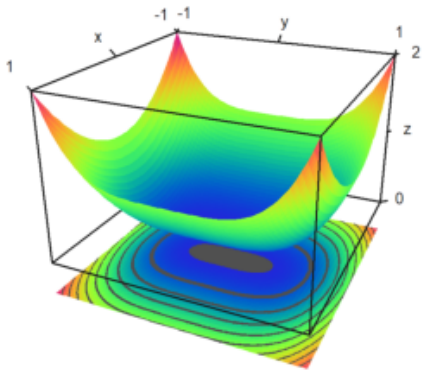
Kita menggunakan garis tipis tunggal untuk level garis.

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



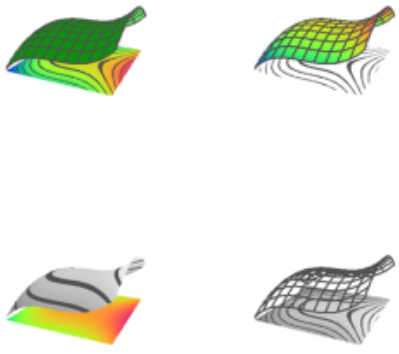
Bisa juga menampilkan bidang kontur di bawah plot. Warna dan jarak ke plot dapat dispesifikasi.

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



Berikut beberapa lagi ragam. Kita selalu mematikan rangka (frame), dan menggunakan banyak 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):
```

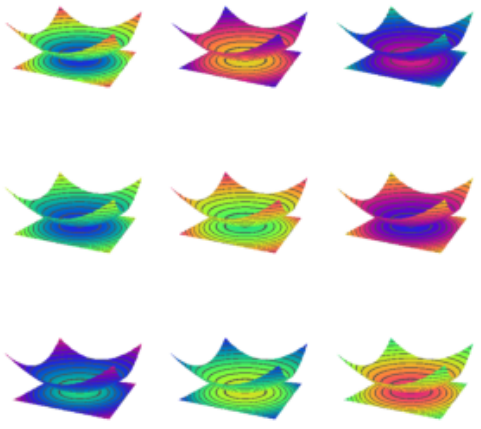


Berikut beberapa skema spektral lainnya, nomor 1 sampai 9. Akan tetapi kita dapat juga menggunak color=nilai, dimana nilainya:

- spectral: untuk range dari biru sampai merah

- white: untuk range yang lebih redup
- yellowblue,purplegreen,blueyellow,greenred
- blueyellow, greenpurple,yellowblue,redgreen

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



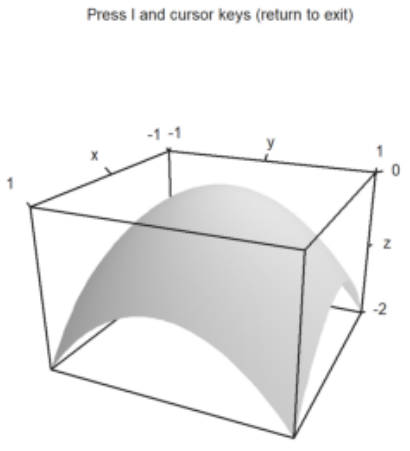
Sumber cahaya dapat diubah dengan l atau kursor selama interaksi pengguna. Sumber cahaya dapat juga diatur dengan parameter.

- light: petunjuk untuk cahaya

```
- amb: sekitaran cahaya, antara 0 dan 1
```

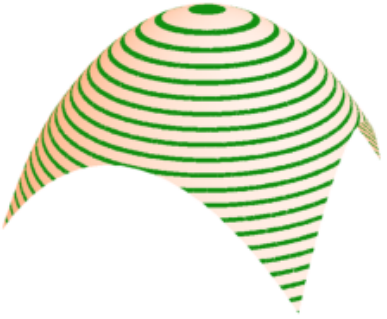
Perhatikan bahea program tidak membuat perbedaan antara sisi plot. Tidak ada bayangan. Jika pelu bayangan, kamu butuh Provray.

```
>plot3d("-x^2-y^2", ...
>  hue=true,light=[0,0.5,1],amb=0.6,user=true, ...
>  title="Press l and cursor keys (return to exit)"):
```



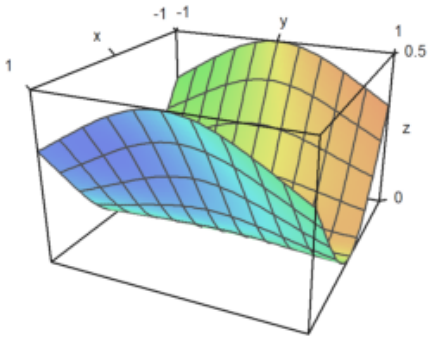
Parameter color mengubah warna permukaan. Warna level garis juga dapat diubah.

```
>plot3d("-x^2-y^2",color=rgb(1,0.5,0.2),hue=true,frame=false, ...
>  zoom=3,contourcolor=green,level=-2:0.1:1,dl=0.01):
```



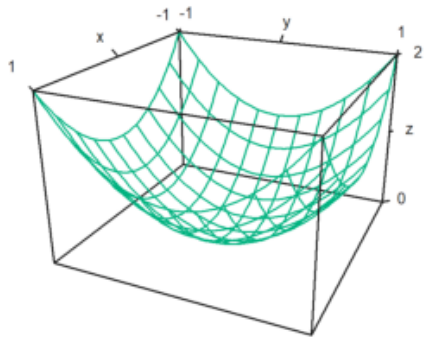
Color=0 memberikan efek warna pelangi yang spesial.

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



Permukaan juga dapat transparan dengan parameter >transparent.

```
>plot3d("x^2+y^2",>transparent,grid=10,wirecolor=rgb(0,0.7,0.5)):
```



*Plot implisit

Ada juga plot implisit pada tiga dimensi. Euler menghasilkan potongan melalui objek. Fitur plot3d termasuk plot implisit. Plot-plot ini memperlihatkan himpunan nol sebuah fungsi pada tiga variabel.

`Solusi dari`

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

Dapat divisualisasikan didalam potongan parallelke bidang x-y, x-z,dan y-z.

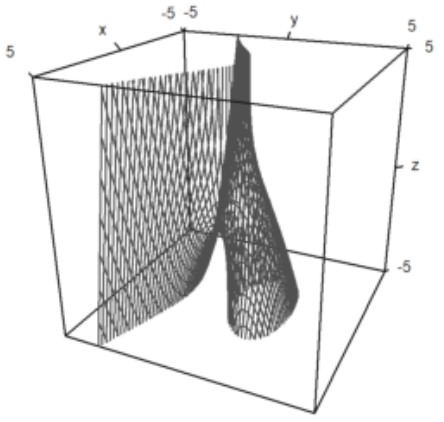
- implisit=1: Potongan parallel ke bidang x-y

- implisit=2: Potongan parallel ke bidang x-z
- implisit=3: Potongan parallel ke bidang y-z

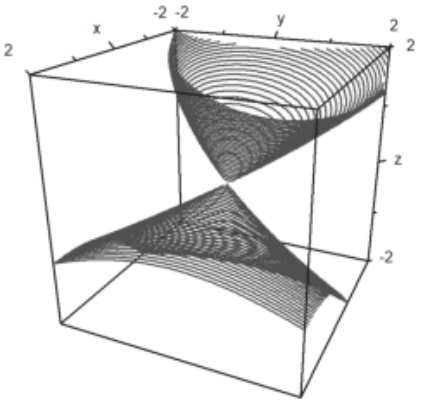
Tambahkan nilai berikut. Di dalam contoh yang kita plot kan

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

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




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


x
-2
-2
y
2
2
2
z
-2

Plot 3D data

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

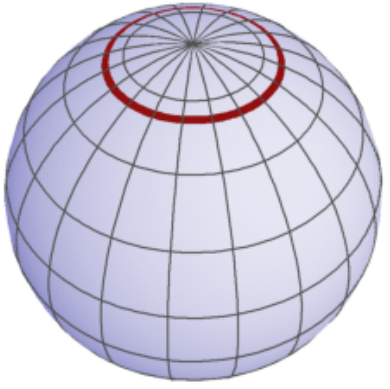
$$\gamma(t,s) = (x(t,s), y(t,s), z(t,s))$$

Oleh karenax,y,z matriks, kita asumsikan (t,s) melalui grid persegi. Sebagai hasilnya, kita dapat plot gambar segiempat pada ruang.

Kita bisa menggunakan bahasa matriks dari Euler untuk membuat koordunat secara efektif.

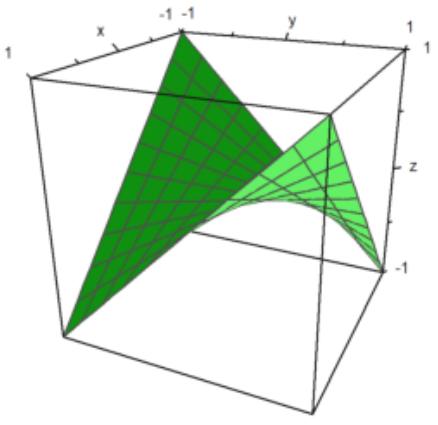
Pada contoh berikut ini, kita menggunakan vektor nilai t dan vektor kolom nilai s untuk membuat parameter permukaan bola. Pada penggambarannya, dalam kasus kita adalah wilayah polar.

```
>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 contoh yang menggambarkan grafik fungsi.

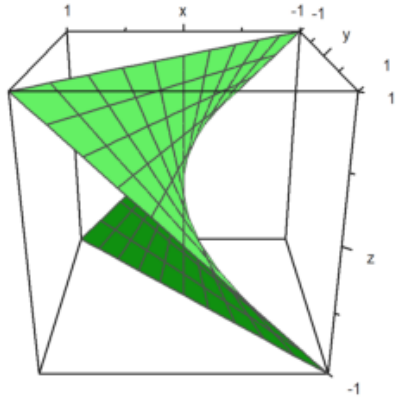
```
>t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,t*s,grid=10):
```



Namun, kita dapat membuat semua urutan permukaan. Berikut permukaan yang sama seperti fungsi.

$$x = y\,z$$

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



Kita dapat membuat lebih banyak bidang.

Pada contoh berikut kita membuat arsiran dari bola yang terdistorsi. Koordinat bola seringnya berupa:

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

dengan

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

Kita mendistorsikan ini dengan

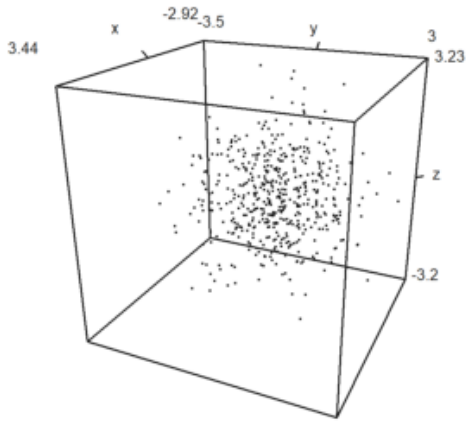
$$d(t,s) = \frac{\cos(4t) + \cos(8s)}{4}.$$

```
>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, himpunan titik-titik juga dapat dibuat. Untuk plot titik yang menandakan data di dalam ruang, kita membutuhkan tiga vektor untuk koordinat titik.

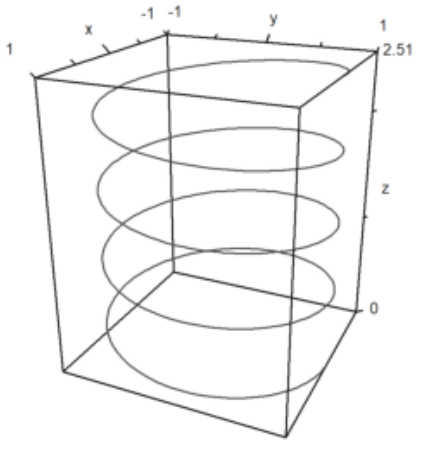
Ragamnya dapat dibuat dengan points=true;

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

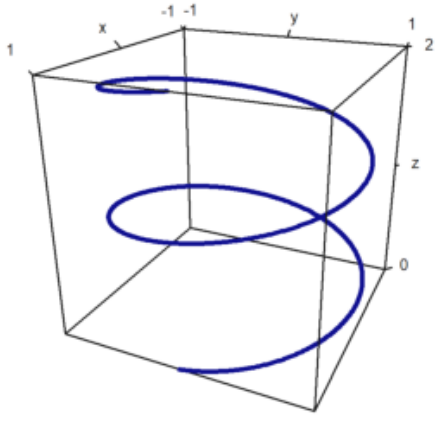


Dapat juga plot sebuah kurva dalam 3D. Di dalam kasus ini, lebih mudah untuk menghitung terlebih dahulu titik pada kurva. Untuk kurva pada bidang, kita menggunakan rangkaian koordinat dan parameter "wire=true".

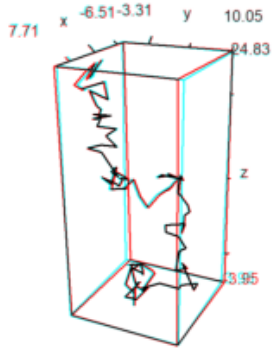
```
>t=linspace(0,8pi,500); ...
>plot3d(sin(t),cos(t),t/10,>wire,zoom=3):
```




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

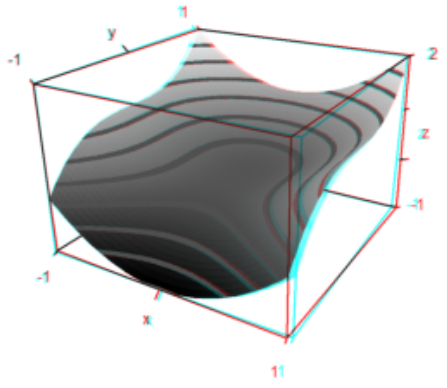



```
>X=cumsum(normal(3,100)); ...
> plot3d(X[1],X[2],X[3],>anaglyph,>wire):
```



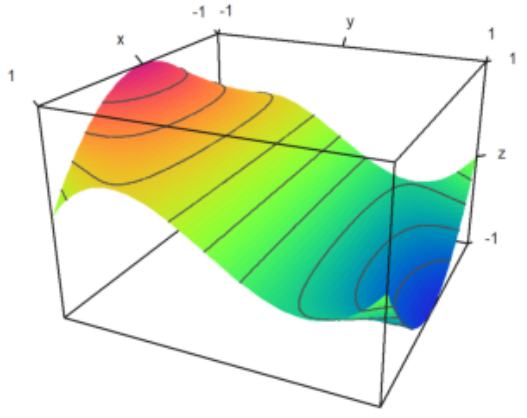
EMT dapat juga plot dalam mode anaglyph. Untuk melihat sebuah plot, kita butuh kacamata merah atau biru muda.

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



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

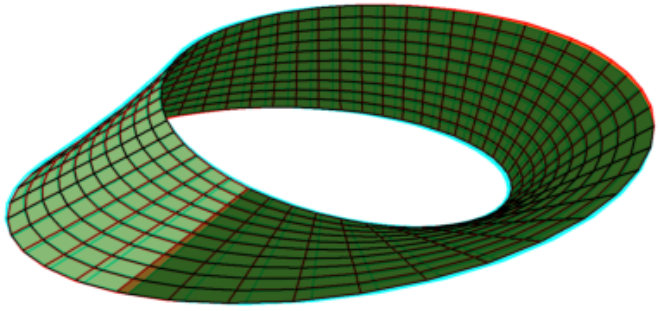
```
>plot3d("x^2*y^3-y",>spectral,>contour,zoom=3.2,level="thin"):
```



Euler juga dapat plot bidang yang di parameterkan, ketika parameternya nilai x,y,z pada gambar grid segiempat dalam ruang.

Untuk demo berikut, kita mengatur parameter u dan v, dan membangkitkan ruang koordinat dari plot-plot 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,<wire,<frame,scale=2.3):
```



Berikut ini adalah contoh yang lebih komplit yang apik dengan glass merah/biru muda.

```
>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=9,light=[1,0,-1],zoom=2.8,>anaglyph):
```

Bar plot juga dapat dibuat. Untuk membuat bar plot berikut, kita harus membuat

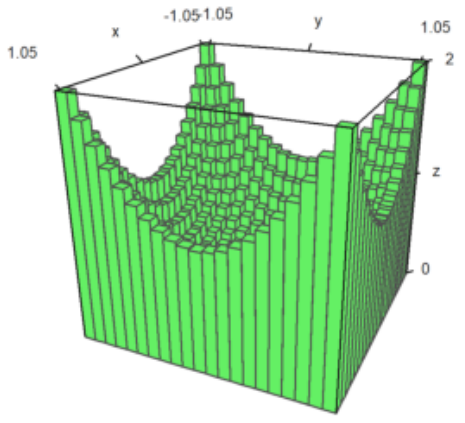
- x: vektor baris dengan n+1 elemen

- y: vektor kolom dengan n+1 elamen
- z: matriks nilai nxn

z dapat dilebarkan, tetapi hanya untuk nxn nilai akan digunakan.

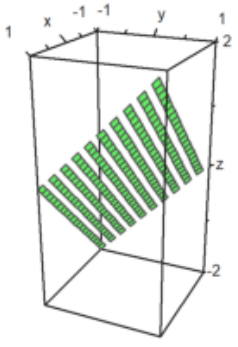
Di dalam contoh berikut, kita menghitung terlebih dahulu nilai. Lalu kita menyesuaikan x dan y, jadi vektor titik pusat pada nilai digunakan.

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



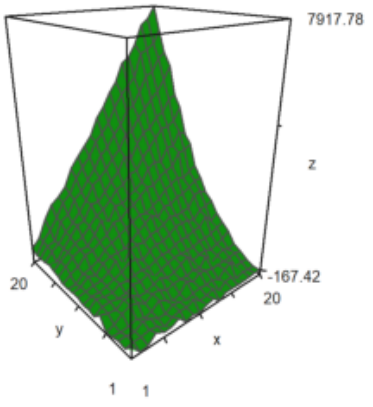
Mungkin juga dapat untuk membagi plot sebuah permukaan dalam dua atau lebih bagian.

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

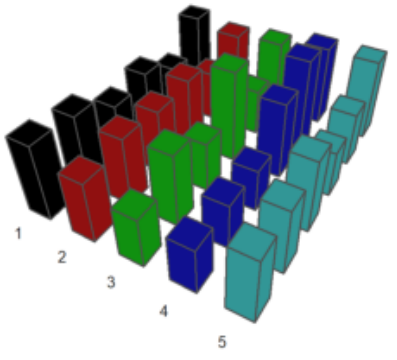


Jika memuat atau membangkitkan sebuah data matriks M dari sebuah file dan butuh untuk plotkannya di dalam 3D kita bisa antara menyekalakan matriks ke [-1,1] dengan scale(M), atau skalakan matriks dengan >zscale. Pembangkitan atau pemuatan sebuah data matriks bisa dikombinasikan dengan skala faktor individual yang diterapkan bertambah.

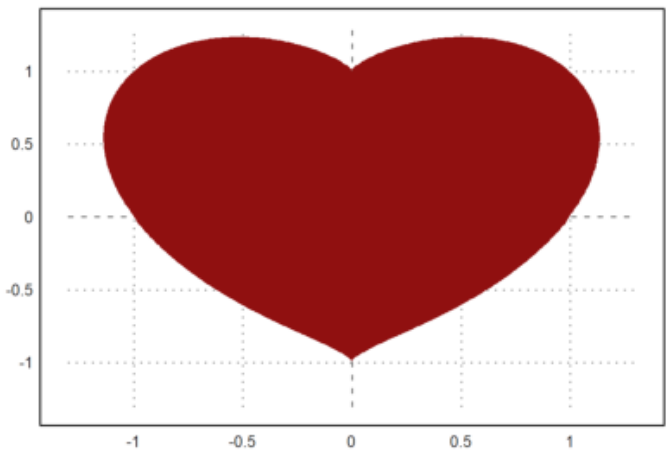
```
>i=1:20; j=i'; ...
>plot3d(i*j^2+100*normal(20,20),>zscale,scale=[1,1,1.5],angle=-40°,zoom=1.8):
```




```
>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]):
```


1
2
3
4
5

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




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

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

Kita menginginkan membentuk kurva hati sekeliling sumbu y. Berikut ini ekspresi yang dapat mendefinisikan hati:

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

Sekarang kita akan mengatur

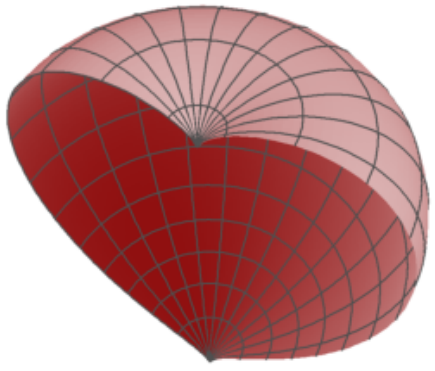
$$x = r.cos(a), \quad y = r.sin(a).$$

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

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

Ekpspresi dapat mendefinisikan fungsi numerik yang memecahkan r, jika a diberikam. Dengan fungsi itu kita dapat plot hati 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°):
```

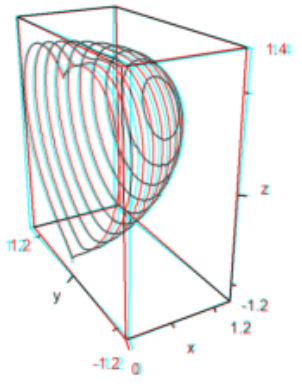


Berikut ini adalah sebuah plot 3D dari gambar yang terotasi di sekeliling dengan sumbuz. Kita mendefinisikan fungsi yang mendeskripsikan objek.

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


1.44
z
1.2
y
-1.2
1.2
x
-1.2
0

3D plot yang spesial

Fungsi plot3d baik untuk dimiliki, tetapi tidak memenuhi semua yang kita butuhkan, Diantara rutinitas sederhana biasanya, dapat juga mendapatkan plot yang telah dibingkai pada bebagai objek yang kamu sukai.

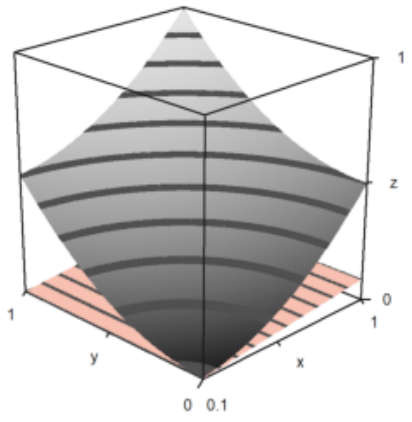
Walaupun Euler bukan sebuah program 3D, Euler dapat mengkombinasikan beberapa objek sederhana. Kita mencoba menvisualisasikan sebuah paraboloida tangen nya.

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

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

Sekarang framedplot() menyajikan bingkai dan mengatur pemandangan.

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

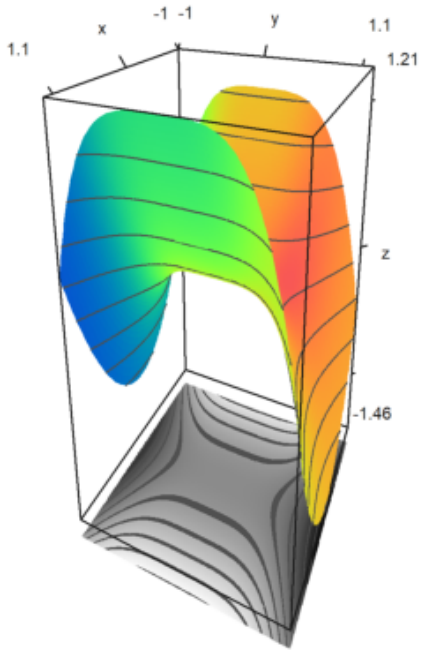


Di dalam langkah yang sama, kita dapat plot kontur bidang secara manual. Perhatikan bahwa plot3d() mengatur jendela ke fullwindow() dalam pengaturan bawaannya, tetapi plotcontourplane() mengambil peran fungsi plot3d().

```
>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,>spectral,level="thin");
    window(wi);
    reset();
  endfunction
```

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



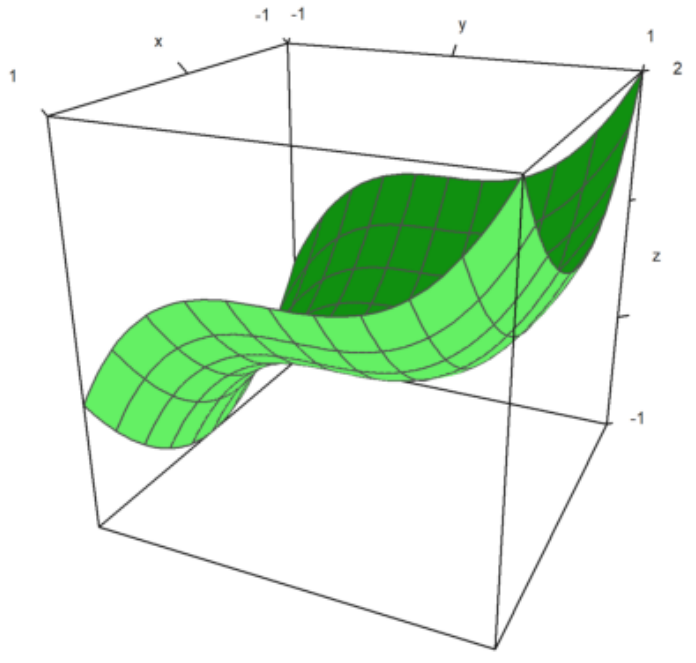
Animasi

Euler dapat menggunakan frames untuk menghitung terlebih dahulu animasi.

Suatu fungsi yang menggunakan teknik tersebut adalah rotate. Rotate dapat mengubah sudut pandang dan menggambar ulang sebuah plot 3D. Fungsi memanggil "addpage()" untuk setiap plot baru. Akhirnya fungsi akan menganimasikan plot.

Tolong untuk belajar sumber dari rotate untuk melihat detail lebih lanjut.

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


-1
-1
x
y
1
1
2
z
-1

Menggambar Povray

Dengan bantuan dari file Euler, povray.e ,Euler dapat membangkitkan file-file Povray. Hasilnya sangat indah untuk dilihat.

Kita butuh menginstall Povray (32bit atau 64bit) dari

```
See: http://www.povray.org | Unduh Povray program
dan meletakkan sub-direktori "bin" Povray ke dalam environtmet path
```

atau memidahkan variabel "defaultpovray" ke "pvengine.exe".

Antarmuka Povray Euler membangkitkan file-file Povray di dalam home directory penggunanya, dan memanggil Povray untuk menguraikan file-file Povray-nya. Nama file bawaannya adalah judulfile.pov, dan direktori bawaannya adalah eulerhome(), seringnya C:\Users\Username\Euler. Povray membangkitkan file PNG yang dapat dimuat dengan Euler ke dalam notebook. Untuk menghapus file-filenya, gunakan povclear().

Fungsi pov3d mirip dengan fungsi plot3d. Fungsi itu dapat membangkitkan grafik fungsi f(x,y), atau bidang dengan koordinat X,Y,Z di dalam matriks, termasuk level garis yang opsional. Fungsi ini mulai menggambar jejak sinar garis secara otomatis, dan memuat gambar ke dalam notebook Euler.

Di samping pov3d(), ada banyak fungsi yang membangkitkan objek Povray. Fungsi-fungsi ini menampilkan string yang berisi kode Povray untuk suatu objek. Akhirnya, akhiri file dengan provend(). Secara bawaannya, penggambaran sinar garis akan dimulai, dan PNG akan disisipkan ke dalam notebook Euler.

fungsi objek memiliki parameter yang disebut "look", yang membutuhkan string dengan kode Povray untuk susunan dan akhiran objek, Fungsi povlook() dapat digunakan untuk membuat string kode Povray-nya. Fungsi ini memiliki parameter untuk warna, ketransparansinya, arsiran Phong, dll.

Perhatikan bahwa universe Povray memiliki sistem koordinatnya sendiri. Antarmuka ini menerjemahkan semua koordinat ke dalam sistem Povray. Jadi kita tetao dapat berpikir dalam sistem koordinat Euler dengan z yang ditunjukkan secara vertikal ke atas, dan sumbu x,y,z di dalam aturan tangan kanan.

Kita perlu untuk memuat file povray-nya.

```
>load povray;
```

Pastikan bahwa direktori penyimpanan Povray di dalam path. Jika tidak, tuliskan perintah berikut.

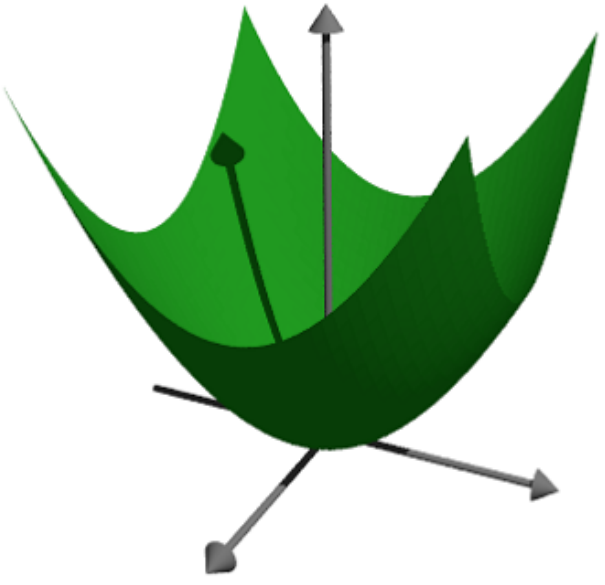
```
>defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

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

Untuk impresi pertama, mari kita plot fungsi yang sederhana. Perintah berikut membangkitkan file povray di dalam user directory kita, dan menjalankan Povray untuk setiap sinar garis menyusuri file ini.

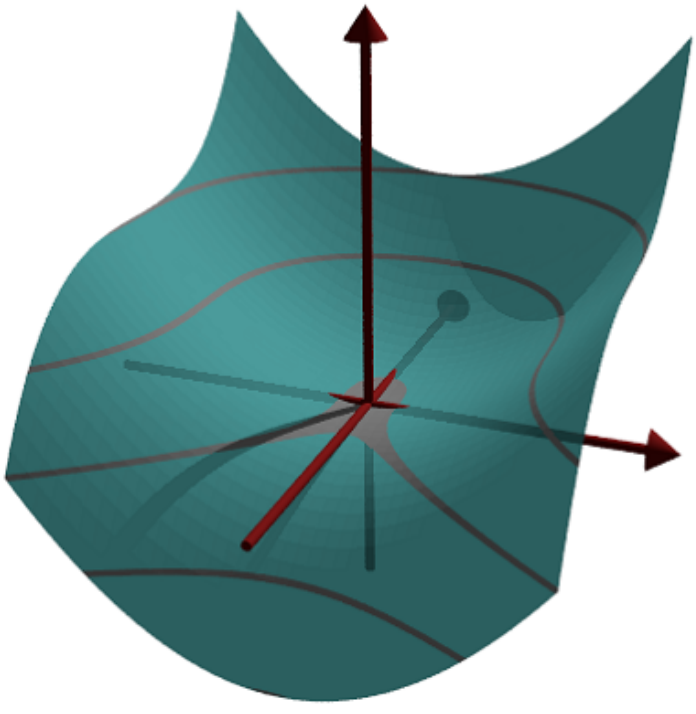
Jika kita memulai perintah berikut, Provray GUI akan dibuka, menjalankan file, dan menutupnya secara otomatis. Oleh karena alasan keamanan, kita akan ditanya, apakah kita akan mengizinkan file tersebut untuk berjalan. Kita dapat menekan cancel untuk memberhentikan pertanyaan berikutnya. Kita mungkin harus menekan OK dalam jendela Provray untuk mengetahui dialog pembuka Provray.

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



Kita dapat membuat fungsi transparan dan menambahkan akhiran yang lainnya. Kita juga dapat menambahkan level garis ke plot fungsi.

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



Seringkali, perlu juga mencegah penyekalaan fungsi dan skalakan fungsi sendiri.

Kita plot himpunan titik di dalam bidang kompleks, yang membentuk jarak ke 1 dan -1 sama dengan 1.

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

Plot Koordinat

Di samping fungsi, kita dapat plot dengan koordinat. Seperti di dalam plot3d, kita perlu tiga matriks untu mendefinisikan objek.

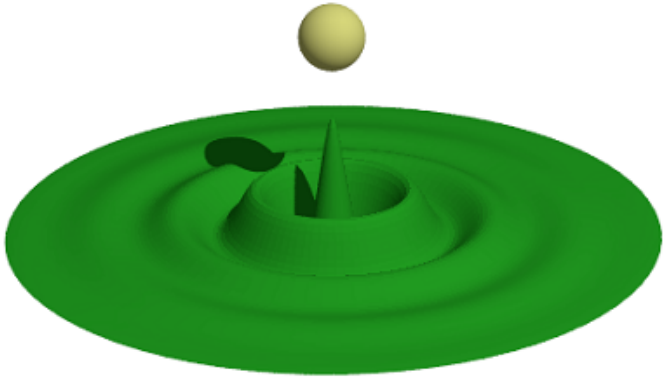
Di dalam contoh kita membuat fungsi di sekitar sumbu z.

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

Di dalam contoh berikut, kita akan plot damped wave (transmisi radio). Kita membangkitkan gelombang dengan bahasa matriks Euler.

Kita jugamemperlihatkan bagaimana sebuah objek tambahan dapat ditambahkan ke layar pov3d. Untuk pembangkitan objek, lihatlah contoh berikut. Perhatikan bahwa plot3d membuat skala pada plot, sehingga plot sesuai ke dalam unit kubik.

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



Dengan metode pengarsiran yang lebih lanjut pada Povray, sangat sedikit titik yang dapat menghasilkan bidang yang sangat halus. Hanya pada batas bidang dan didalam bayangan efek cahaya mungkin terlihat jelas.

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

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

```
  2  3
x  y
```

Persamaan dari bidang adalah [x,y,Z]. Kita komputasikan dua turunan ke x dan y persamaan dan cross product kedua turunan (turunan pada x dan pada y) sebagai normal.

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

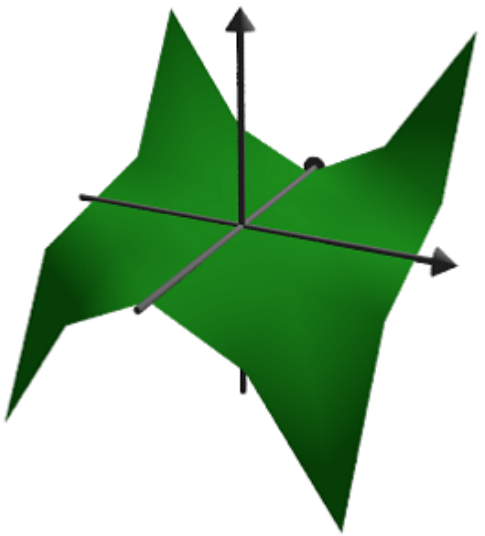
Kita mendefinisikan normal sebagai cross prodect dari turunan berikut dan mendefinisikan koordinat fungsi.

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

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

Kita hanya menggunakan 25 titik.

```
>x=-1:0.5:1; y=x';
>pov3d(x,y,Z(x,y),angle=25°, ...
>  xv=NX(x,y),yv=NY(x,y),zv=NZ(x,y),<shadow);
```



Berikut adalah buhul trefoil yang dibuat oleh A. Busser dalam Povray. Berikut adalah contoh buhul trefoil versi yang telah terimprovisasi.

See: Examples\Trefoil Knot | Trefoil Knot

Untuk gambar yang bagus dengan titik yang tidak terlalu banyak, kita menambahkan vektor normal di sini. Kita menggunakan Maxima untuk normal. 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);
```

Langkah kedua adalah menurunkan vektor ke x dan y.

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

Now the normal, which is the cross product of the two derivatives.

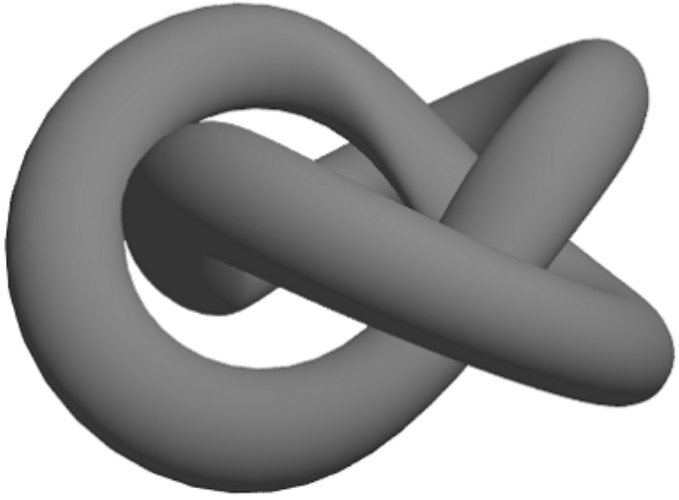
```
>dn &= crossproduct(dx,dy);
```

Kita sekarang menyelesaikan semuanya secara numerik.

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

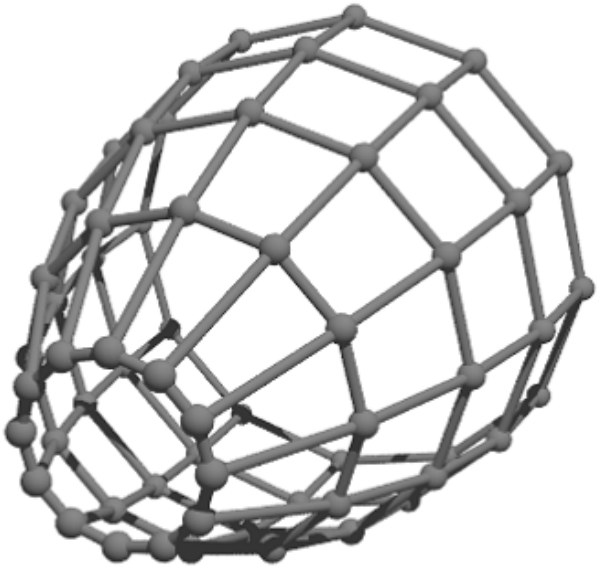
Vektor normal adalah penyelesaian ekspresi simbolik dn[i] untuk i=1,2,3. Sintaks untuk ini adalah &"ekspresi"(parameter). Metode ini adalah alternatif pada metode sebelumnya, di mana kita mendefinisikan ekspresi simbolik NX, NY, NZ terlebih dahulu.

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



Kita juga dapat membangkitkan sebuah grid dalam 3D.

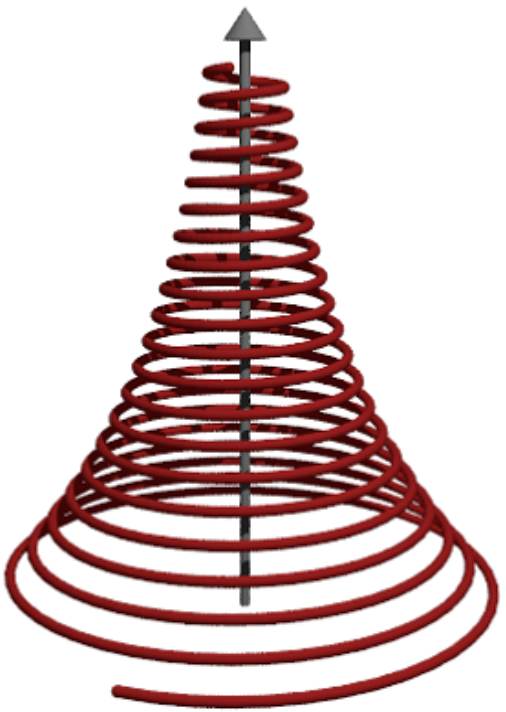
```
>povstart(zoom=4); ...
>x=-1:0.5:1; r=1-(x+1)^2/6; ...
>t=(0°:30°:360°)'; y=r*cos(t); z=r*sin(t); ...
>writeln(povgrid(x,y,z,d=0.02,dballs=0.05)); ...
>povend();
```



Dengan povgrid(), kurva dapat dibentuk

```
>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 kita menggunakan plot3d untuk plot bidang. Antarmuka povray dalam Euler juga membangkitkan objek Povrey. Objek-objek tersebut tersimpan sebagai string di Euler, dan perlu untuk dituliskan ke dalam file Povray

Kita memulai output dengan povstart().

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

Pertama kita mendefinisikan tiga silinder dan menyimpannya ke dalam string di Eule.

Fungsi povx() dll. Dengan sederhana menampilkan vektor [1,0,0], yang bisa digunakan

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

String mengandung kode Povray, yang tidak perlu dipahami pada hal 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 dapat kita lohat, kita menambahkan yekstur ke objek dalam tiga warna yang berbeda.

Ini dapat dilakukan dengan povlook() yang menampilkan string dengan kode Povray yang relevan. Kita dapat menggunakan warna dasar Euler, atau kita dapat mendefinisikan warna sendiri. Kita dapat juga menambahkan transparansi, atau mengubah warna di sekelilingnya.

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

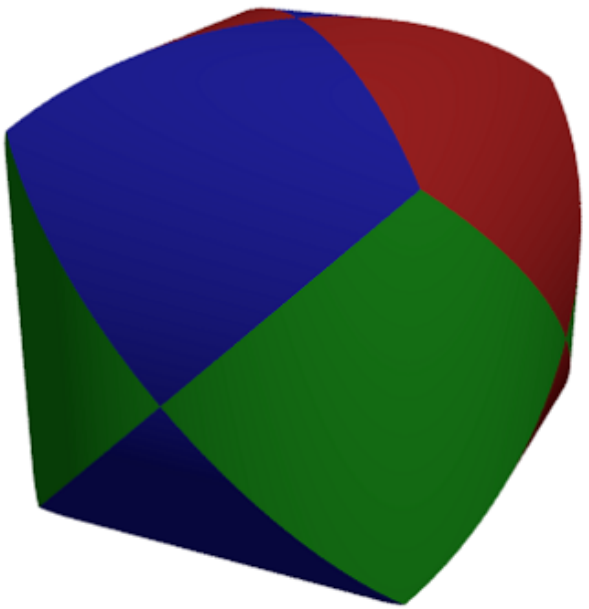
```
texture { pigment { color rgbf <0.101961,0.2,0.301961,0.1> }  }
finish { ambient 0.5 }
```

Sekarang kita mendefinisikan perpotongan objek adn menulis hasilnya ke file.

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

Perpotongan tiga silinder susah untuk divisualisasikan.

```
>povend;
```



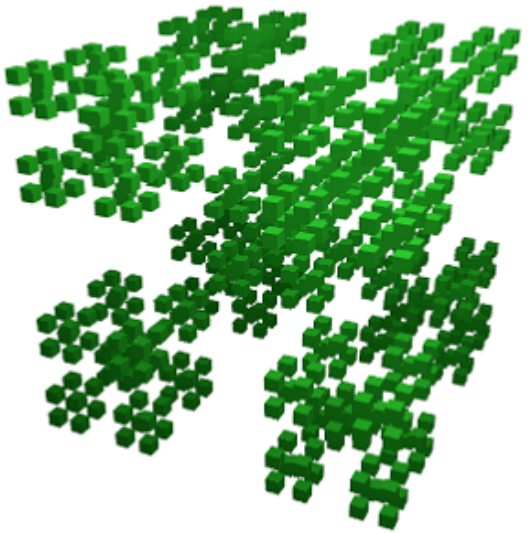
Fungsi berikut membangkitkan sebuah fraktal secara rekursif.

Fungsi pertama memperlihatkan bagaimana Euler menangani objek Povray. Fungsi povbox() menampilkan sebuah string, berisi koordinat kubus, tekstur dan akhiran.

```
>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 akan memotong suatu objek dari objek lainnya. Seperti perpotongan, ada bagian objek CSG dari Povray.

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

Koordinat kubik dari -1 berarti [-1,-1,-1].

Untuk mendemonstrasikan hal ini, kita mendefinisikan sebuah objek di

dalam Povray, dibanding menggunakan string di dalam Euler. Definisi ditulis ke dalam file.

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

Kita dapat menggunakan objek ini di dalam povobject() yang menampilkan string seperti biasanya.

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

Kita mengembangkan kubik kedua dan merotasi serta menyekalakannya sedikit.

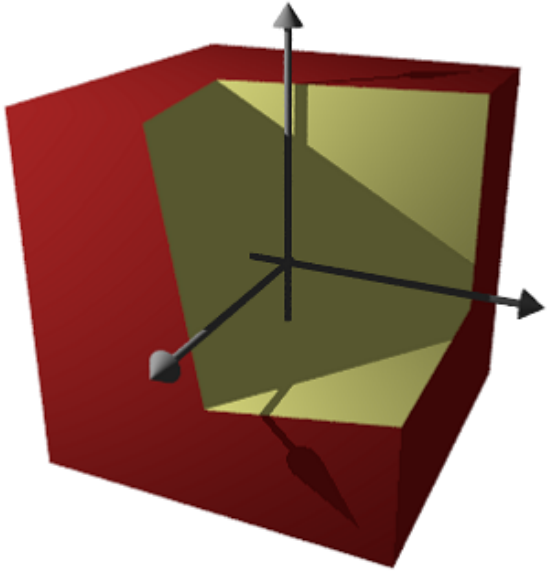
```
>c2=povobject("mycube",povlook(yellow),translate=[1,1,1], ...
>  rotate=xrotate(10°)+yrotate(10°), scale=1.2);
```

Selanjutnya kita akan menunjukkan perbedaan dua objek.

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

Sekarang kita menambahkan 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

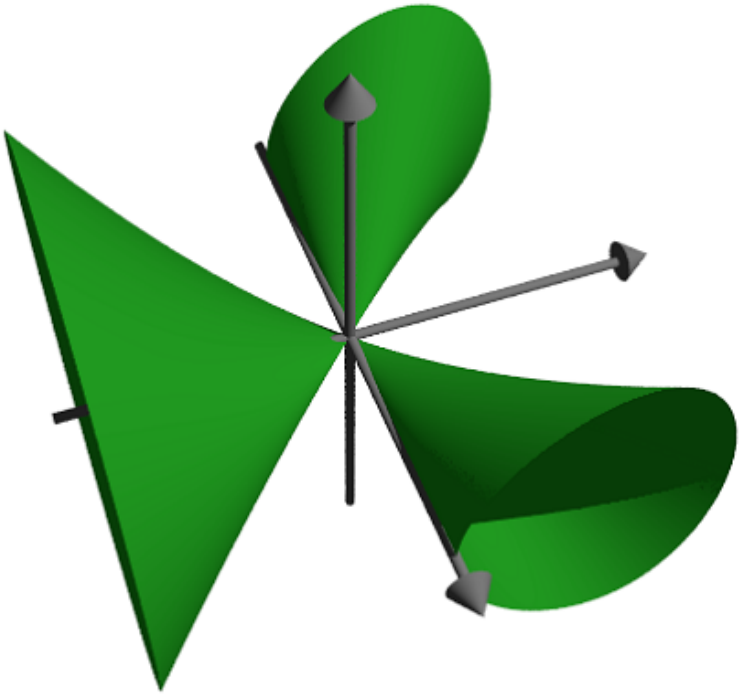
Povray dapat plot himpunan dimana f(x,y,z)=0 seperti parameter implisit dalam plot3d. Hasilnya terlihat jauh lebih baik.

Sintaks untuk fungsi sedikit berbeda. Kita tidak dapat menggunakan output Maxima atau ekspresi Euler.

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

Membuat bidang implisit. Perhatikan perbedaan sintaks di dalam ekspresi.

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



Objek Mesh

Di dalam contoh ini kita dapat membuat sebuah objek mesh dan menggambarkannya dengan informasu tambahan.

Kita akan memaksimalkan xy pada kondisi x+y=1 dan mendemonstrasikan sebuah level garis singgung tangensial.

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

Kita dapat menyimpan objek di dalam string seperti sebelumnya oleh karena terlalu besar. Jadi kita mendefinisikan objek di dalam sebuah file Povray menggunakan declare. Fungsi povtriangle() melaksanakannya secara otomatis. Fungsi itu dapat menerima vektor normal seperti pov3d().

Berikut ini mendefinisikan objek mesh, dan 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 disk yang akan berpotongan dengan bidang.

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

Menulis bidang minus dua disk.

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

Menuliskan dua perpotongan.

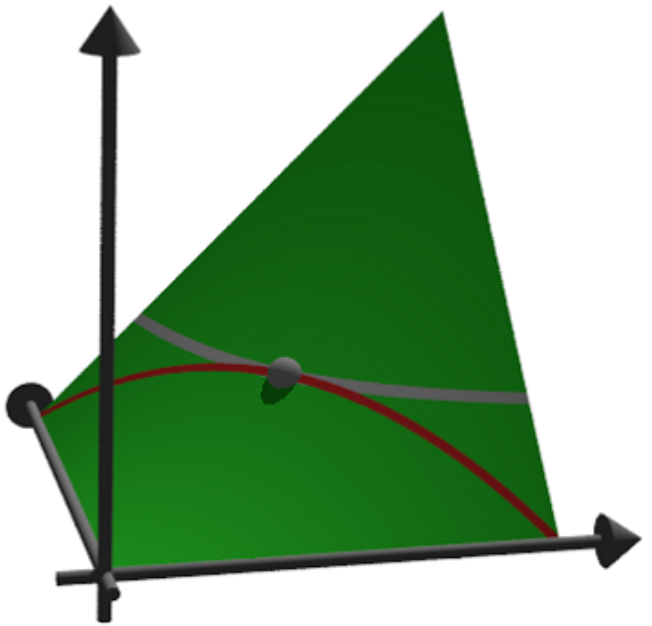
```
>writeln(povintersection([mesh,cl],povlook(red)); ...
>writeln(povintersection([mesh,ll],povlook(gray)));
```

Menunlis titik maksimum.

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

Menambahkan sumbu dan akhiran.

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



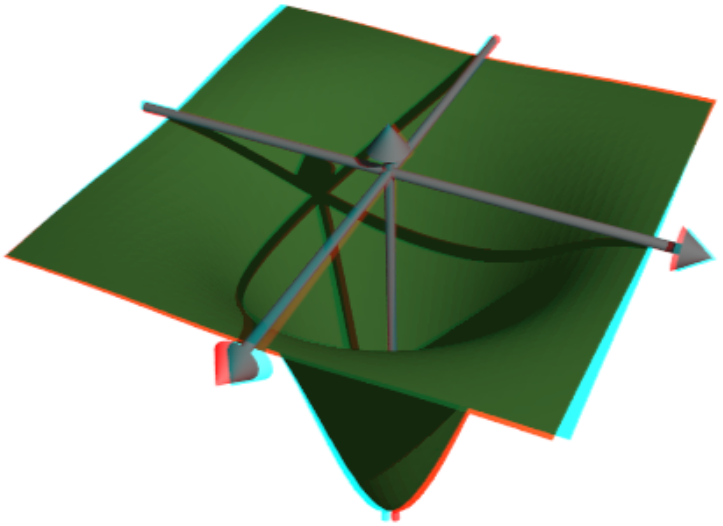
Anaglyph pada Povray

Untuk membangkitkan sebuah anaglyph untuk sebuah merah atau biru muda glasses, Povray akan berjalan dari dua posisi kamera yang berbeda. Hal ini akan membangkitkan dua file Povray dan dua file PNG yang dimuat dengan fungsi loadanaglyph()

Tentu saja, kita butuh merah atau biru muda glasses untuk melihat contoh berikut sebelumnya.

Fungsi pov3d() memiliki pergantian sederhana untuk membangkitkan anaglyphs.

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



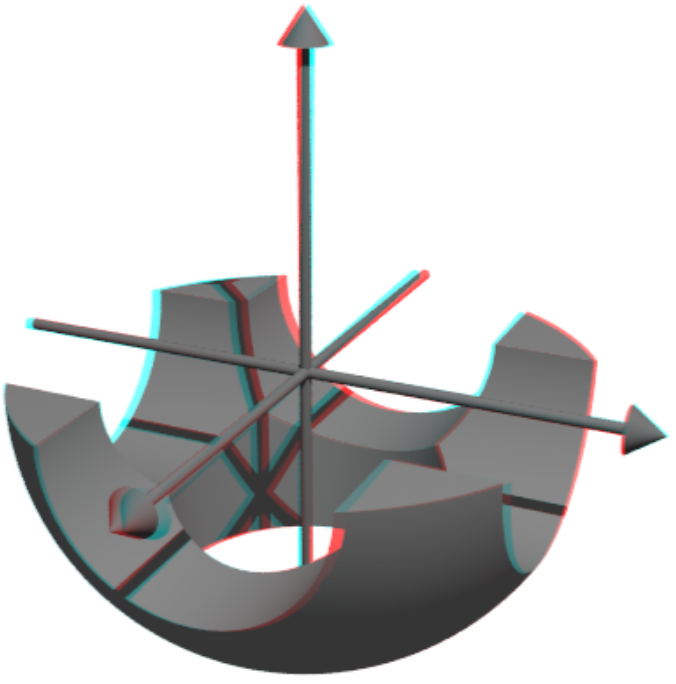
Jika kita membuat scene dengan objek, kita perlu membangkitkan sebuah pemandangan ke dalam fungsi, dan menjalankannya dua kali dengan nilai yang berbeda dari parameter anaglyph.

```
>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 ponaglyph() melakukan semua ini. Parameternya seperti kombinasi povstart() dan povend().

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



Mendefinisikan sendiri objek

Antar muka povray pada Euler mengandung banyak sekali objek. Akan tetapi kita tidak terpaku ke sana. Kita dapat membuat objek sendiri yang mengkombinasikan objek lain atau membuat objek yang baru sepenuhnya.

Kita mendemostrasikan sebuah torus. Perintah Povray untuk ini adalah "torus". Jadi kita menampilkan string dengan perintah ini dan parameternya. Perhatikan bahwa torus selalu terpaku pada titik asal.

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

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

Berikut adalah torus pertama kita.

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

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

Mari kita menggunakan objek ini untuk membuat sebuah torus kedua, diterjemahkan dan dirotasikan.

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

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

Sekarang kita menempatkan objrk ini ke dalam sebuah pemandangan. Untuk gambarannya, 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();
```

calls the Povray program. However, in case of errors, it does not display the error. You should therefore use

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

if anything did not work. This will leave the Povray window open.

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

Berikut adalah contoh penjelasan lebih lanjut. Kita memecahkan

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

dan memperlihatkan titik-titik yang mungkin dan optimum di dalam sebuah plot 3D.

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

Pertama, mari kita cek, jika contoh ini memiliki solusi pada semua permukaan.

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

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

Tenyata permukaan memiliki solusi.

Selanjutnya kita mendefinisikan dua objek. Pertama adalah bidang

$$a \cdot x \leq b$$

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

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

Selanjutnya kita mendefinisikan perpotongan untuk 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
```

Kita sekarang dapat plot pemandangan.

```
>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 sebuah lingkaran sekeliling nilai optimum.

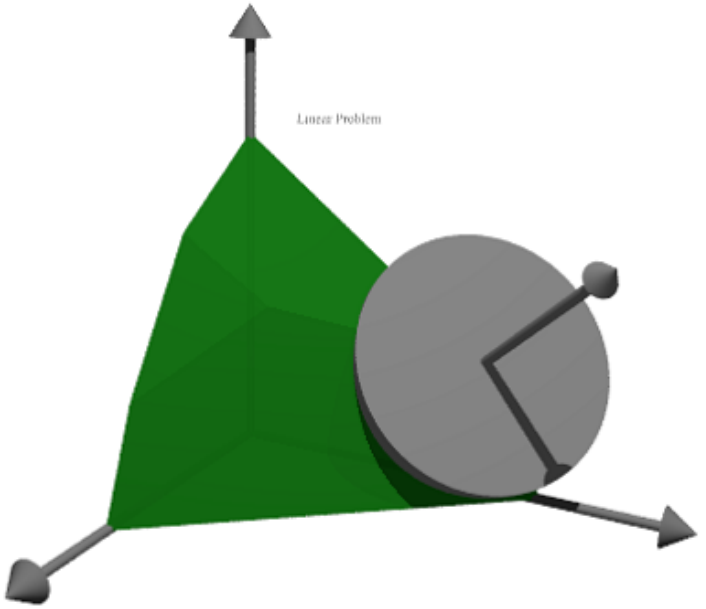
```
>writeln(povintersection([povsphere(x,0.5),povplane(c,c.x')], ...
>  povlook(red,0.9)));
```

Dan error di dalam arah dari optimum.

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

Kita menambahkan teks ke layar. Teks adalah sebuah objek 3D. Kita perlu menempatkannya menurut gambaran kita.

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


Linear Problem

Contoh lebih lanjut

Kita dapat menemukan beberapa contoh untuk Povray pada Euler dalam file berikut.

See: Examples/Dandelin Spheres

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