

Nama : Nandhita Putri Shalsabila  
Nim : 23030630075  
Kelas : Matematika B 2023

## Menggambar Plot 3D dengan EMT

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

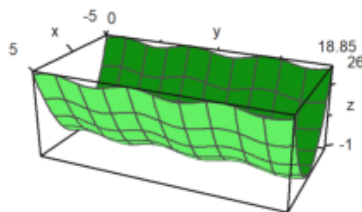
Euler menggambar fungsi-fungsi tersebut dengan menggunakan algoritme pengurutan untuk menyembunyikan bagian-bagian di latar belakang. Secara umum, Euler menggunakan proyeksi pusat. Standarnya adalah dari kuadran x-y positif ke arah asal  $x=y=z=0$ , tetapi sudut  $=0^\circ$  terlihat dari arah sumbu-y. Sudut pandang dan ketinggian dapat diubah.

Euler dapat memetakan

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

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

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



aspect(1.5);

untuk mengatur rasio aspek dari plot. Dalam hal ini, aspek ditetapkan menjadi 1.5, yang berarti rasio antara sumbu-sumbu grafik akan disesuaikan agar proporsional.

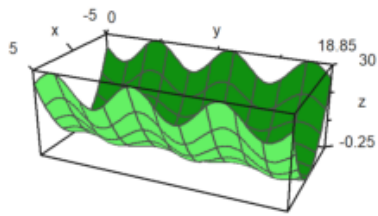
plot3d("x^2+sin(y)", -5, 5, 0, 6\*pi); untuk membuat grafik 3D dari fungsi  $z=x^2 + \sin(y)$ , di mana:

$x^2 + \sin(y)$  adalah fungsi yang dipetakan ke sumbu z berdasarkan nilai x dan y.

-5 dan 5 adalah batas-batas untuk nilai x (dari -5 sampai 5).

0 dan 6\*pi adalah batas-batas untuk nilai y (dari 0 hingga 6pi).

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



`plot3d("x^2+x*sin(y)", -5, 5, 0, 6*pi);` untuk menghasilkan grafik 3D dari fungsi  $z = x^2 + x \sin(y)$ , dengan:  
 Fungsi  $x^2 + x \sin(y)$  yang diplot di sumbu z.  
 Batas nilai x dari -5 hingga 5.  
 Batas nilai y dari 0 hingga  $6\pi$ .

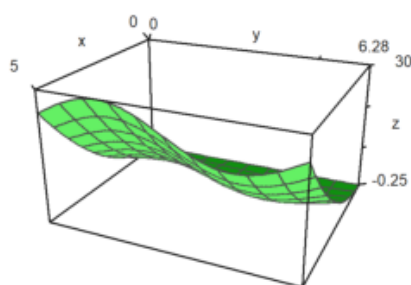
## Latihan Soal

---

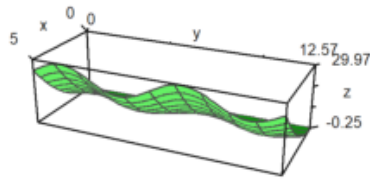
Silakan lakukan modifikasi agar gambar "talang bergelombang" tersebut tidak lurus melainkan melengkung/melingkar, baik melingkar secara mendatar maupun melingkar turun/naik (seperti papan peluncur pada kolam renang).  
 Temukan rumusnya.

1. talang bergelombang melengkung secara mendatar (koordinat polar)

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



```
>plot3d("x^2+x*sin(y)", 0, 5, 0, 4*pi):
```



## Fungsi dari dua Variabel

---

Untuk grafik sebuah fungsi, gunakan

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

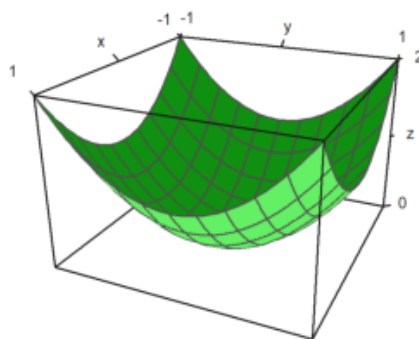
Standarnya adalah kisi-kisi kawat yang terisi dengan warna yang berbeda di kedua sisi. Perhatikan bahwa jumlah default interval grid adalah 10, namun plot menggunakan jumlah default 40x40 persegi panjang untuk membangun permukaan. Hal ini dapat diubah.

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

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

Kami menggunakan default n=40 dan grid=10.

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



Fungsi  $z=x^2 + y^2$  : fungsi kuadratik yang menghasilkan permukaan berbentuk paraboloid. Nilai z akan meningkat secara simetris karena  $x^2$  dan  $y^2$  selalu bernilai positif atau nol.

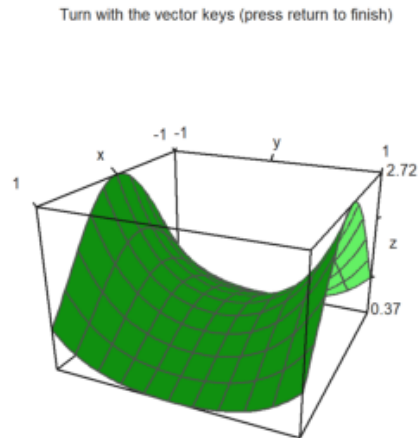
Plot3d: menghasilkan grafik 3D dari fungsi yang diberikan dalam rentang default untuk x dan y (biasanya dari -5 hingga 5 jika tidak ditentukan).

Interaksi pengguna dapat dilakukan dengan >user Parameter. Pengguna dapat menekan 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
- kembali: mengakhiri interaksi

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



Rentang plot untuk fungsi dapat ditentukan dengan

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

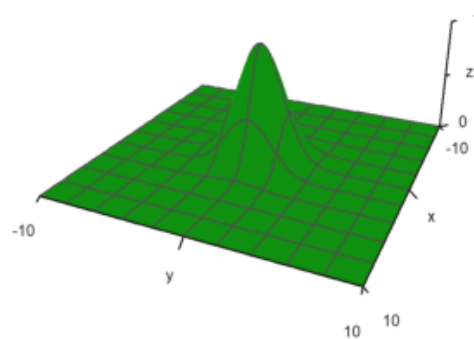
Ada beberapa parameter untuk menskalakan fungsi atau mengubah tampilan grafik.

fscale: menskalakan ke nilai fungsi (defaultnya adalah <fscale>).

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

bingkai: jenis bingkai (default 1).

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



Perintah ini menghasilkan grafik 3D dari fungsi Gaussian berbentuk lonceng dengan distribusi yang diperlebar, rentang dari -10 hingga 10 pada sumbu x dan y, dan dengan resolusi tinggi (80 titik per sumbu). Grafik ini juga diskalakan pada sumbu vertikal (fscale 4) dan keseluruhan grafik diperbesar 1.2 kali.

Tampilan dapat diubah dengan berbagai cara.

- Jarak: jarak pandang ke plot.
- zoom: nilai zoom.
- sudut: sudut terhadap sumbu y negatif dalam radian.
- tinggi: ketinggian pandangan dalam radian.

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

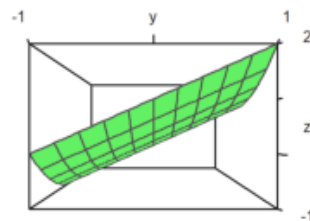
```
>view
```

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

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

Pada contoh berikut, `sudut=0` dan `tinggi=0` dilihat dari sumbu y negatif. Label sumbu untuk y disembunyikan dalam kasus ini.

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



Perintah ini menghasilkan grafik 3D dari fungsi  $z = x^2 + y$  dengan sudut pandang khusus:

Kamera berada pada jarak 3 unit dari grafik.

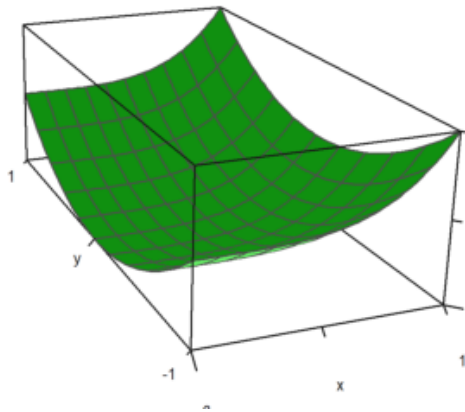
Grafik ditampilkan tanpa pembesaran (zoom 1).

Kamera diputar 90 derajat (sudut  $\pi/2$ ), sehingga tampilan grafik lebih dari samping.

Kamera ditempatkan sejajar dengan bidang horizontal (ketinggian 0).

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

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



Perintah ini menghasilkan grafik 3D dari fungsi  $z=x^4 + y^2$  dengan aturan sebagai berikut:

Rentang nilai  $x$  dari 0 hingga 1 dan nilai  $y$  dari -1 hingga 1.

Sudut pandang sedikit diputar (-20 derajat) dan kamera diposisikan 20 derajat di atas bidang horizontal.

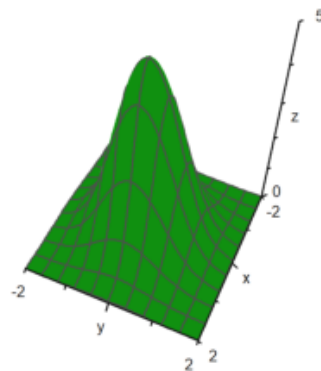
Grafik berpusat di titik (0.4, 0, 0).

Grafik diperbesar lima kali dari ukuran normal.

Plotnya diskalakan agar sesuai dengan unit kubus untuk dilihat. Jadi tidak perlu mengubah jarak atau zoom tergantung ukuran plot. Namun labelnya mengacu pada ukuran sebenarnya.

Jika Anda memamatkannya dengan `scale=false`, Anda harus berhati-hati agar plot tetap masuk ke dalam jendela plotting, dengan mengubah jarak pandang atau zoom, dan memindahkan bagian tengah.

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



Perintah ini menghasilkan grafik 3D dari fungsi  $z=5.\exp(-x^2-y^2)$  dengan aturan sebagai berikut:

Rentang nilai  $x$  dan  $y$  dari -2 hingga 2.

Kamera diposisikan pada jarak 13 unit dari grafik dan pada ketinggian 50 derajat, memberikan sudut pandang yang baik.

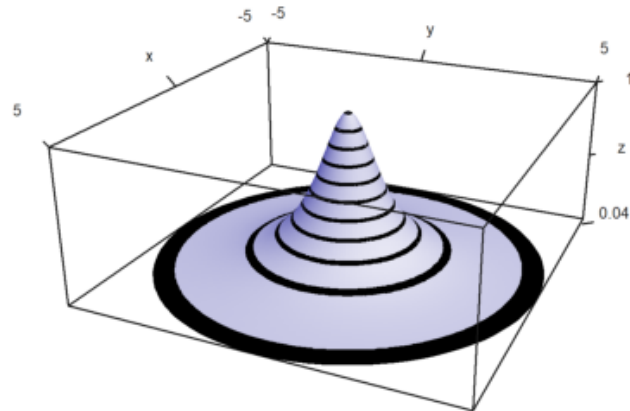
Grafik berpusat di titik (0, 0, -2).

Skala fungsi dan skala keseluruhan grafik menggunakan nilai default.

Tampilan dengan kerangka atau grid 3D yang membantu visualisasi.

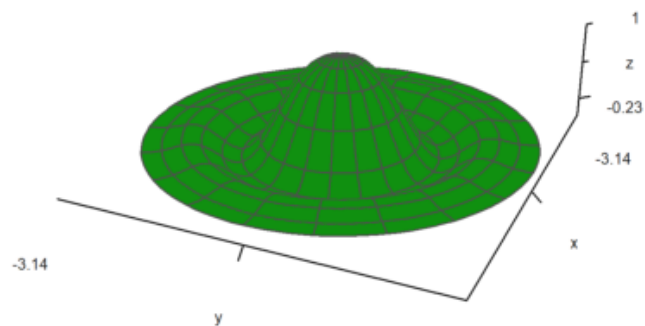
Plot kutub juga tersedia. Parameter `polar=true` menggambar plot kutub. Fungsi tersebut harus tetap merupakan fungsi dari  $x$  dan  $y$ . Parameter "`fscale`" menskalakan fungsi dengan skalanya sendiri. Kalau tidak, fungsinya akan diskalakan agar sesuai dengan kubus.

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



Perintah ini menghasilkan grafik 3D dari fungsi  $z=1/(x^2 + y^2 + 1)$  dengan aturan sebagai berikut:  
 Rentang nilai x dan y dari -5 hingga 5.  
 Grafik ditampilkan dalam koordinat polar.  
 Skala vertikal diperbesar dua kali (fscale=2).  
 Warna permukaan berdasarkan hue (nuansa).  
 100 titik digunakan untuk resolusi yang lebih halus.  
 Grafik diperbesar empat kali dari ukuran normal.  
 Garis kontur juga ditampilkan pada grafik.  
 Warna grafik ditetapkan ke biru.

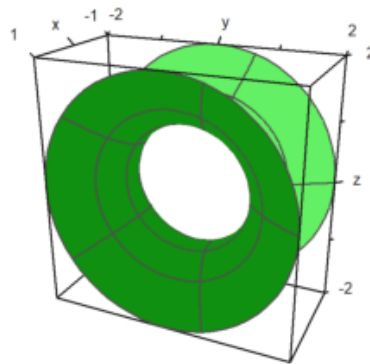
```
>function f(r) := exp(-r/2)*cos(r); ...
>plot3d("f(x^2+y^2)",>polar,scale=[1,1,0.4],r=pi,frame=3,zoom=4):
```



Perintah ini menghasilkan grafik 3D dari fungsi  $f(r)=e^{-r/2} \cdot \cos(r)$  dengan aturan sebagai berikut:  
 Fungsi z ditentukan dengan menggunakan  $r=x^2 + y^2$

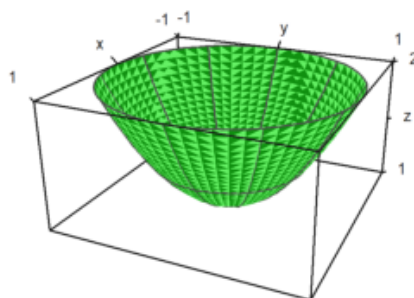
Grafik ditampilkan dalam koordinat polar, sehingga lebih intuitif untuk fungsi radial.  
 Skala pada sumbu z dikurangi menjadi 0.4 untuk membuat tampilan lebih datar.  
 Rentang nilai r dibatasi hingga pi.  
 Tampilan dengan kerangka 3D yang lebih terperinci.  
 Grafik diperbesar empat kali untuk detail yang lebih baik.  
 Parameter memutar memutar fungsi di x di sekitar sumbu x.  
 - putar=1: Menggunakan sumbu x  
 - putar=2: Menggunakan sumbu z

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



plot3d: digunakan untuk menggambar grafik 3D.  
 "x^2+1": ekspresi fungsi matematika yang akan digambar, yaitu  $y = x^2 + 1$ . Fungsi ini menggambarkan parabola yang digeser satu unit ke atas.  
 a=-1, b=1: batas rentang x. Dalam hal ini, x dibatasi dari -1 hingga 1.  
 rotate=true: Opsi ini memungkinkan grafik berputar sehingga pengguna bisa melihatnya dari berbagai sudut.  
 Grid=5: mengatur jumlah grid atau partisi pada sumbu untuk membantu visualisasi. Nilai 5 menunjukkan grafik akan dibagi menjadi 5 bagian pada setiap sumbu.

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





plot3d: digunakan untuk menggambar grafik 3D.

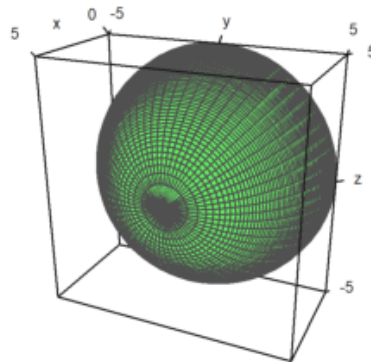
" $x^2+1$ ": ekspresi fungsi matematika yang akan digambar, yaitu  $y=x^2+1$ , sebuah parabola yang digeser satu unit ke atas.

$a=-1, b=1$ : rentang untuk variabel  $x$ , yang dibatasi dari -1 hingga 1.

$rotate=2$ : Nilai  $rotate=2$  menentukan rotasi awal grafik 3D pada tampilan. Berbeda dengan  $rotate=true$  yang memungkinkan rotasi interaktif,  $rotate=2$  merujuk pada sudut atau posisi tetap rotasi tertentu yang telah ditentukan.

$grid=5$ : Mengatur jumlah grid atau partisi pada sumbu untuk membantu visualisasi. Grid akan dibagi menjadi 5 bagian per sumbu, memberikan tampilan grafik dengan detail yang lebih sederhana.

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



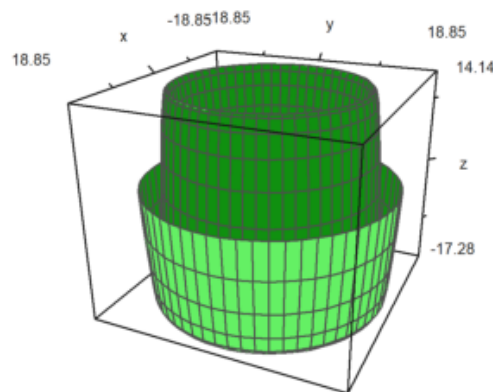
plot3d: untuk menggambar grafik 3D.

$\sqrt{25-x^2}$ : fungsi matematika yang digambarkan, yaitu  $y = \sqrt{25-x^2}$ . Fungsi ini adalah setengah lingkaran dengan jari-jari 5 pada bidang xy, didefinisikan hanya untuk  $x$  dari -5 sampai 5 karena nilai dalam akar kuadrat harus positif.

$a=0, b=5$ : menentukan rentang  $x$  dari 0 hingga 5. Karena fungsi ini merepresentasikan setengah lingkaran, rentang  $x$  hanya setengah dari lingkaran tersebut (bagian positif).

$rotate=1$ : menentukan rotasi awal grafik 3D, memberikan sudut pandang tertentu pada grafik saat ditampilkan.

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



plot3d: digunakan untuk menggambar grafik 3D.

"x\*sin(x)": ekspresi fungsi matematika yang akan digambar, yaitu

$y = x \sin(x)$ . Fungsi ini menghasilkan grafik gelombang sinus yang amplitudonya berubah seiring dengan bertambahnya nilai  $x$ .

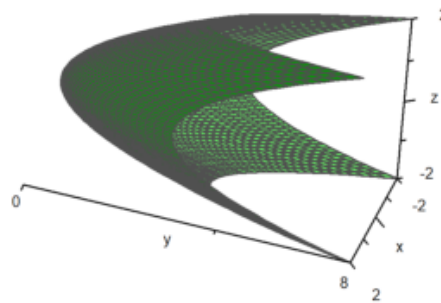
$a=0$ ,  $b=6\pi$ : rentang untuk variabel  $x$ . Nilai  $x$  diatur dari 0 hingga

$6\pi$  (sekitar 18.85), yang mencakup beberapa siklus dari fungsi sinus.

$rotate=2$ : Parameter ini menentukan rotasi awal grafik dalam tampilan 3D. Nilai  $rotate=2$  berarti grafik akan dimulai dengan sudut atau posisi rotasi tertentu yang telah ditetapkan.

Berikut adalah plot dengan tiga fungsi.

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



plot3d: digunakan untuk menggambar grafik 3D.

"x", "x^2+y^2", "y": tiga ekspresi fungsi matematika yang merepresentasikan tiga dimensi (sumbu  $x$ ,  $z$ , dan  $y$ ) dari grafik 3D.

$x$  berfungsi sebagai sumbu  $x$ ,

$x^2 + y^2$  menggambarkan ketinggian (sumbu  $z$ , biasanya terkait dengan fungsi atau permukaan),

$y$  merepresentasikan sumbu  $y$ .

Secara keseluruhan, ini menggambarkan permukaan 3D yang berbentuk paraboloid (karena  $x^2 + y^2$  adalah persamaan paraboloid di ruang 3D).

$r=2$ : parameter untuk mengatur radius dari grid yang dipetakan dalam grafik, sehingga grafik akan diplot dalam rentang dengan jari-jari 2 unit.

$zoom=3.5$ : mengatur tingkat pembesaran (zoom) pada grafik. Nilai ini memperbesar atau memperkecil tampilan grafik agar lebih jelas.

$frame=3$ : Parameter ini menentukan tipe bingkai yang akan digunakan untuk grafik. Nilai  $frame=3$  mungkin menunjukkan pengaturan tertentu untuk bingkai grafik, seperti bagaimana sumbu-sumbu ditampilkan atau bagaimana grid ditata di sekitar grafik.

## Plot Kontur

Untuk plotnya, Euler menambahkan garis grid. Sebaliknya, dimungkinkan untuk menggunakan garis datar dan rona satu warna atau rona warna spektral. Euler dapat menggambar ketinggian fungsi pada plot dengan arsiran. Di semua plot 3D, Euler dapat menghasilkan anaglyph merah/cyan.

->hue: Mengaktifkan bayangan cahaya, bukan kabel.

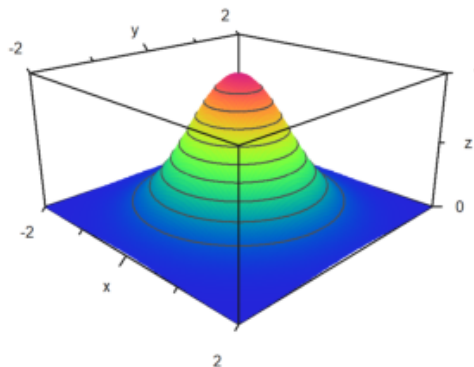
->kontur: Membuat plot garis kontur otomatis pada plot.

- level=... (atau level): Vektor nilai garis kontur.

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

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

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



plot3d: digunakan untuk menggambar grafik 3D.

"exp(-x<sup>2</sup>-y<sup>2</sup>)": ekspresi fungsi matematika yang akan digambar. Fungsi ini adalah fungsi Gaussian 2D (fungsi lonceng) yang memiliki puncak di (0,0) dan menurun dengan cepat menuju nol saat x<sup>2</sup> + y<sup>2</sup> bertambah. Grafik ini menggambarkan permukaan yang simetris berbentuk lonceng.

r=2: mengatur rentang x dan y dari -2 hingga 2. Grafik akan diplot dalam lingkup jari-jari 2 unit pada sumbu x dan y.

n=100: Ini menentukan resolusi grid, dengan 100 titik di sepanjang setiap sumbu. Semakin tinggi nilainya, semakin halus grafik yang dihasilkan.

level="thin": Ini mengatur tampilan level (kontur) yang lebih tipis pada grafik, sehingga garis kontur yang dihasilkan lebih halus dan lebih presisi.

>contour: menambahkan garis kontur pada grafik 3D. Garis-garis kontur digunakan untuk menunjukkan nilai-nilai yang sama pada permukaan grafik, memberikan informasi tambahan tentang bentuk fungsi pada bidang tertentu.

>spectral: mengatur spektrum warna untuk grafik, yang berarti warna yang digunakan akan didasarkan pada skala spektral, biasanya warna pelangi atau gradasi warna lainnya.

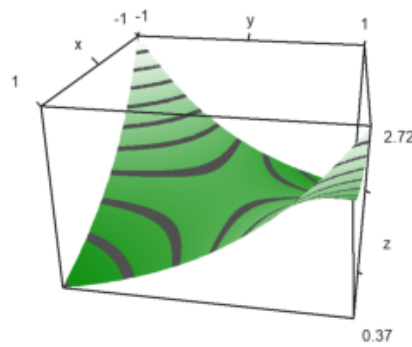
fscale=1: mengatur faktor skala untuk kontur dan grafik ke nilai standar (tanpa pembesaran atau pengecilan tambahan).

scale=1.1: memperbesar grafik secara keseluruhan sebesar 1.1 kali lipat, sehingga grafik terlihat sedikit lebih besar dari ukuran aslinya.

angle=45°: mengatur sudut rotasi tampilan grafik 3D sebesar 45 derajat di sekitar sumbu vertikal (sumbu z), sehingga grafik akan terlihat dari sudut miring.

height=20°: Ini mengatur sudut elevasi atau ketinggian dari mana grafik dilihat. Dalam hal ini, grafik akan dilihat dari sudut 20 derajat di atas bidang horizontal.

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



plot3d: digunakan untuk menggambar grafik 3D.

"exp(x\*y)": ekspresi fungsi matematika yang akan digambar, yaitu  $z = \exp(x \cdot y)$ . Fungsi ini eksponensial di mana variabel

x dan y saling dikalikan. Grafik ini akan menunjukkan peningkatan yang cepat di satu arah dan penurunan di arah yang lain, membentuk permukaan yang cukup curam di beberapa area.

angle=100°: mengatur sudut rotasi tampilan grafik 3D sebesar 100 derajat di sekitar sumbu vertikal (z). Grafik akan diputar untuk dilihat dari sudut ini, memungkinkan tampilan yang berbeda dari grafik.

>contour: Perintah ini menambahkan garis kontur pada grafik 3D. Garis kontur digunakan untuk menunjukkan nilai-nilai yang sama pada permukaan grafik, yang membantu dalam memahami variasi elevasi atau perubahan fungsi dalam bidang tertentu.

color=green: Ini mengatur warna grafik menjadi hijau. Grafik dan kontur akan ditampilkan menggunakan skema warna hijau.

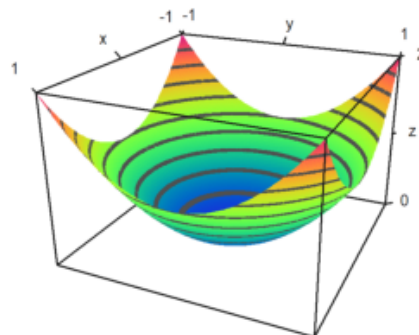
Bayangan defaultnya menggunakan warna abu-abu. Namun rentang warna spektral juga tersedia.

->spektral: Menggunakan skema spektral default

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

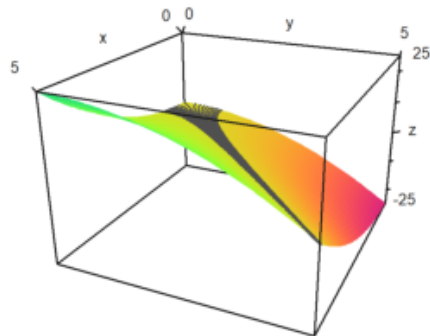
Untuk plot berikut, kami menggunakan skema spektral default dan menambah jumlah titik untuk mendapatkan tampilan yang sangat mulus.

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



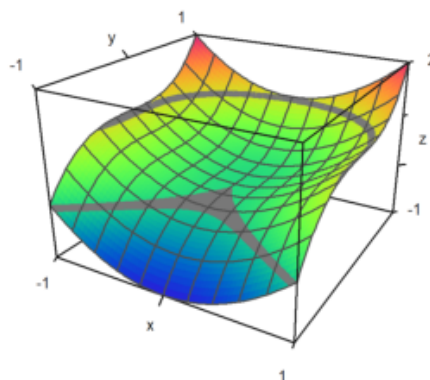
Selain garis level otomatis, kita juga dapat menetapkan nilai garis level. Ini akan menghasilkan garis level yang tipis, bukan rentang level.

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



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

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

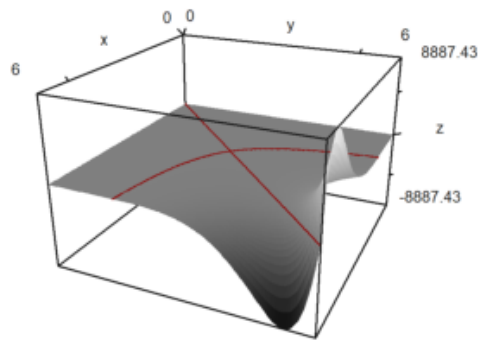


Pada contoh berikut, kita memplot himpunan, di mana

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

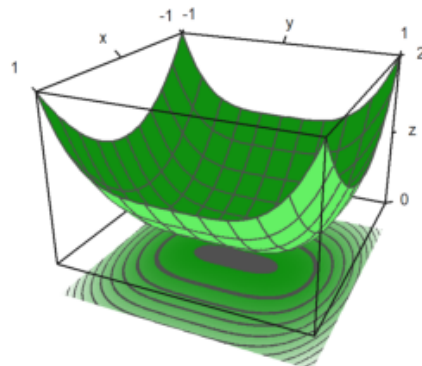
Kami menggunakan satu garis tipis untuk garis level.

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



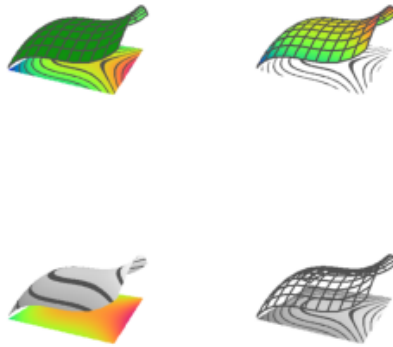
Dimungkinkan untuk menampilkan bidang kontur di bawah plot. Warna dan jarak ke plot dapat ditentukan.

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



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

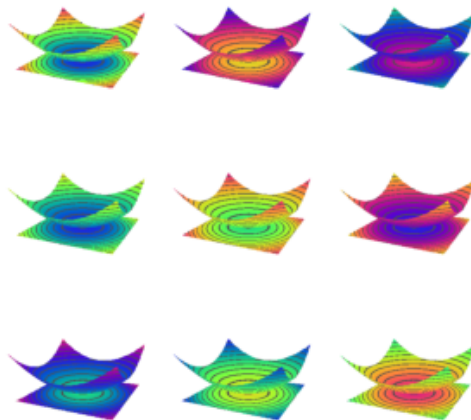
```
>figure(2,2); ...
>expr="y^3-x^2"; ...
>figure(1); ...
> plot3d(expr,<frame,>cp,cpcolor=spectral); ...
>figure(2); ...
> plot3d(expr,<frame,>spectral,grid=10,cp=2); ...
>figure(3); ...
> plot3d(expr,<frame,>contour,color=gray,nc=5,cp=3,cpcolor=greenred); ...
>figure(4); ...
> plot3d(expr,<frame,>hue,grid=10,>transparent,>cp,cpcolor=gray);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



Ada beberapa skema spektral lainnya, yang diberi nomor dari 1 hingga 9. Namun Anda juga dapat menggunakan warna=nilai, di mana nilai

- spektral: untuk rentang dari biru hingga merah
- putih: untuk rentang yang lebih redup
- kuningbiru, unguhijau, birukuning, hijaumerah
- birukuning, hijauungu, kuningbiru, merahhijau

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

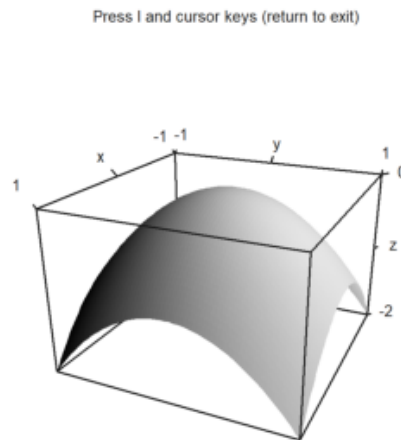


Sumber cahaya dapat diubah dengan l dan tombol kursor selama interaksi pengguna. Itu juga dapat diatur dengan parameter.

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

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

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



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

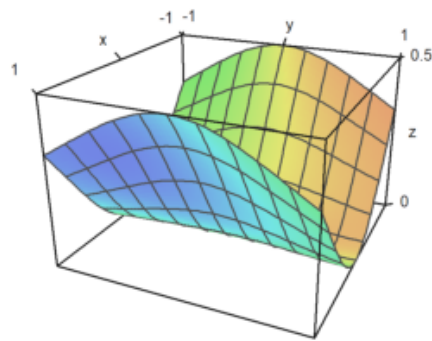
```
>plot3d("-x^2-y^2",color=rgb(0.2,0.2,0),hue=true,frame=false, ...
> zoom=3,contourcolor=red,level=-2:0.1:1,d1=0.01):
```



Warna 0 memberikan efek pelangi yang istimewa.

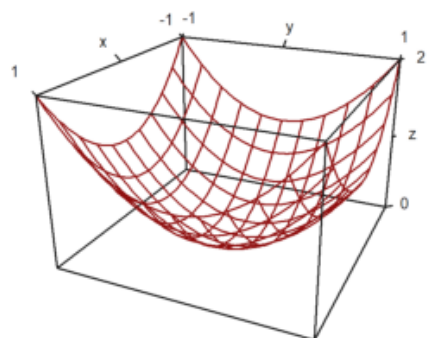
```
>plot3d("x^2/(x^2+y^2+1)",color=0,hue=true,grid=10):
```





Permukaannya juga bisa transparan.

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



## Plot Implisit

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

Solusi dari

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

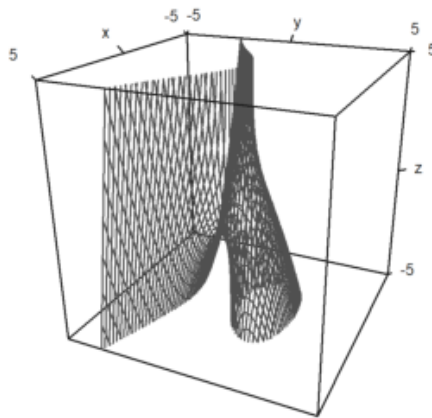
dapat divisualisasikan dalam potongan yang sejajar dengan bidang x-y-, x-z- dan y-z.

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

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

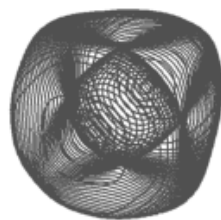
$$M = \{(x, y, z) : x^2 + y^3 + zy = 1\}$$

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



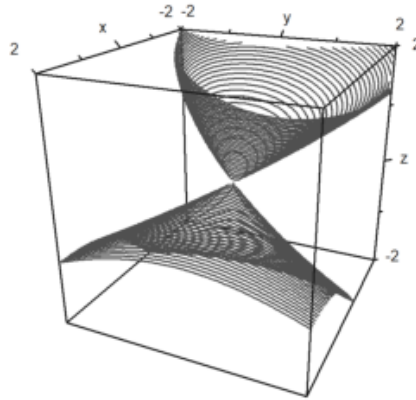
```
>c=1; d=1;
```

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



Perintah ini menggambarkan grafik 3D dari persamaan implisit yang cukup kompleks, yang melibatkan fungsi-fungsi kuadrat dan konstanta c dan d. Rentang plot dibatasi dari -2 hingga 2 di setiap sumbu, dan grafik ini ditampilkan dengan bingkai (frame) sumbu koordinat. Persamaan ini menghasilkan bentuk geometris yang rumit dalam ruang 3D yang mungkin menyerupai permukaan non-linear.

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



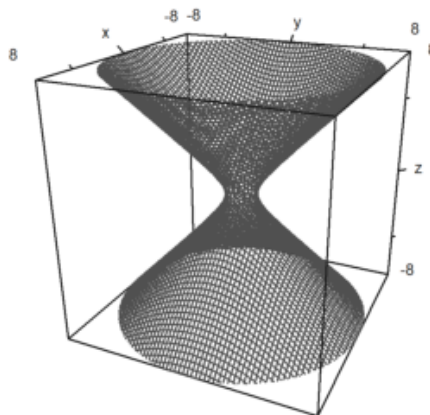
Perintah ini menggambarkan grafik tiga dimensi dari persamaan implisit  $x^2 + y^2 + 4xz + z^3$ , di mana grafik diplot dalam rentang dari -2 hingga 2 pada setiap sumbu, dengan tingkat pembesaran 2.5 kali. Grafik ini akan menunjukkan bentuk geometris non-linear yang kompleks dalam ruang 3D sesuai dengan persamaan tersebut.

## Latihan soal

1. gambarlah fungsi implisit berikut dalam 3D

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

```
>plot3d("x^2+y^2-z^2-1",r=8,implicit=3):
```

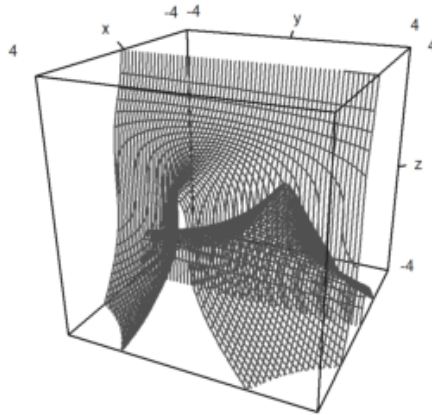


2. gambarlah fungsi 3D dari fungsi implisit berikut ini

$$f(x, y, z) = xy + x^3y^2 + xz^3 - 9 = 0$$

dengan  $r=4$

```
>plot3d("x*y+x^3*y^2+x*z^3-9", r=4, implicit=3):
```



## Merencanakan Data 3D

Sama seperti `plot2d`, `plot3d` menerima data. Untuk objek 3D, Anda perlu menyediakan matriks nilai  $x$ -,  $y$ - dan  $z$ , atau tiga fungsi atau ekspresi  $f_x(x,y)$ ,  $f_y(x,y)$ ,  $f_z(x,y)$ .

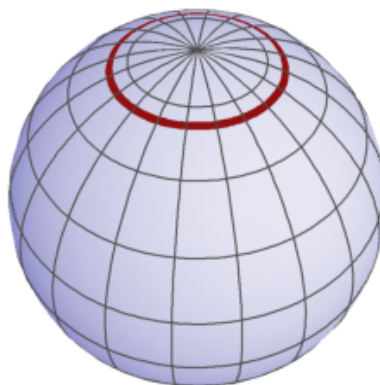
$$\gamma(t, s) = (x(t, s), y(t, s), z(t, s))$$

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

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

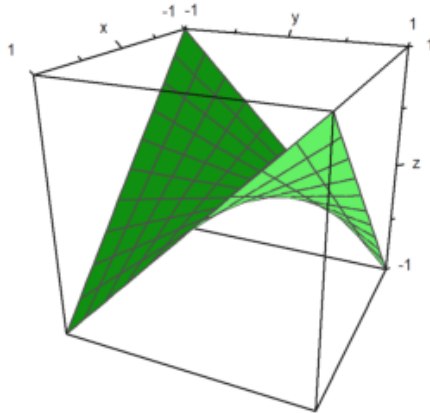
Dalam contoh berikut, kita menggunakan vektor nilai  $t$  dan vektor kolom nilai  $s$  untuk membuat parameter permukaan bola. Dalam gambar kita dapat menandai wilayah, dalam kasus kita wilayah kutub.

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



Berikut ini contohnya yaitu grafik suatu fungsi.

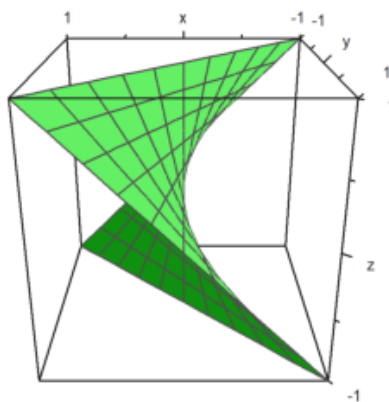
```
>t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,t*s,grid=10):
```



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

$$x = yz$$

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



Dengan lebih banyak usaha, kita dapat menghasilkan banyak permukaan.

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

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

dengan

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

Kami mendistorsi ini dengan sebuah faktor

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

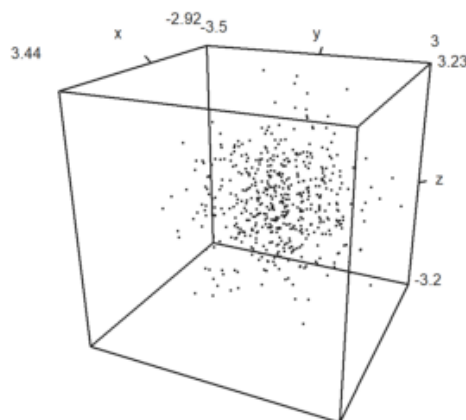
```
>t=linspace(0,2pi,320); s=linspace(-pi/2,pi/2,160)'; ...  
>d=1+0.2*(cos(4*t)+cos(8*s)); ...  
>plot3d(cos(t)*cos(s)*d,sin(t)*cos(s)*d,sin(s)*d,hue=1, ...  
> light=[1,0,1],frame=0,zoom=5):
```



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

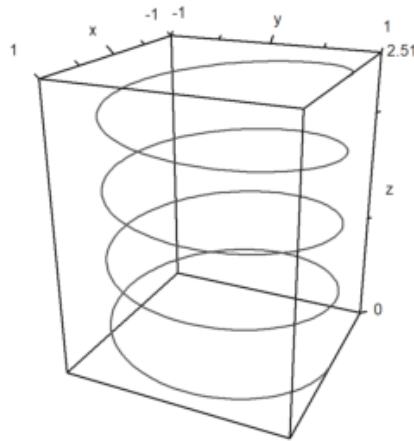
Gayanya sama seperti di plot2d dengan points=true;

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



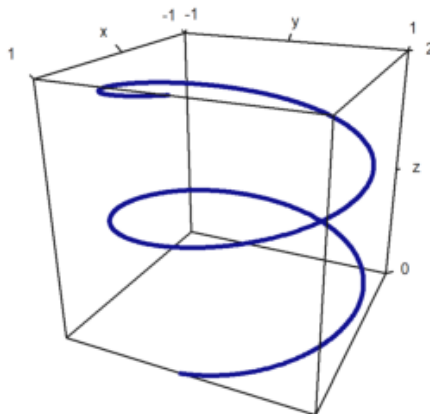
Dimungkinkan juga untuk memplot kurva dalam 3D. Dalam hal ini, lebih mudah untuk menghitung terlebih dahulu titik-titik kurva. Untuk kurva pada bidang kita menggunakan barisan koordinat dan parameter wire=true.

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



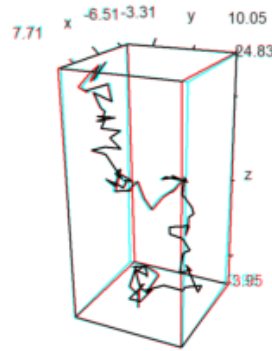
Perintah ini menggambar spiral 3D menggunakan fungsi sinus dan cosinus dengan variasi di sumbu  $x$ ,  $y$ , dan  $z$ . Grafiknya diplot sebagai kurva wireframe (tanpa permukaan) dengan zoom sebesar 3 kali. Ini menghasilkan grafik yang menyerupai heliks atau spiral, karena nilai  $\sin(t)$  dan  $\cos(t)$  membentuk lingkaran dalam ruang  $x$ - $y$ , sementara  $t/10$  secara linear meningkat pada sumbu  $z$ .

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



Perintah ini menggambar spiral 3D dalam ruang menggunakan fungsi cosinus dan sinus untuk menghasilkan kurva melingkar dalam bidang  $x$ - $y$ , sementara variabel  $t/2\pi$  memberikan pergerakan linier pada sumbu  $z$ . Grafik diplot sebagai kurva wireframe dengan ketebalan garis 3 dan warna biru, sehingga menghasilkan spiral biru yang tebal dalam ruang 3D.

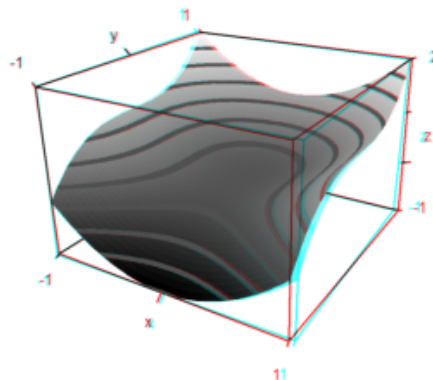
```
>X=cumsum(normal(3,100)); ...
> plot3d(X[1],X[2],X[3],>anaglyph,>wire):
```



Perintah ini menggambar grafik 3D dari jalur acak yang dihasilkan berdasarkan distribusi normal dalam ruang 3D, diplot sebagai kurva wireframe (hanya garis) dengan efek anaglyph untuk memberikan pengalaman visual 3D. Jalur ini dihasilkan melalui jumlah kumulatif dari vektor-vektor acak, yang menghasilkan jalur yang terus terhubung tetapi acak.

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

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

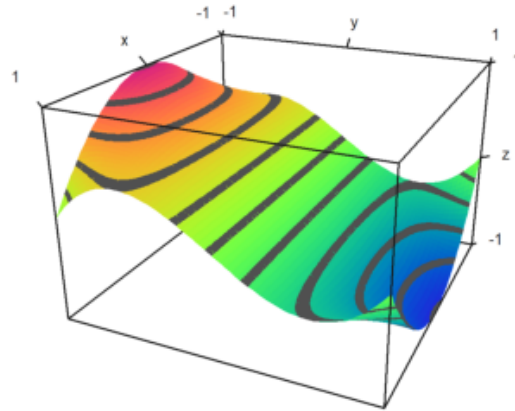


Perintah ini digunakan untuk menggambar grafik 3D dari fungsi  $x^2 + y^3$ , dengan tampilan dalam mode anaglyph (untuk efek 3D), menampilkan garis kontur untuk menunjukkan level ketinggian pada permukaan, dan dengan sudut pandang 30 derajat. Hasilnya adalah tampilan 3D yang memungkinkan pengguna melihat bentuk permukaan dan variasi ketinggian melalui garis-garis kontur.

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

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





Perintah ini menggambar grafik 3D dari fungsi  $x^2y^3 - y$  dalam ruang tiga dimensi, dengan tampilan warna spektral yang mewakili ketinggian pada permukaan, serta garis kontur untuk membantu visualisasi level ketinggian. Pembesaran (zoom) diatur pada 3.2 kali lipat, memungkinkan tampilan lebih dekat pada grafik. Hasilnya adalah grafik permukaan non-linear yang ditampilkan secara visual dengan warna dan kontur untuk menunjukkan perbedaan ketinggian di permukaan.

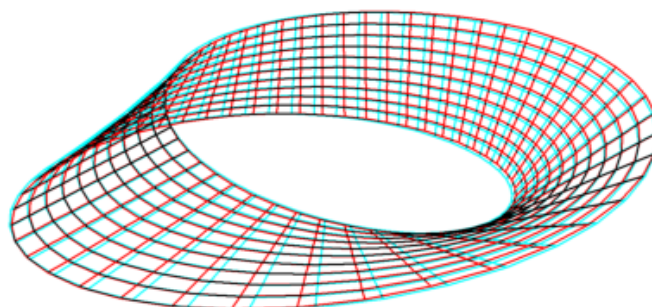
Euler juga dapat memplot permukaan yang diparameterisasi, jika parameternya adalah nilai  $x$ ,  $y$ , dan  $z$  dari gambar kotak persegi panjang di ruang tersebut.

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

```
>u=linspace(-1,1,10); v=linspace(0,2*pi,50)'; ...
>X=(3+u*cos(v/2))*cos(v); Y=(3+u*cos(v/2))*sin(v); Z=u*sin(v/2); ...
```

Perintah ini menghasilkan titik-titik dalam ruang 3D yang digunakan untuk membentuk permukaan kompleks. Parameter  $u$  dan  $v$  digunakan untuk mendefinisikan koordinat  $X$ ,  $Y$ , dan  $Z$  berdasarkan fungsi trigonometrik, yang memungkinkan penciptaan bentuk yang menarik. Setelah ini, Anda dapat menggunakan fungsi `plot3d` untuk memvisualisasikan permukaan ini. Permukaan yang dihasilkan akan memiliki bentuk yang menarik dan berputar, mungkin menyerupai struktur seperti torus atau permukaan kompleks lainnya tergantung pada rumus yang digunakan.

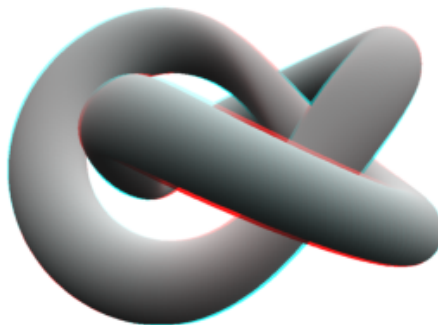
```
>plot3d(X,Y,Z,>anaglyph,<frame,>wire,scale=2.3):
```



Perintah ini digunakan untuk menggambar grafik 3D dari titik-titik yang telah dihitung sebelumnya menggunakan fungsi `plot3d`, dengan tampilan dalam mode `anaglyph` untuk efek 3D, menampilkan bingkai di sekitar grafik, dan menggunakan representasi `wireframe` untuk menyoroti struktur permukaan. Skala 2,3 kali memperbesar tampilan grafik, memungkinkan pengguna untuk melihat detail lebih dekat pada bentuk yang dihasilkan. Ini sangat berguna untuk menganalisis bentuk kompleks atau permukaan dalam ruang 3D.

Berikut 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):
```



Perintah ini menghasilkan grafik 3D dari permukaan yang kompleks berdasarkan fungsi  $x$ ,  $y$ , dan  $z$  yang telah ditentukan dengan parameter yang menciptakan bentuk menarik. Grafik tersebut diplot dalam mode `anaglyph` untuk efek 3D, tanpa bingkai, dengan skala dan zoom yang ditingkatkan, serta pencahayaan yang diatur untuk meningkatkan visualisasi. Hasilnya adalah tampilan 3D yang menarik dan kaya warna, memungkinkan analisis bentuk dan struktur permukaan yang dihasilkan oleh fungsi tersebut.

## Plot Statistik

---

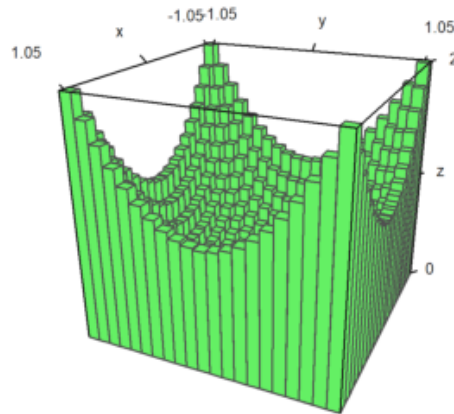
Plot batang juga dimungkinkan. Untuk itu, kita harus menyediakannya

- $x$ : vektor baris dengan  $n+1$  elemen
- $y$ : vektor kolom dengan  $n+1$  elemen
- $z$ : matriks nilai  $n \times n$ .

$z$  bisa lebih besar, tetapi hanya nilai  $n \times n$  yang akan digunakan.

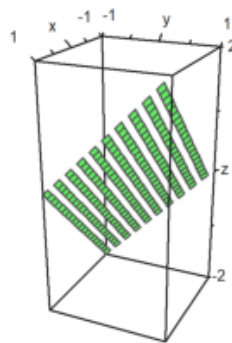
Dalam contoh ini, pertama-tama kita menghitung nilainya. Kemudian kita sesuaikan  $x$  dan  $y$ , sehingga vektor-vektornya berpusat pada nilai yang digunakan.

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



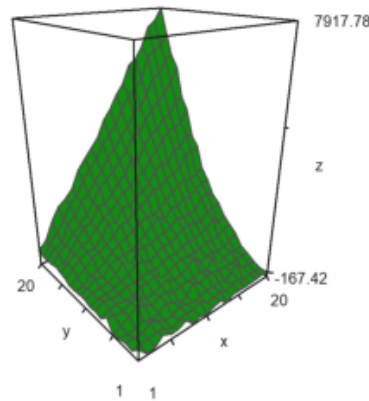
Dimungkinkan untuk membagi plot suatu permukaan menjadi dua bagian atau lebih.

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



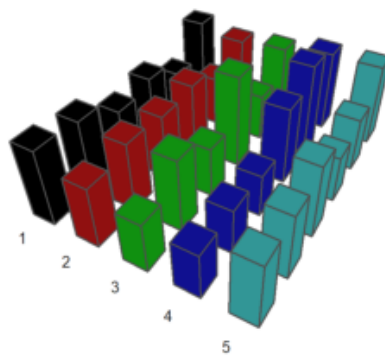
Jika memuat atau menghasilkan matriks data  $M$  dari file dan perlu memplotnya dalam 3D, Anda dapat menskalakan matriks ke  $[-1,1]$  dengan `skala(M)`, atau menskalakan matriks dengan `>zscale`. Hal ini dapat dikombinasikan dengan faktor penskalaan individual yang diterapkan sebagai tambahan.

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



Perintah ini menghasilkan grafik 3D dari permukaan yang ditentukan oleh kombinasi variabel  $i$  dan  $j$  dengan penambahan noise acak dari distribusi normal. Grafik ini diplot dengan penyesuaian skala pada sumbu  $z$  untuk memberikan visualisasi yang lebih baik dari ketinggian, dengan sudut dan zoom yang diatur untuk meningkatkan tampilan keseluruhan. Hasilnya adalah permukaan yang kompleks dengan variasi acak, memberikan representasi visual yang menarik dan informatif.

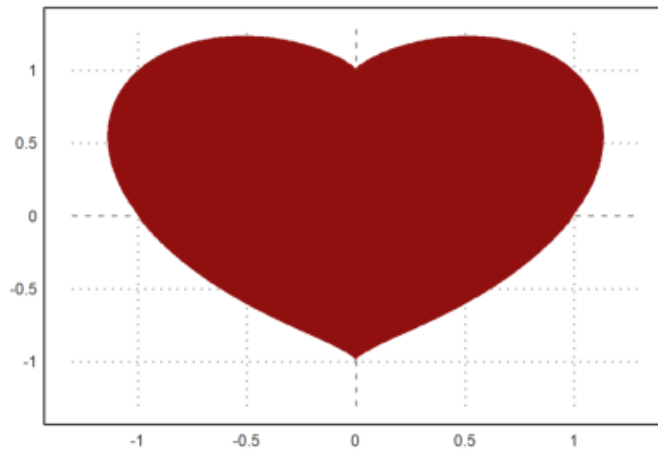
```
>Z=intrandom(5,100,6); v=zeros(5,6); ...
>loop 1 to 5; v[#]=getmultiplicities(1:6,Z[#]); end; ...
>columnsplot3d(v',scols=1:5,ccols=[1:5]):
```



Perintah ini digunakan untuk menghasilkan matriks acak dari angka bulat, menghitung frekuensi kemunculan angka 1 hingga 6 dalam setiap baris dari matriks tersebut, dan kemudian memvisualisasikan frekuensi tersebut dalam bentuk grafik 3D. Hasil grafik ini akan memberikan pemahaman yang lebih baik tentang distribusi angka yang dihasilkan secara acak dalam matriks  $Z$ .

## Permukaan Benda Putar

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



Secara keseluruhan, perintah ini digunakan untuk menggambar grafik 2D dari fungsi implicit yang kompleks, dengan parameter untuk menentukan gaya garis, warna, level, dan jumlah titik yang digunakan untuk membentuk kurva. Hasilnya akan menjadi visualisasi yang jelas dan informatif tentang bentuk fungsi yang digambarkan, dengan detail pada dua level yang ditentukan.

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

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

Kami ingin memutar kurva hati di sekitar sumbu y. Inilah ungkapan yang mendefinisikan hati:

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

Selanjutnya kita atur

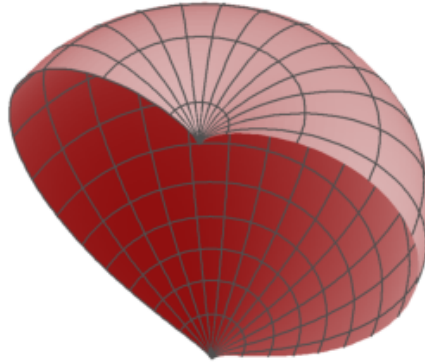
$$x = r \cdot \cos(a), \quad y = r \cdot \sin(a).$$

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

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

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

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

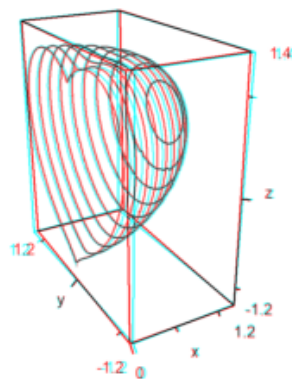


Berikut ini adalah plot 3D dari gambar di atas yang diputar mengelilingi sumbu z. Kami mendefinisikan fungsi yang mendeskripsikan objek.

```
>function f(x,y,z) ...
```

```
    r=x^2+y^2;
    return (r+z^2-1)^3-r*z^3;
endfunction
```

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



perintah ini memplot grafik 3D dengan  
 batas x dari 0 hingga 1.2,  
 batas y -1.2 hingga 1.2,  
 batas z dari -1.2 hingga 1.4,  
 dengan implisit 1  
 angel 30 derajat berarti sudut pandangnya 30 derajat,  
 grafik di zoom 2.5

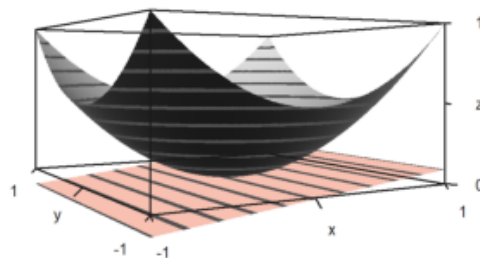
## Plot 3D Khusus

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

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

Sekarang `framedplot()` menyediakan bingkai, dan mengatur tampilan.

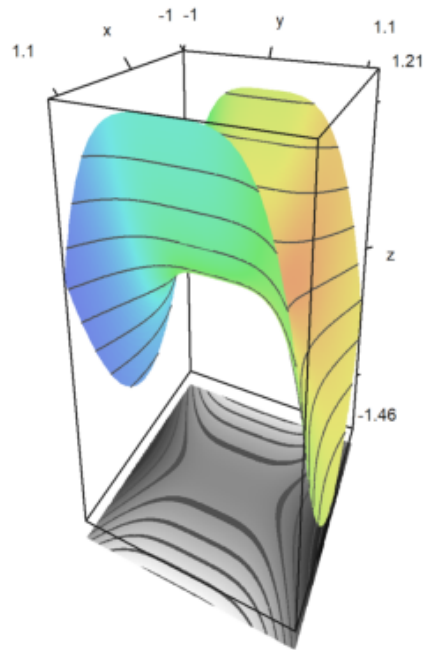
```
>framedplot("myplot",[-1,1,-1,1,0,1],height=0,angle=-30°, ...  
> center=[0,0,-0.7],zoom=3):
```



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

```
>x=-1:0.02:1.1; y=x'; z=x^2-y^4;  
>function myplot (x,y,z) ...  
  
    zoom(2);  
    wi=fullwindow();  
    plotcontourplane(x,y,z,level="auto",<scale);  
    plot3d(x,y,z,>hue,<scale,>add,color=white,level="thin");  
    window(wi);  
    reset();  
endfunction
```

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



perintah ini mendefinisikan fungsi yang menggambar grafik kontur dan grafik 3D dari fungsi matematis  $z=x^2-y^4$ . Fungsi ini memungkinkan pengguna untuk dengan mudah menghasilkan visualisasi dari permukaan dan kontur dalam satu langkah, dengan pengaturan tampilan yang telah disesuaikan. Hasilnya adalah representasi grafis yang informatif tentang hubungan antara variabel  $x$ ,  $y$ , dan  $z$ .

## Animasi

---

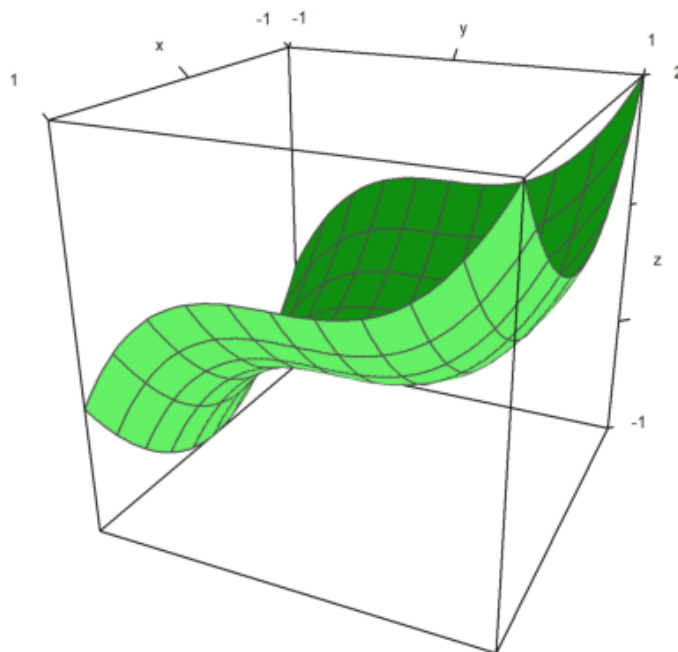
Euler dapat menggunakan frame untuk melakukan pra-komputasi animasi.

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

Silakan pelajari sumber rotasi untuk melihat lebih detail.

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





Secara keseluruhan, perintah ini menghasilkan grafik 3D dari fungsi  $z=x^2+y^3$  melalui fungsi `testplot`, kemudian memutar tampilan grafik tersebut untuk memberikan perspektif yang berbeda. Dengan cara ini, pengguna dapat menjelajahi visualisasi dari permukaan tersebut dengan lebih interaktif, melihat fitur dan karakteristik dari grafik dalam ruang 3D.

## Menggambar Povray

---

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

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

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

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

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

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

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

Anda perlu memuat file `povray`.

```
>load povray;
```

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

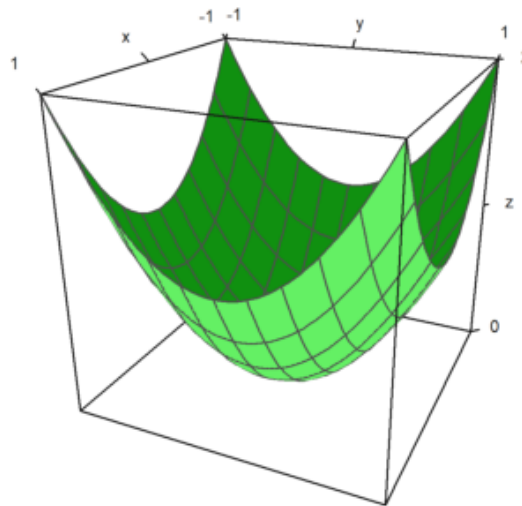
```
>defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

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

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

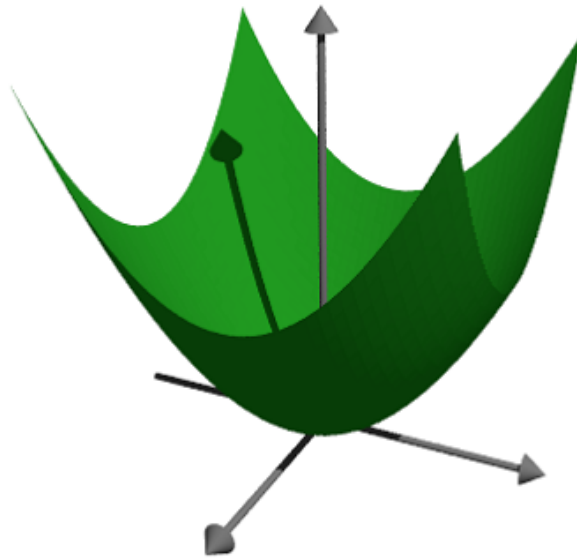
Jika Anda memulai perintah berikut, GUI Povray akan terbuka, menjalankan file, dan menutup secara otomatis. Karena alasan keamanan, Anda akan ditanya apakah Anda ingin mengizinkan file exe dijalankan. Anda dapat menekan batal untuk menghentikan pertanyaan lebih lanjut. Anda mungkin harus menekan OK di jendela Povray untuk mengonfirmasi dialog pengaktifan Povray.

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



Secara keseluruhan, perintah ini digunakan untuk menghasilkan grafik 3D dari fungsi  $Z=x^2+y^2$  dengan tingkat zoom senilai 2.

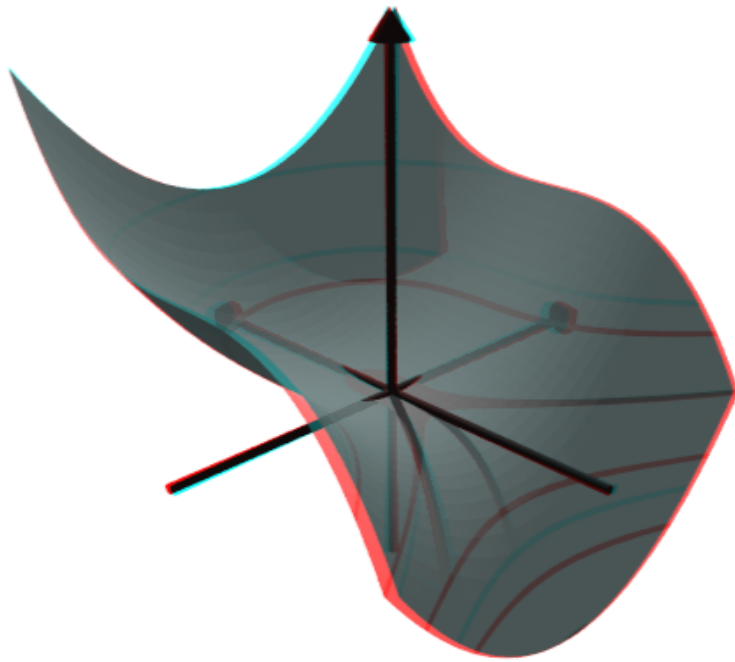
```
>pov3d("x^2+y^2", zoom=3) ;
```



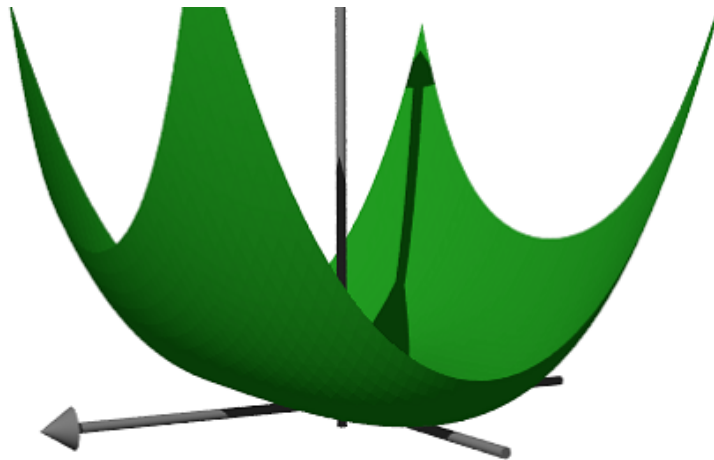
Secara keseluruhan, perintah ini digunakan untuk menghasilkan grafik 3D dari fungsi  $z=x^2+y^2$  dengan tingkat zoom senilai 3.

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

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



```
>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);
```

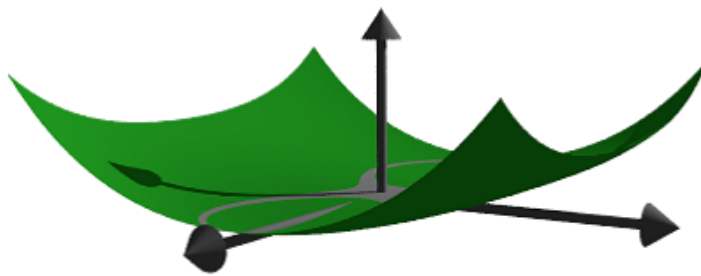


Terkadang perlu untuk mencegah penskalaan fungsi, dan menskalakan fungsi secara manual. Kita memplot himpunan titik pada bidang kompleks, dimana hasil kali jarak ke 1 dan -1 sama dengan 1.

## Latihan Soal

---

```
>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=10°,n=50, ...  
>  <fscale, zoom=3.8);
```



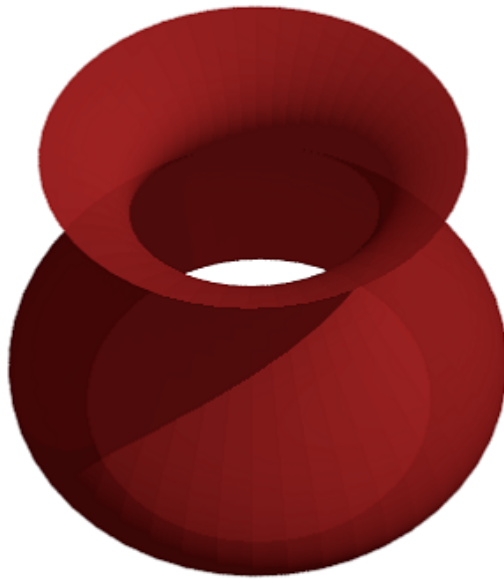
## Merencanakan dengan Koordinat

---

Daripada menggunakan fungsi, kita bisa memplotnya dengan koordinat. Seperti di plot3d, kita memerlukan tiga matriks untuk mendefinisikan objek.

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

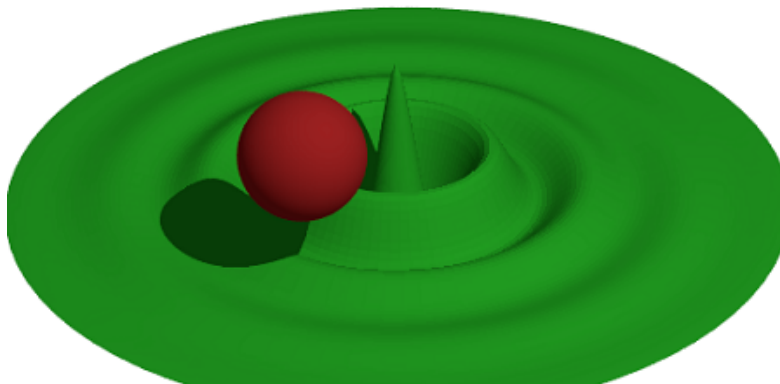
```
>function f(x) := x^3-x+1; ...  
>x=-1:0.01:1; t=linspace(0,2pi,50)'; ...  
>Z=x; X=cos(t)*f(x); Y=sin(t)*f(x); ...  
>pov3d(X,Y,Z,angle=40°,look=povlook(red,0.1),height=50°,axis=0, zoom=4, light=[10,5,15]);
```



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

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

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



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

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

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

$$x^2 y^3$$

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

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

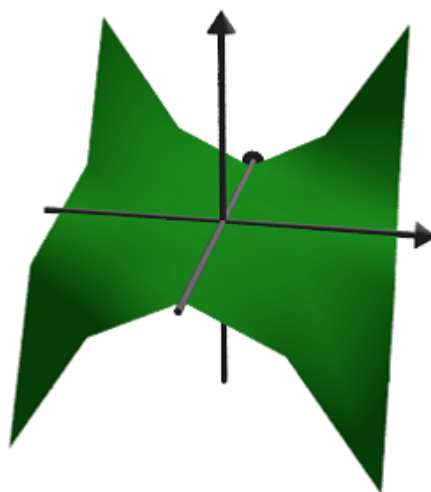
Kami mendefinisikan normal sebagai produk silang dari turunan ini, dan mendefinisikan fungsi koordinat.

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

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

Kami hanya menggunakan 25 poin.

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



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

See: Examples\Trefoil Knot | Trefoil Knot

Untuk tampilan yang bagus dengan poin yang tidak terlalu banyak, kami menambahkan vektor normal di sini. Kami menggunakan Maxima untuk menghitung normalnya bagi kami. Pertama, tiga fungsi koordinat sebagai ekspresi simbolik.

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

Kemudian kedua vektor turunan ke x dan y.

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

Sekarang normalnya, yaitu perkalian silang kedua turunannya.

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

Kami sekarang mengevaluasi semua ini secara numerik.

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

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

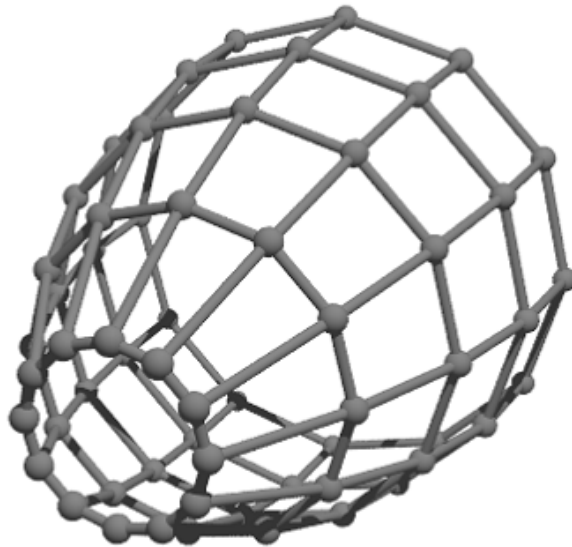
```
>pov3d(X(x,y),Y(x,y),Z(x,y),>anaglyph,axis=0,zoom=5,w=450,h=350, ...  
> <shadow,look=povlook(blue), ...  
> xv=&"dn[1]"(x,y), yv=&"dn[2]"(x,y), zv=&"dn[3]"(x,y));
```





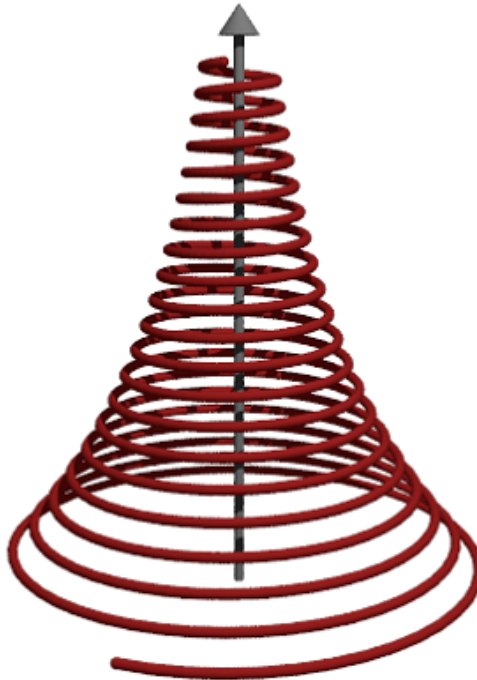
Kami juga dapat menghasilkan grid dalam 3D.

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



Dengan povgrid(), kurva dimungkinkan.

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



## Objek Povray

---

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

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

Pertama kita mendefinisikan tiga silinder, dan menyimpannya dalam string di Euler. Fungsi povx() dll. hanya mengembalikan vektor [1,0,0], yang dapat digunakan sebagai gantinya.

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

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

```
>c2
```

```
cylinder { <0,0,-1>, <0,0,1>, 1
  texture { pigment { color rgb <0.941176,0.941176,0.392157> } }
  finish { ambient 0.2 }
}
```

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

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

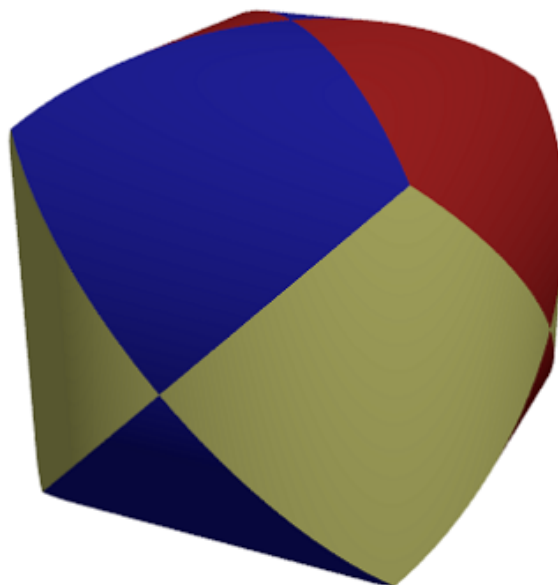
```
texture { pigment { color rgbf <0.101961,0.2,0.301961,0.1> } }  
finish { ambient 0.5 }
```

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

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

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

```
>povend;
```



Fungsi berikut menghasilkan fraktal secara rekursif.

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

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

```

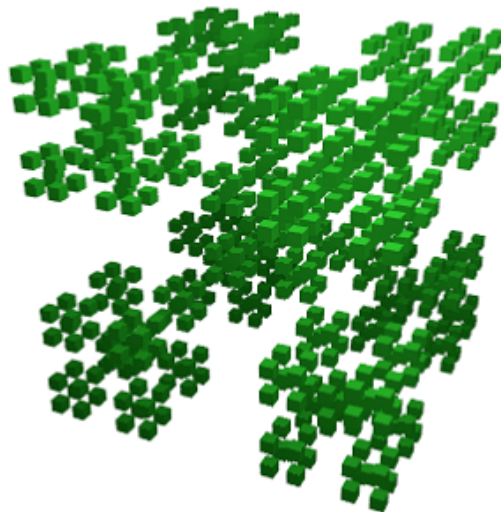
if n==1 then writeln(onebox(x,y,z,h));
else
  h=h/3;
  fractal(x,y,z,h,n-1);
  fractal(x+2*h,y,z,h,n-1);
  fractal(x,y+2*h,z,h,n-1);
  fractal(x,y,z+2*h,h,n-1);
  fractal(x+2*h,y+2*h,z,h,n-1);
  fractal(x+2*h,y,z+2*h,h,n-1);
  fractal(x,y+2*h,z+2*h,h,n-1);
  fractal(x+2*h,y+2*h,z+2*h,h,n-1);
  fractal(x+h,y+h,z+h,h,n-1);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

```

```

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

```



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

```

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

```

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

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

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

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

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

Kami membuat kubus kedua, dan memutar serta menskalakannya sedikit.

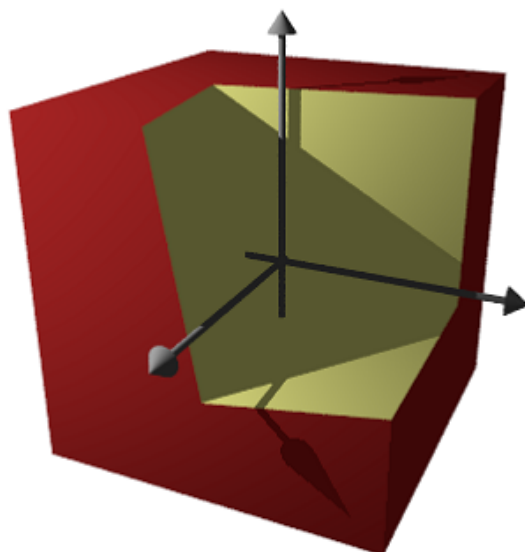
```
>c2=povobject ("mycube",povlook (yellow),translate=[1,1,1], ...  
> rotate=xrotate (10°)+yrotate (10°), scale=1.2);
```

Lalu kita ambil selisih kedua benda tersebut.

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

Sekarang tambahkan tiga sumbu.

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



## Fungsi Implisit

---

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

Sintaks untuk fungsinya sedikit berbeda. Anda tidak dapat menggunakan keluaran ekspresi Maxima atau Euler.

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

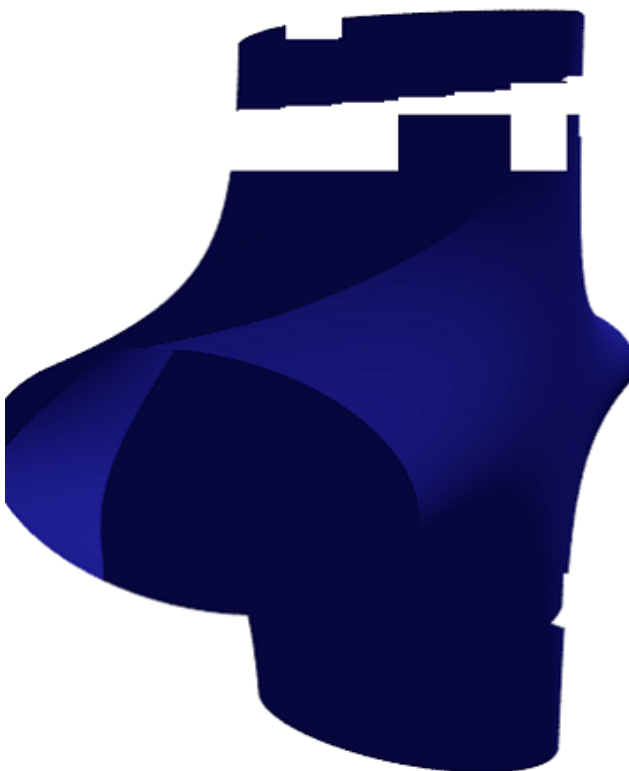
```
>povstart (angle=70°,height=50°,zoom=4);
>c=0.1;
>d=0.1;
>writeln(povsurface (" (pow (pow (x,2)+pow (y,2)-pow (c,2),2)+pow (pow (z,2)-1,2) ) * "+" (pow (pow (y,2)+pow (z,2)-pow (c,2),2)+pow (x^2-1,2) ) * "+" (pow (pow (z,2)+pow (x,2)-pow (c,2),2)+pow (y^2-1,2) ) ) );
>povend();
```

Error : Povray error!

Error generated by error() command

```
povray:
  error("Povray error!");
Try "trace errors" to inspect local variables after errors.
povend:
  povray(file,w,h,aspect,exit);
```

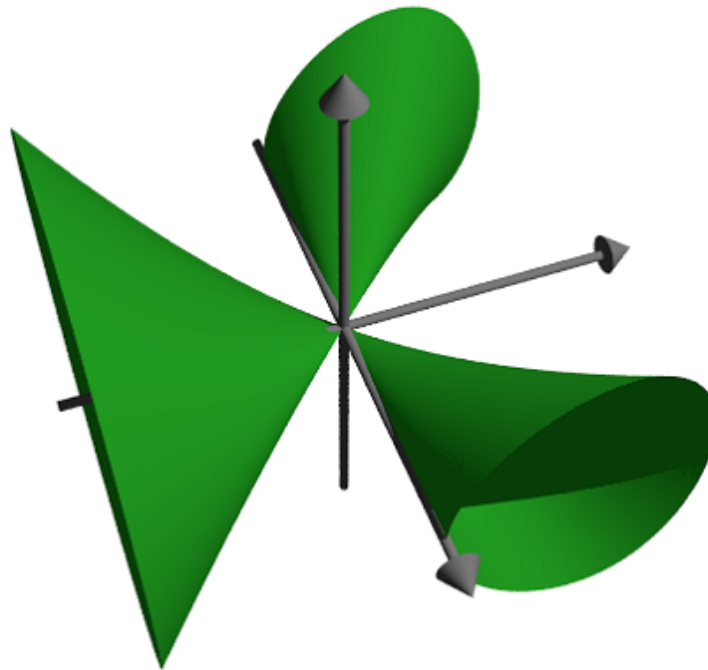
```
>povstart (angle=25°,height=10°);
>writeln(povsurface ("pow (x,2)+pow (y,2)*pow (z,2)-1",povlook (blue),povbox (-2,2,"") ));
>povend();
```



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

Buat permukaan implisit. Perhatikan sintaksis yang berbeda dalam ekspresi.

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



## Objek Jaring

Dalam contoh ini, kami menunjukkan cara membuat objek mesh, dan menggambarinya dengan informasi tambahan.

Kita ingin memaksimalkan  $xy$  pada kondisi  $x+y=1$  dan mendemonstrasikan sentuhan tangensial garis datar.

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

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

Berikut ini definisi objek mesh, dan segera menuliskannya ke dalam file.

```
>x=0:0.02:1; y=x'; z=x*y; vx=-y; vy=-x; vz=1;  
>mesh=povtriangles(x,y,z,"",vx,vy,vz);
```

Sekarang kita mendefinisikan dua cakram, yang akan berpotongan dengan permukaan.

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

Tulis permukaannya dikurangi kedua cakram.

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

Tuliskan kedua perpotongan tersebut.

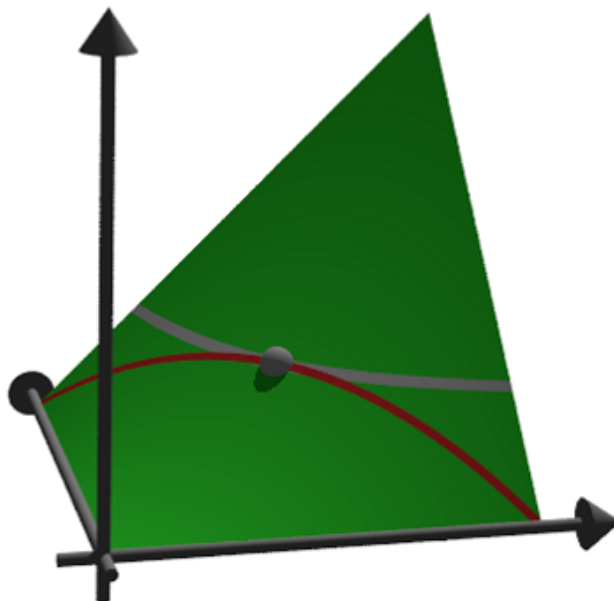
```
>writeln(povintersection([mesh,cl],povlook(red))); ...  
>writeln(povintersection([mesh,ll],povlook(gray)));
```

Tulis poin maksimal.

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

Tambahkan sumbu dan selesai.

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





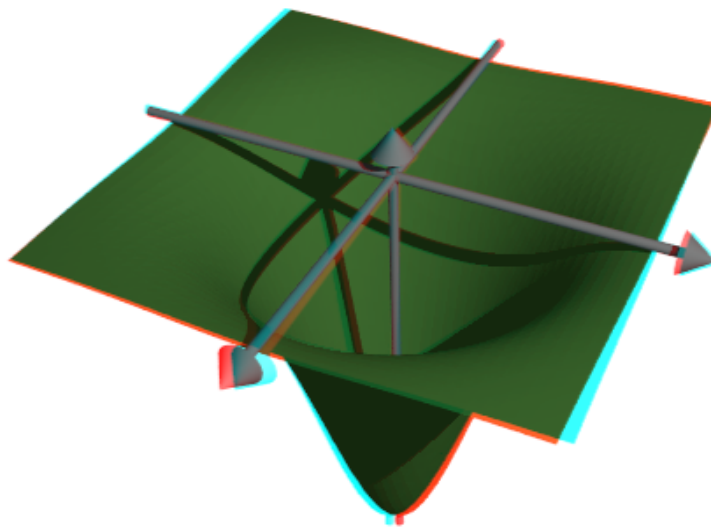
perintah ini digunakan untuk menggambar sumbu koordinat 3D (x, y, z) dengan rentang dari 0 hingga 1 pada masing-masing sumbu dan dengan margin kecil 0.015, lalu menutup blok POV rendering.

## Anaglyph di Povray

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

Tentu saja, Anda memerlukan kacamata berwarna merah/cyan untuk melihat contoh berikut dengan benar. Fungsi pov3d() memiliki saklar sederhana untuk menghasilkan anaglyph.

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

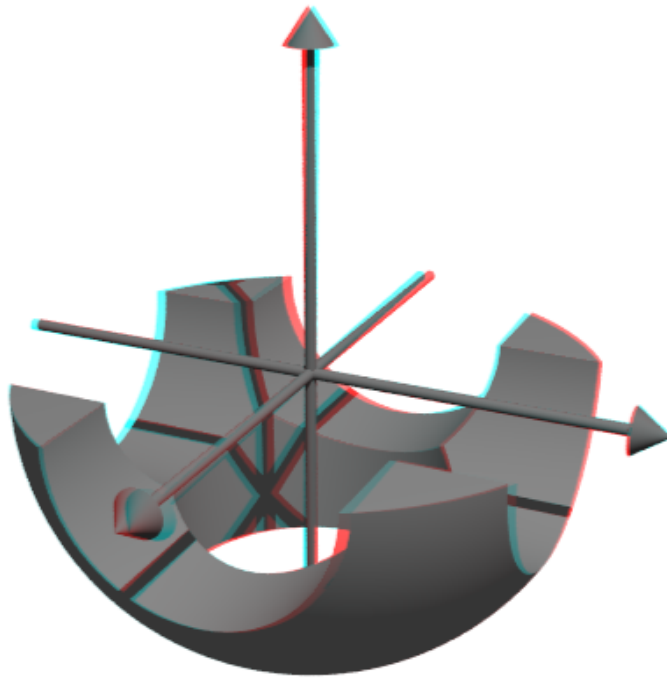


Jika Anda membuat adegan dengan objek, Anda perlu memasukkan pembuatan adegan ke dalam fungsi, dan menjalankannya dua kali dengan nilai berbeda untuk parameter anaglyph.

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

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

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



perintah ini digunakan untuk menghasilkan tampilan 3D dari adegan yang bernama "myscene" dengan tingkat pembesaran tertentu, yang dapat dilihat menggunakan kacamata anaglyph (misalnya merah-biru atau merah-hijau).

## Mendefinisikan Objek sendiri

---

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

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

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

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

Ini torus pertama kami.

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

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

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

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

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

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

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

```
>povend();
```

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

```
>povend(<keluar>);
```

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

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



Berikut adalah contoh yang lebih rumit. Kami memecahkannya

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

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

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

Pertama, mari kita periksa, apakah contoh ini punya solusinya.

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

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

Ya, sudah.

Selanjutnya kita mendefinisikan dua objek. Yang pertama adalah pesawat

$$a \cdot x \leq b$$

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

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

Kemudian kita tentukan perpotongan semua setengah ruang dan kubus.

```
>function adm (A, b, r, look="") ...
```

```
    ol=[];
    loop 1 to rows(A); ol=ol|oneplane(A[#],b[#]); end;
    ol=ol|povbox([0,0,0],[r,r,r]);
    return povintersection(ol,look);
endfunction
```

Sekarang kita dapat merencanakan adegannya.

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

Berikut ini adalah lingkaran di sekitar optimal.

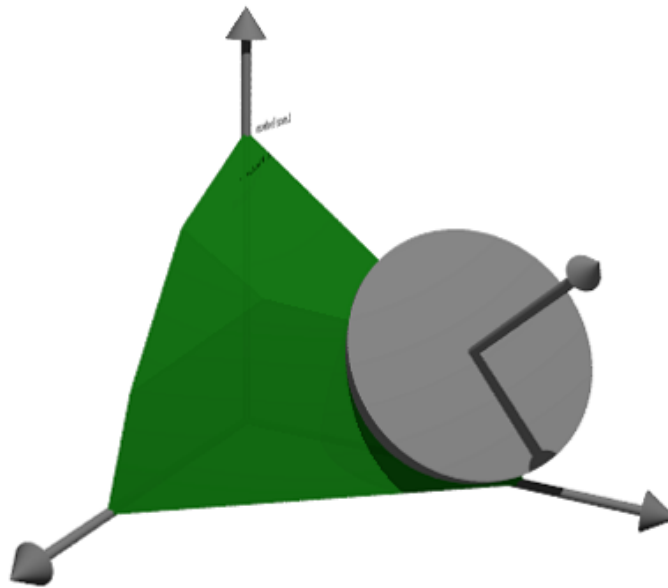
```
>writeln(povintersection([povsphere(x,0.5),povplane(c,c.x')], ...
>  povlook(red,0.9)));
```

Dan kesalahan ke arah optimal.

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

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

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



perintah ini digunakan untuk menempatkan teks "Linear Problem" pada posisi tertentu dalam grafik atau ruang 3D, dengan ukuran dan rotasi tertentu, dan kemudian mengakhiri proses render gambar

## Lebih Banyak Contoh

---

Anda dapat menemukan beberapa contoh Povray di Euler di file berikut.

See: Examples/Dandelion Spheres

See: Examples/Donat Math

See: Examples/Trefoil Knot

See: Examples/Optimization by Affine Scaling

## Latihan Soal

---

1. buatlah gabungan 2 silinder dengan fungsi povx() berwarna merah dan povz() berwarna kuning dan zoom 4

```
>povstart (zoom=4)
>c1 = povcylinder (-povx,povx,1,povlook (red)) ;
>c2 = povcylinder (-povz,povz,1,povlook (yellow)) ;
>writeln (povintersection ([c1,c2]));
>povend()
```

