Menggambar Plot 3D dengan EMT

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

```
Euler menggambar fungsi tersebut menggunakan algoritma pengurutan untuk menyembunyikan bagian-bagi
```

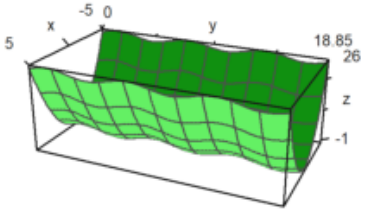
proyeksi. Standarnya adalah dari kuadran x-y positif menuju titik asal x=y=z=0, tetapi sudut=0Â° terlihat dari arah sumbu y. Sudut pandang dan ketinggian dapat diubah.

Euler dapat memplot

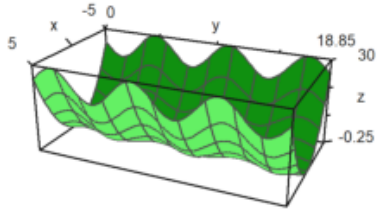
- permukaan dengan bayangan dan garis-garis level atau rentang level,
- awan titik-titik,
- kurva parametrik,
- permukaan implisit.

Plot 3D dari suatu fungsi menggunakan plot3d. Cara termudah adalah dengan memplot ekspresi dalam x dan y. Parameter r mengatur rentang dari 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

Fungsi dari dua Variabel

Untuk grafik sebuah fungsi, gunakan

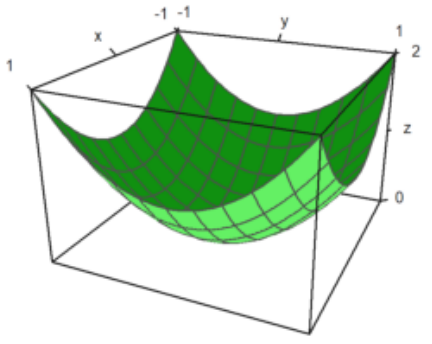
- ekspresi sederhana dalam x dan y,
- nama fungsi dari dua variabell
- atau matriks data.

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

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

Kita menggunakan nilai default n=40 dan grid=10.

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



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

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

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

```
Interrupt
Try "trace errors" to inspect local variables after errors.
plot3d:
    k=key(pr);
```

Rentang plot untuk fungsi dapat ditentukan dengan

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

Ada beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.

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

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

```
Interrupt
Try "trace errors" to inspect local variables after errors.
plot3d:
    k=key(pr);
```

Tampilan dapat diubah dengan berbagai cara.

- jarak: jarak pandang ke plot.
- zoom: nilai zoom.
- sudut: sudut ke sumbu y negatif dalam radian.
- ketinggian: ketinggian tampilan dalam radian.

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

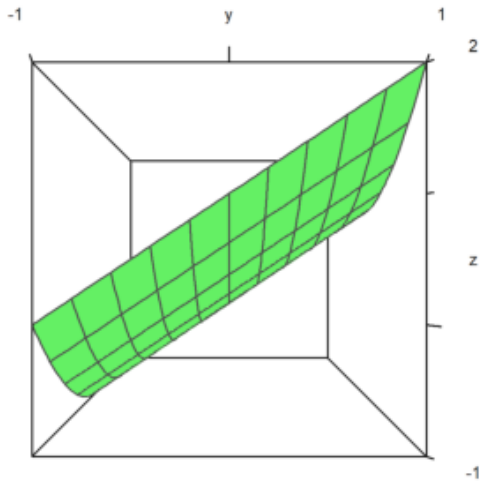
```
> view
```

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

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

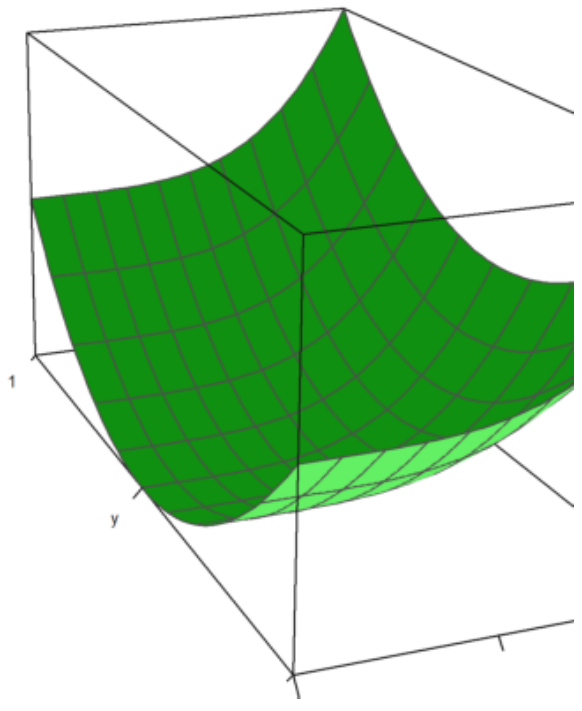
Dalam contoh berikut ini, sudut = 0 dan tinggi = 0 terlihat dari Sumbu y. Label sumbu untuk y disembunyikan dalam kasus ini.

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



Plot terlihat selalu berada di tengah-tengah kubus plot. Anda dapat memindahkan pusat dengan parameter center.

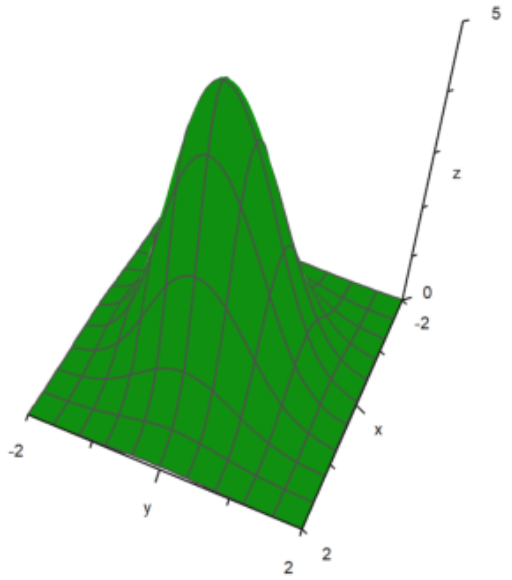
```
>plot3d("x^4+y^2",a=0,b=1,c=-1,d=1,angle=-20°,height=20°, ...
> center=[0.4,0,0],zoom=5):
```



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

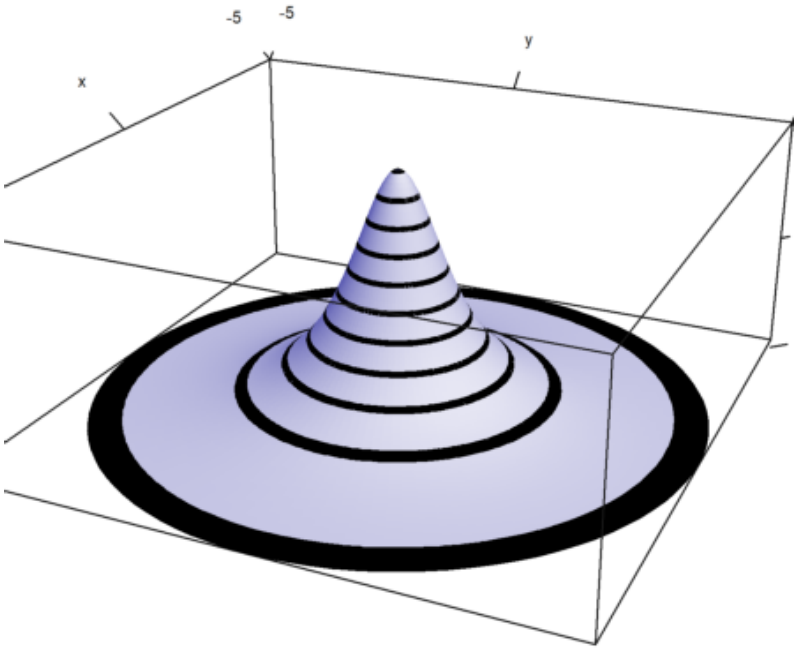
Jika Anda menonaktifkannya dengan scale=false, Anda harus berhati-hati agar plot tetap muat di dalam jendela plotting, dengan mengubah jarak pandang atau zoom, dan memindahkan bagian tengah.

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

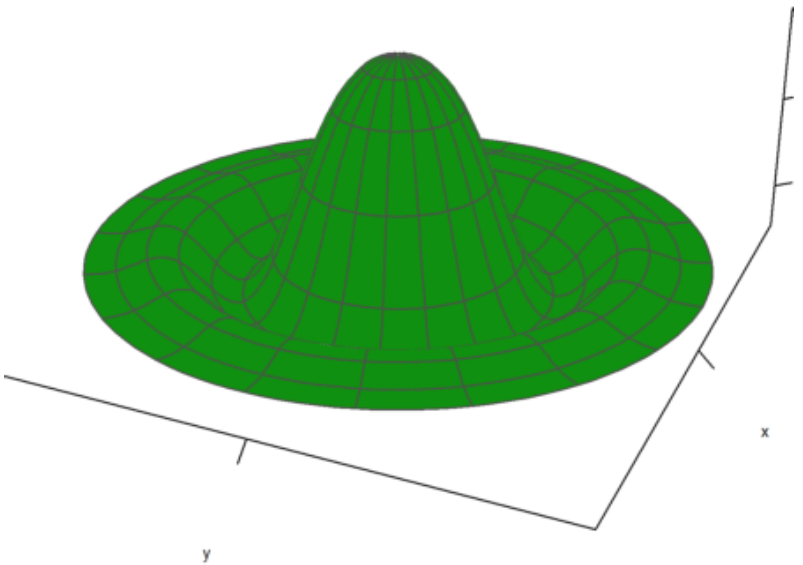


Plot kutub juga tersedia. Parameter polar=true menggambar plot polar. Fungsi ini masih harus berupa fungsi dari x dan y.

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



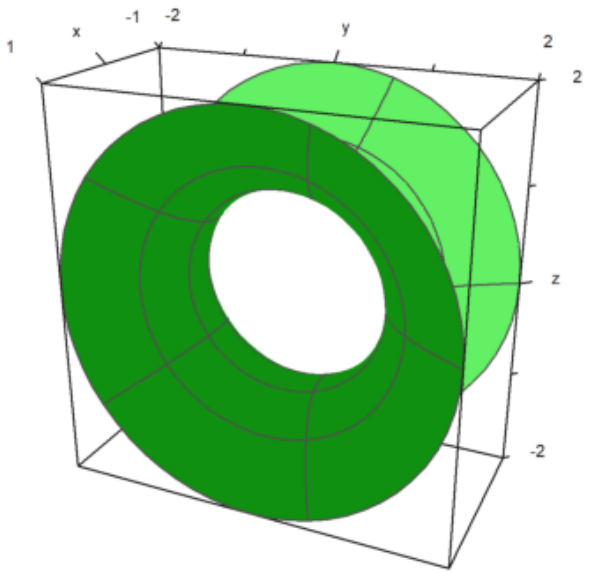
```
>function f(r) := exp(-r/2)*cos(r); ...
>plot3d("f(x^2+y^2)",>polar,scale=[1,1,0.4],r=pi,frame=3,zoom=4):
```



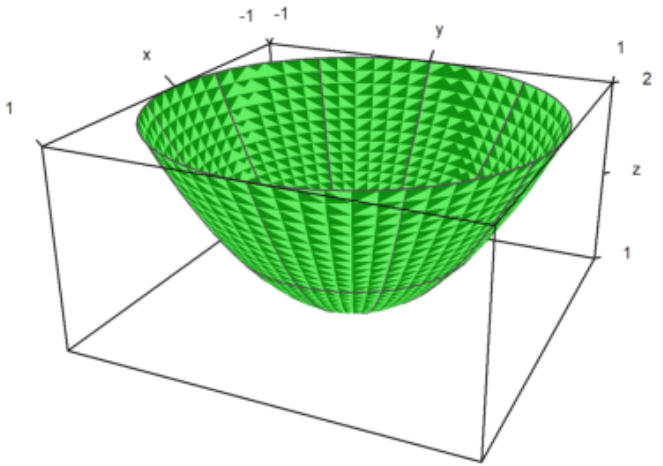
Parameter rotate memutar fungsi dalam x di sekitar sumbu x.

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

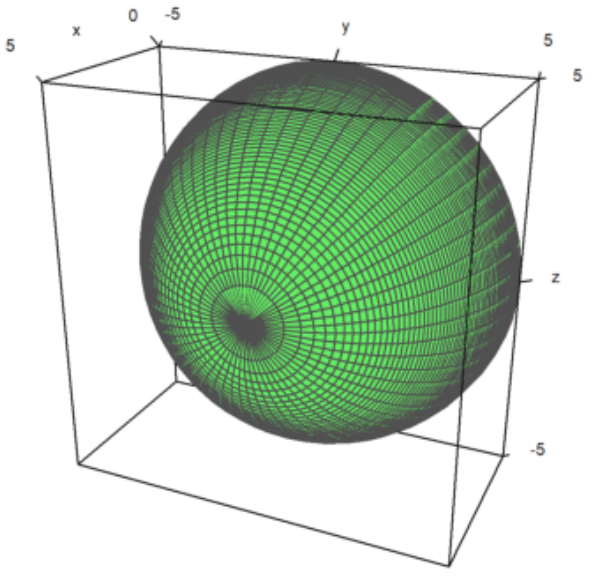
```
>plot3d("x^2+1",a=-1,b=1,rotate=true,grid=5):
```



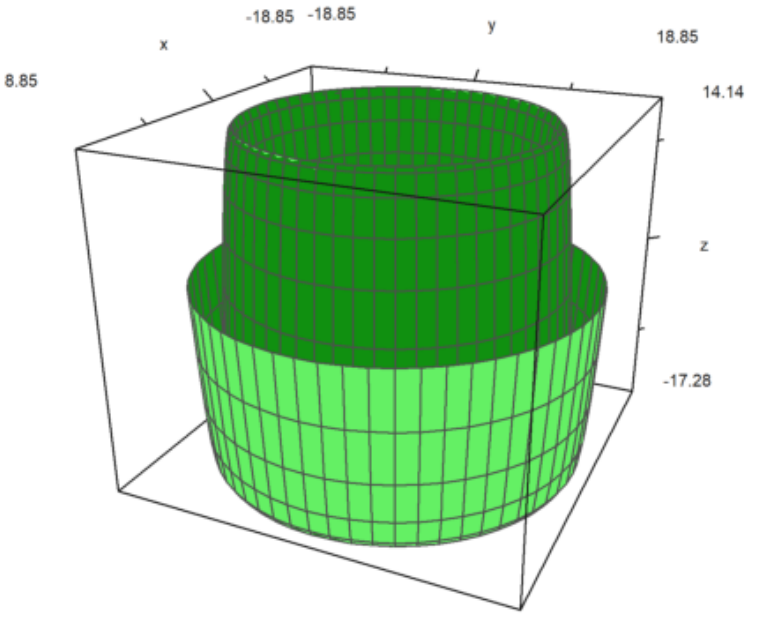
```
>plot3d("x^2+1",a=-1,b=1,rotate=2,grid=5):
```




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

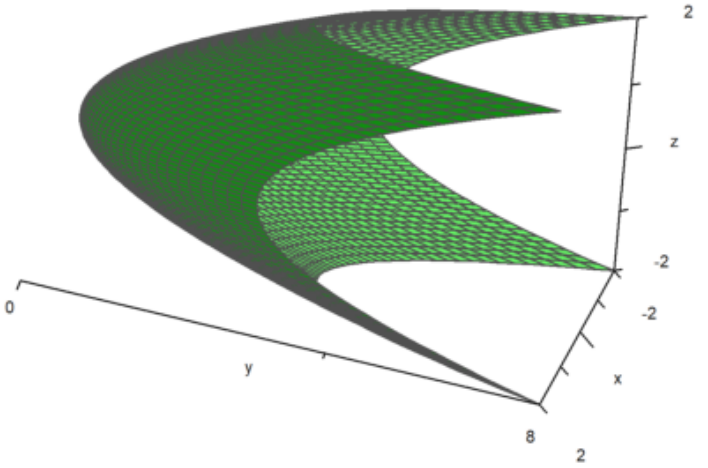



```
>plot3d("x*sin(x)",a=0,b=6pi,rotate=2):
```



Berikut adalah plot dengan tiga fungsi.

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


2
z
-2
-2
x
8
2
0
y

Plot Kontur

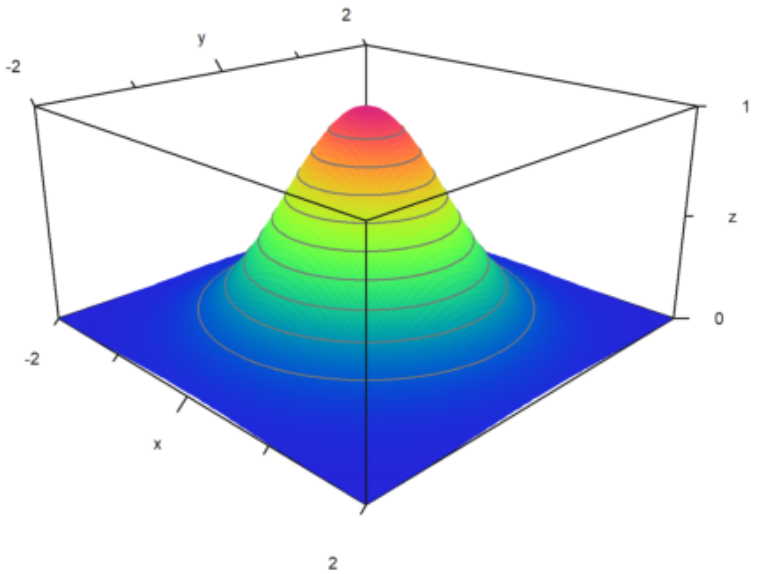
Untuk plot, Euler menambahkan garis-garis grid. Sebagai gantinya, Anda dapat menggunakan garis level dan plot dengan bayangan. Pada semua plot 3D, Euler dapat menghasilkan anaglyph merah/cyan.

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

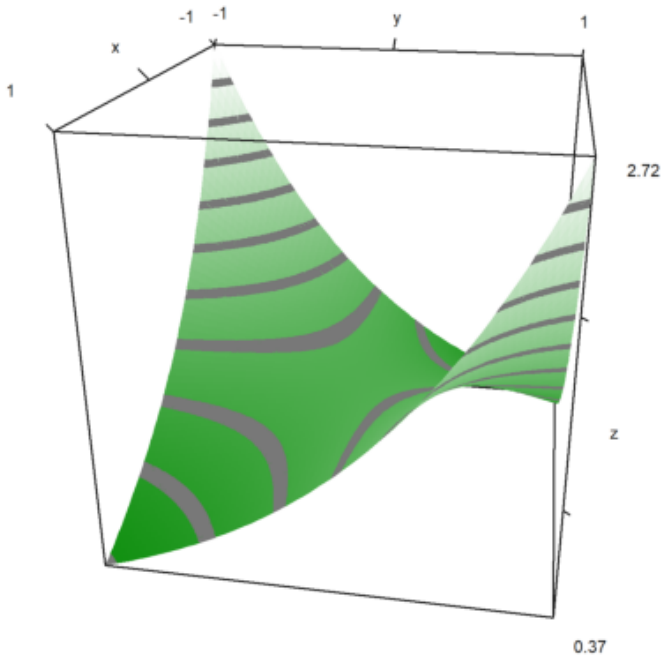
Defaultnya adalah level="auto", yang menghitung beberapa garis level secara otomatis. Seperti yang Anda
lihat pada plot, level sebenarnya adalah rentang level.

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

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




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

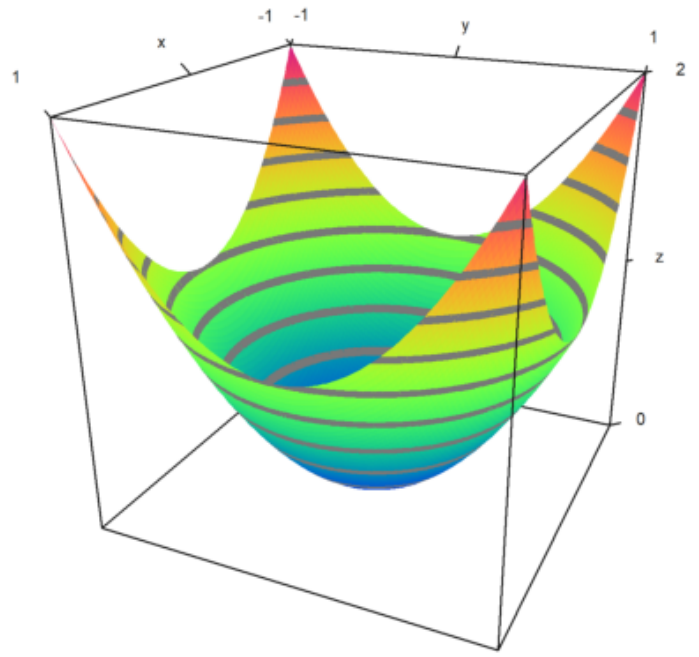


Bayangan default menggunakan warna abu-abu. Tetapi, rentang warna spektral juga tersedia.

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

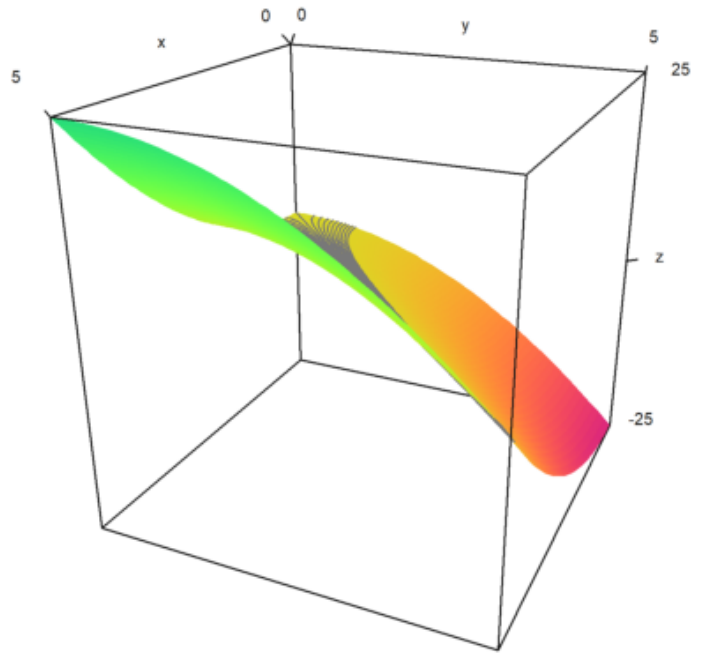
Untuk plot berikut ini, kita menggunakan skema spektral default dan menambah jumlah titik untuk mendapatkan tampilan yang sangat halus.

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



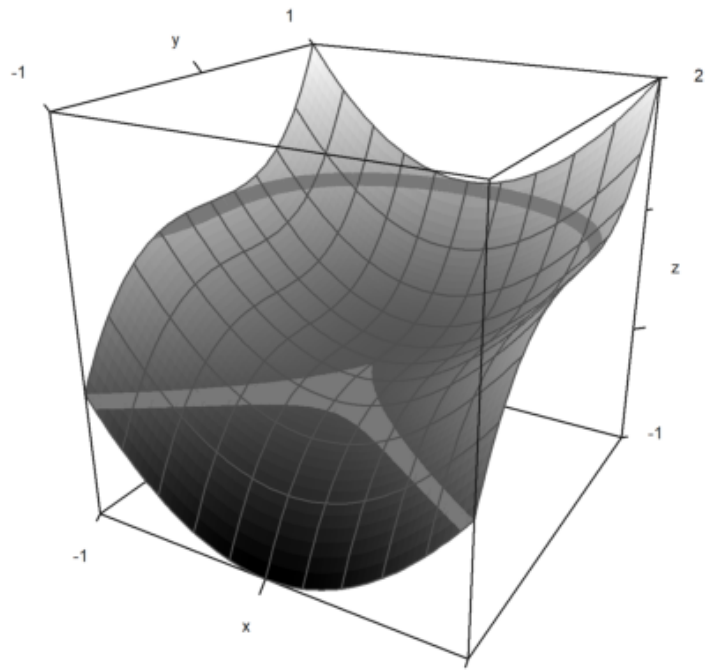
Sebagai pengganti garis level otomatis, kita juga dapat mengatur nilai dari garis level. Hal ini akan menghasilkan garis level yang tipis sebagai gantinya dari rentang level.

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



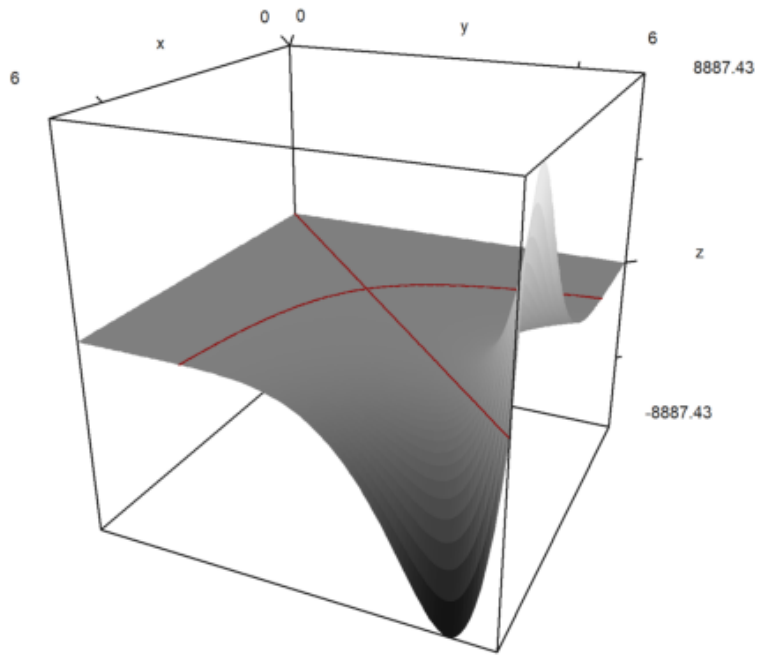
Pada plot berikut ini, kami menggunakan dua pita level yang sangat luas dari -0,1 hingga 1, dan dari 0,9 hingga 1. Ini dimasukkan sebagai matriks dengan batas-batas level sebagai kolom.

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



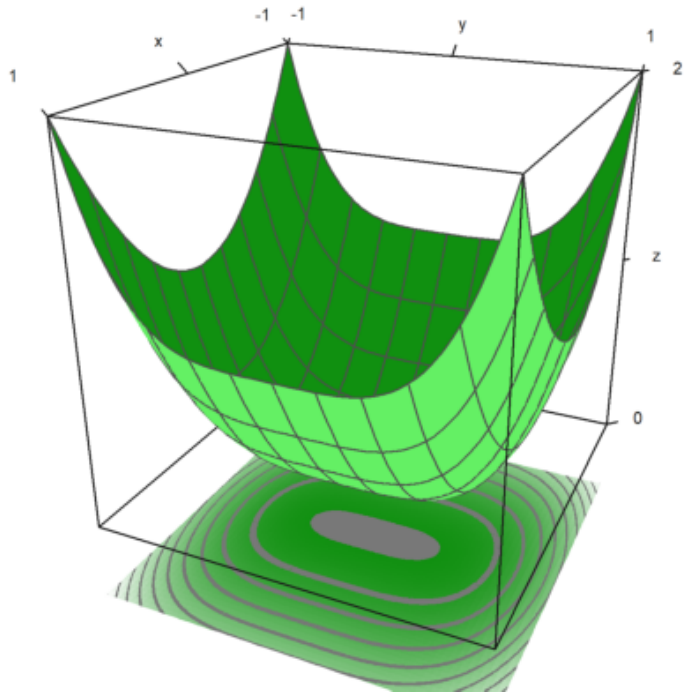
Pada contoh berikut, kami memplot himpunan, di mana
lateks: f(x,y) = xˆy-yˆx = 0 % gambar% EMT4Plot3D-021.png
Kita menggunakan satu garis tipis untuk garis level.

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



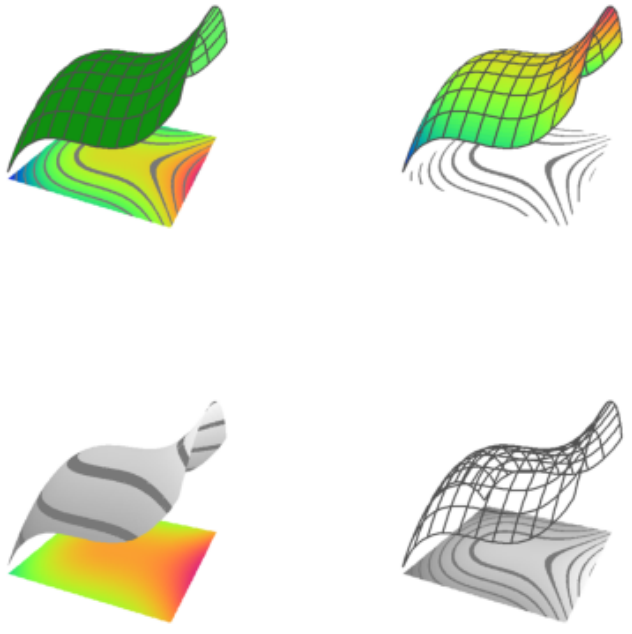
Dimungkinkan untuk menampilkan bidang kontur di bawah plot. Warna dan

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



Berikut adalah beberapa gaya lainnya. Kami 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, yang diberi nomor dari 1 sampai 9. Tetapi Anda juga dapat menggunakan color=value, di mana value

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

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

```
Floating point error!
Try "trace errors" to inspect local variables after errors.
image:
    a=A/norm(A);
```

Sumber cahaya dapat diubah dengan l dan tombol kursor selama interaksi pengguna. Ini juga dapat diatur dengan parameter.

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

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

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

```
Interrupt
Try "trace errors" to inspect local variables after errors.
plot3d:
    k=key(pr);
```

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

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

Warna 0 memberikan efek pelangi khusus.

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

Permukaan juga bisa menjadi 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-fitur plot3d termasuk plot implisit. Plot-plot ini menunjukkan himpunan nol dari sebuah fungsi dalam tiga variabel.
Solusi dari

lateks: f(x,y,z) = 0 % gambar% EMT4Plot3D-030.png

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

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

Tambahkan nilai-nilai ini, jika Anda mau. Pada contoh, kami memplot

lateks: M = \{ (x,y,z) : xˆ2+yˆ3+zy=1 \} % gambar% EMT4Plot3D-031.png

```
>plot3d("xˆ2+yˆ3+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,
```

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

Memplot Data 3D

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

lateks: \gamma(t,s) = (x(t,s),y(t,s),z(t,s))

Karena x,y,z adalah matriks, kita asumsikan bahwa (t,s) berjalan melalui kisi-kisi persegi. Hasilnya, Anda dapat memplot gambar persegi panjang dalam ruang.

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

Pada contoh berikut, kita menggunakan vektor nilai t dan vektor kolom nilai s untuk memparameterkan permukaan bola. Pada gambar kita dapat menandai daerah, dalam kasus kita daerah kutub.

```
>t=linspace(0,2pi,180); s=linspace(-pi/2,pi/2,90)'; ...
>x=cos(s)*cos(t); y=cos(s)*sin(t); z=sin(s); ...
>plot3d(x,y,z,>hue, ...
>color=blue,<frame,grid=[10,20], ...
>values=s,contourcolor=red,level=[90°-24°;90°-22°], ...
>scale = 1.4, height = 50°):
```

Berikut ini adalah sebuah contoh, yang merupakan grafik dari sebuah fungsi.

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

sebagai sebuah fungsi
lateks: x = y \, z

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

Dengan lebih banyak usaha, kita dapat menghasilkan banyak permukaan.

Pada contoh berikut, kita membuat tampilan berbayang dari bola yang terdistorsi. Koordinat biasa untuk bola adalah

lateks: \gamma(t,s) = (\cos(t)\cos(s),\sin(t)\sin(s),\cos(s)) % gambar% EMT4Plot3D-039.png

dengan

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

Kami mengubahnya dengan sebuah faktor

lateks: 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 dalam ruang, kita membutuhkan tiga vektor untuk koordinat titik-titik.

Gaya-gayanya sama seperti pada plot2d dengan points=true;

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

Anda juga dapat memplot kurva dalam bentuk 3D. Dalam hal ini, akan lebih mudah untuk melakukan perhitungan awal titik-titik kurva. Untuk kurva pada 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 memplot dalam mode anaglyph. Untuk melihat plot seperti itu, Anda memerlukan kacamata merah/cyan.

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

Sering kali, skema warna spektral digunakan untuk plot. Hal ini menekankan ketinggian dari fungsi.

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

Euler juga dapat memplot permukaan yang diparameterkan, jika parameternya adalah nilai x, y, dan nilai z dari sebuah gambar grid persegi panjang di dalam ruang.

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

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

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

```
>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 batang juga dapat dilakukan. Untuk ini, kita harus menyediakan

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

z bisa lebih besar, tetapi hanya nilai nxn yang akan digunakan.
Dalam contoh, pertama-tama kita menghitung nilai. Kemudian kita sesuaikan x dan y, sehingga vektor-vektor tersebut berada di tengah-tengah 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 bagian atau lebih.

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

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

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

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

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

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

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

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

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

Selanjutnya kita atur

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

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

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

Ini memungkinkan untuk mendefinisikan fungsi numerik, yang menyelesaikan r, jika a diberikan. Dengan fungsi tersebut kita dapat memplotkan jantung yang diputar sebagai sebuah 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°):
```

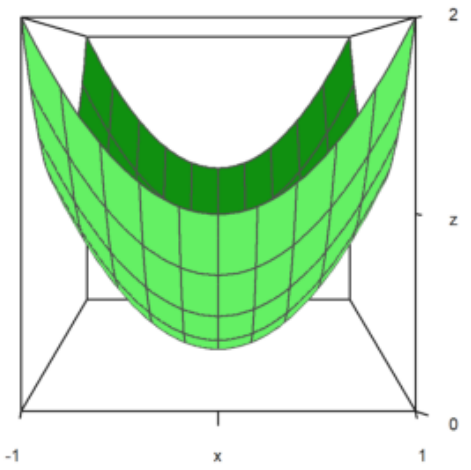
```
Variable or function hue not found.
Error in:
... t3d(r*cos(t)*sin(s),r*cos(t)*cos(s),r*sin(t), hue,<frame,color ...
                                                  ^
```

Berikut ini adalah plot 3D dari gambar di atas yang diputar di sekitar sumbu z. 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=[10,100,60],>anaglyph):
```


2
z
0
-1
x
1

Plot 3D Khusus

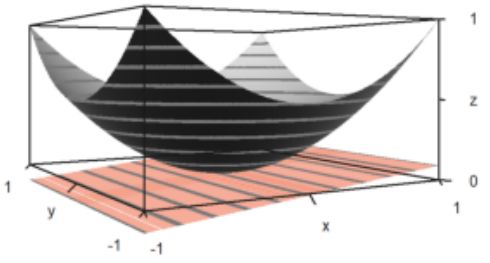
Fungsi plot3d memang bagus untuk dimiliki, tetapi tidak memenuhi semua kebutuhan. Selain rutinitas yang lebih mendasar, dimungkinkan untuk mendapatkan plot berbingkai dari objek apa pun yang Anda suka.

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

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

Sekarang framedplot() menyediakan frame, dan mengatur tampilan.

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

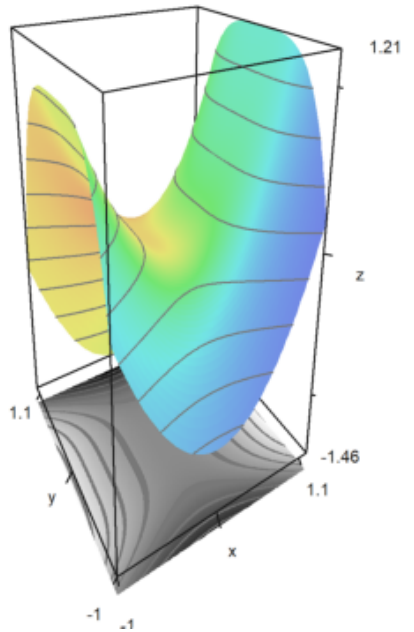


plotcontourplane() mengasumsikan hal tersebut.

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

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

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



Animasi

Euler dapat menggunakan frame untuk melakukan pra-komputasi animasi.

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

Silakan pelajari sumber rotate untuk melihat lebih detail.

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

```
Press space to stop, return to end
Press space to stop, return to end
Interrupt
animate:
    if key()==13; break; endif;
Try "trace errors" to inspect local variables after errors.
rotate:
    animate(d);
```

Menggambar Povray

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

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

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

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

Kode Povray untuk objek-objek tersebut. Untuk menggunakan fungsi-fungsi ini, mulai file Povray dengan povstart(). Kemudian gunakan writeln(...) untuk menulis objek ke file scene. Terakhir, akhiri file dengan povend(). Secara default, raytracer akan dimulai, dan file

Fungsi objek memiliki parameter yang disebut "look", yang membutuhkan sebuah string dengan kode povray untuk tekstur dan hasil akhir dari objek tersebut. Fungsi povlook() dapat digunakan untuk menghasilkan string ini. Fungsi ini memiliki parameter untuk warna, nilai transparansi, Phong Shading, dll.

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

```
>load povray;
```

ke eksekusi povray.

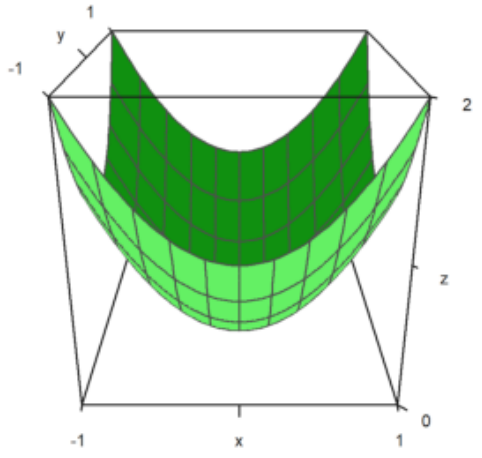
```
>defaultpovray = "C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

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

Untuk kesan pertama, kita merencanakan sebuah fungsi sederhana. Perintah berikut ini menghasilkan file povray di direktori pengguna Anda, dan menjalankan Povray untuk menelusuri berkas ini.

Jika Anda memulai perintah berikut, GUI Povray akan terbuka, menjalankan file, dan menutup secara otomatis. Karena alasan keamanan alasan keamanan, Anda akan ditanya, apakah Anda ingin mengizinkan file exe untuk dijalankan. Anda dapat menekan cancel untuk menghentikan pertanyaan lebih lanjut. Anda

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




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

```
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;
```

Kita dapat membuat fungsi menjadi transparan dan menambahkan hasil akhir lainnya. Kita bisa juga dapat menambahkan garis level pada plot fungsi.

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

```
  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;
```

Kadang-kadang perlu untuk mencegah penskalaan fungsi, dan menskalakan fungsi dengan tangan.

Kami memplot sekumpulan titik pada 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);
```

```
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;
```

Merencanakan dengan Koordinat

Alih-alih menggunakan fungsi, kita dapat membuat plot dengan koordinat. Seperti pada plot3d, kita membutuhkan tiga matriks untuk mendefinisikan objek.

Pada contoh kita memutar sebuah fungsi pada sumbu z.

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

```
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;
```

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

Kami juga menunjukkan, bagaimana sebuah objek tambahan dapat ditambahkan ke scene pov3d. Untuk pembuatan objek, lihat contoh berikut contoh. Perhatikan bahwa plot3d menskalakan plot, sehingga sesuai dengan kubus satuan.

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

```
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;
```

Dengan metode bayangan canggih Povray, sangat sedikit titik yang bisa menghasilkan permukaan yang sangat halus. Hanya pada batas-batas dan bayangan triknya mungkin menjadi jelas.

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

$$x^2 y^3$$

Persamaan permukaannya adalah [x,y,Z]. Kami menghitung dua turunannya terhadap x dan y dan mengambil hasil perkalian silang sebagai normal.

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

Kita mendefinisikan normal sebagai hasil kali silang dari turunan-turunan ini, dan mendefinisikan fungsi koordinat.

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

$$\begin{bmatrix} -2xy, & -3xy^2, & 1 \end{bmatrix}$$

Kami hanya menggunakan 25 titik.

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

% gambar% EMT4Plot3D-069.png
Berikut ini adalah simpul Trefoil yang dibuat oleh A. Busser di Povray. Ada

adalah versi yang lebih baik dari ini dalam contoh.

Lihat: Contoh\ Simpul Trefoil | Simpul Trefoil

Untuk tampilan yang bagus dengan tidak terlalu banyak titik, kita menambahkan vektor normal di sini. Kita menggunakan Maxima untuk menghitung vektor normal untuk kita. Pertama, tiga berfungsi untuk koordinat sebagai ekspresi simbolik.

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

Kemudian dua vektor turunan terhadap x dan y.

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

Sekarang normal, yang merupakan hasil perkalian silang dari dua turunan.

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

Kita sekarang mengevaluasi semua ini secara numerik.

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

Vektor normal adalah evaluasi dari ekspresi simbolik dn[i] untuk i = 1,2,3. Sintaks untuk ini adalah &"ekspresi"(parameter). Ini adalah sebuah alternatif dari metode pada contoh sebelumnya, 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));
```

Kita juga dapat membuat kisi-kisi dalam bentuk 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 dibuat.

```
>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 pov3d untuk memplot permukaan. Antarmuka povray di Euler juga dapat menghasilkan objek Povray. Objek-objek ini adalah disimpan sebagai string di Euler, dan perlu ditulis ke file Povray.

Kita memulai output dengan povstart().

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

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

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

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

String berisi kode Povray, yang tidak perlu kita pahami saat itu titik.

```
>c2
```

```
cylinder { <0,0,-1>, <0,0,1>, 1
 tekstur { pigmen { warna rgb <0,941176,0,941176,0,392157> }  }
 selesai { ambient 0.2 }
 }
```

Seperti yang Anda lihat, kami menambahkan tekstur ke objek dalam tiga warna berbeda.
Hal itu dilakukan oleh povlook(), yang mengembalikan string dengan

```
Kode povray yang relevan. Kita bisa menggunakan warna default Euler, atau mendefinisikan warna kit
```

warna. Kita juga dapat menambahkan transparansi, atau mengubah cahaya sekitar.

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

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

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

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

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

```
>povend;
```

Fungsi-fungsi berikut menghasilkan fraktal secara rekursif.
Fungsi pertama menunjukkan, bagaimana Euler menangani objek Povray sederhana. Fungsi

```
  Fungsi povbox() mengembalikan sebuah string, yang berisi koordinat kotak,
```

tekstur dan hasil akhir.

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

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

```
fractal(x+2*h,y,z+2*h,h,n-1);
fractal(x,y+2*h,z+2*h,h,n-1);
fractal(x+2*h,y+2*h,z+2*h,h,n-1);
fractal(x+h,y+h,z+h,h,n-1);
endif;
endfunction
```

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

%gambar% EMT4Plot3D-074.png

Perbedaan memungkinkan untuk memotong satu objek dari objek lainnya. Seperti persimpangan, ada bagian dari objek CSG Povray.

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

Untuk demonstrasi ini, kita mendefinisikan sebuah objek di Povray, alih-alih menggunakan sebuah string di Euler. Pendefinisian akan dituliskan ke file dengan segera.

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

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

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

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

Kita membuat kubus kedua, dan memutar dan menskalakannya sedikit.

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

Kemudian kita ambil selisih dari kedua objek tersebut.

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

Sekarang tambahkan tiga sumbu.

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

Fungsi Implisit

Povray dapat memplot himpunan di mana f(x,y,z)=0, seperti parameter implisit di plot3d. Hasilnya terlihat jauh lebih baik, Namun.

Sintaks untuk fungsi-fungsi tersebut sedikit berbeda. Anda tidak dapat 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)-po
>povend();
```

```
Kesalahan : Kesalahan povray!

Kesalahan yang dihasilkan oleh perintah error()

povray:
    error("Povray error!");
Coba "trace errors" untuk memeriksa variabel lokal setelah terjadi kesalahan.
povend:
    povray(file,w,h,aspect,exit);
```

```
>povstart(angle=25°,height=10°);
>writeln(povsurface("pow(x,2)+pow(y,2)*pow(z,2)-1",povlook(blue),povbox(-2,2,"")));
>povend();
>povstart(angle=70°,height=50°,zoom=4);
```

Membuat permukaan implisit. Perhatikan sintaks yang berbeda pada ekspresi ekspresi.

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

Objek Jaring

Pada contoh ini, kami menunjukkan cara membuat objek mesh, dan menggambarnya dengan informasi tambahan.

Kita ingin memaksimalkan xy pada kondisi x+y=1 dan mendemonstrasikan persinggungan tangensial dari garis level.

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

Kita tidak dapat menyimpan objek dalam sebuah string seperti sebelumnya, karena ukurannya terlalu besar. Jadi kita mendefinisikan objek dalam file Povray menggunakan deklarasikan. Fungsi povtriangle() melakukan hal ini secara otomatis. Fungsi ini dapat menerima vektor normal seperti halnya pov3d().

Berikut ini mendefinisikan objek mesh, dan langsung 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 buah cakram, yang akan berpotongan dengan permukaan.

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

Tuliskan permukaan dikurangi dengan dua cakram.

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

Tuliskan dua persimpangan.

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

Menuliskan sebuah titik secara maksimal.

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

Tambahkan sumbu dan selesaikan.

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

Anaglyph dalam Povray

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

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

Fungsi pov3d() memiliki sebuah saklar sederhana untuk menghasilkan anaglyph.

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

%gambar% EMT4Plot3D-080.png

Jika Anda membuat adegan dengan objek, Anda perlu memasukkan pembuatan adegan ke dalam sebuah fungsi, dan jalankan 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. Parameter-parameternya adalah seperti pada povstart() dan povend() digabungkan.

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

Mendefinisikan Objek sendiri

Antarmuka povray dari Euler berisi banyak objek. Tetapi Anda % Anda tidak terbatas pada objek-objek tersebut. Anda dapat membuat objek sendiri, yang menggabungkan objek lain, atau objek yang benar-benar baru.

Kami mendemonstrasikan sebuah torus. Perintah Povray untuk ini adalah "torus". Jadi kami mengembalikan sebuah string dengan perintah ini dan parameternya. Perhatikan bahwa torus selalu berpusat pada titik asal.

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

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

Inilah torus pertama kita.

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

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

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

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

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

Sekarang kita tempatkan objek-objek ini ke dalam sebuah scene. 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();

memanggil program Povray. Namun, jika terjadi kesalahan, program ini tidak menampilkan kesalahan. Oleh karena itu, Anda harus menggunakan

>povend(<keluar);

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

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

```
Fungsi povstart tidak ditemukan.
Daftar coba ... untuk menemukan fungsi!
Kesalahan dalam:
povstart(center=[0.4,0,0],angle=0Â°,zoom=3.8,aspect=1.5); writeln(povobject(t1,povlook(green,phong
                                                                ^
Berikut adalah contoh yang lebih rumit. Kami menyelesaikan
```

lateks: Ax \le b, \quad x \ge 0, \quad c.x \to \text{Max.} % gambar% EMT4Plot3D-082.png
dan menunjukkan titik-titik yang layak dan optimal dalam plot 3D.

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

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

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

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

Ya, ada.

Selanjutnya kita mendefinisikan dua objek. Yang pertama adalah bidang

lateks: a \cdot x \le b % gambar% EMT4Plot3D-083.png

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

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

Kemudian kita mendefinisikan perpotongan dari semua setengah spasi dan sebuah 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 memplot adegan.

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

Berikut ini adalah sebuah lingkaran di sekitar titik optimal.

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

Dan kesalahan ke arah optimum.

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

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

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

% gambar% EMT4Plot3D-084.png

Lebih Banyak Contoh

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

Lihat: Examples/Dandelin Spheres
Lihat: Contoh/Contoh/Donat Matematika
Lihat: Contoh/Simpul Trefoil
Lihat: Contoh/Contoh/Optimalisasi dengan Penskalaan Affine