SIF5005 våren 2003: Maple-øving 3

Løsningsforslag.

1 Oppsett

Her importerer vi noen navn vi skal bruke senere, så vi slipper å si plots[spacecurve], etc.

```
> with(plots,display,display3d,tubeplot,spacecurve,fieldplot,fieldplot3
> d);
```

[*display*, *display3d*, *tubeplot*, *spacecurve*, *fieldplot*, *fieldplot3d*]

Plotsetup: bruk window eller inline etter ønske. For interaktiv bruk foretrekker jeg window, for jeg får større og mer forståelige figurer. Men når figurene skal med i en utskrift, må vi bruke inline.

```
> plotsetup(window);
```

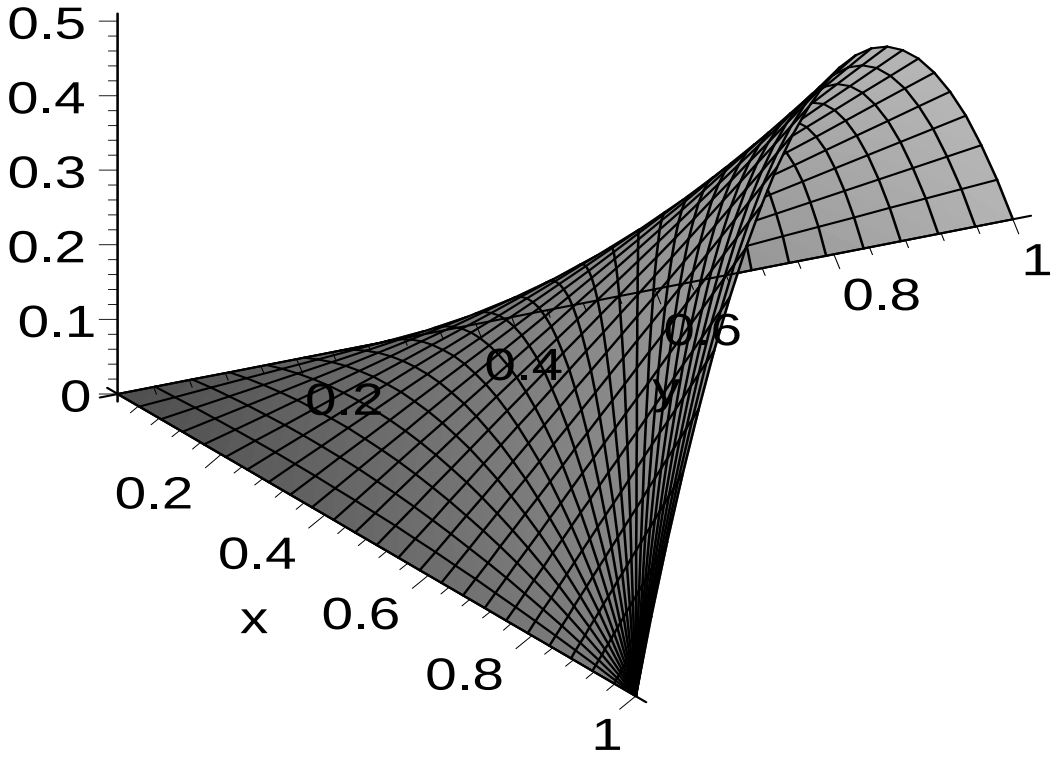
Denne er klippet ut av legeme.mws og limt inn her:

```
> # Hjelpefunksjon:  P er en permutasjon på tre variable.
> plotbody:=proc(P,xx,yy,zz)
> local x,xlo,xhi,y,ylo,yhi,z,zlo,zhi,etc,ps;
> etc:=args[5..nargs];
> if not typematch(xx,x::name=(xlo::algebraic..xhi::algebraic))
> then error("Bad first argument to plotxyz"); fi;
> if not typematch(yy,y::name=(ylo::algebraic..yhi::algebraic))
> then error("Bad second argument to plotxyz"); fi;
> if not typematch(zz,z::name=(zlo::algebraic..zhi::algebraic))
> then error("bad third argument to plotxyz"); fi;
> ps:=NULL;
> try ps:=ps, plot3d(P(x,y,zhi),xx,yy,etc);
> catch:  print("3max surface", lasterror);
> end;
> try ps:=ps, plot3d(P(x,y,zlo),xx,yy,etc);
> catch:  print("3min surface",lasterror);
> end;
> try ps:=ps,
> plot3d(P(x,ylo,z),x=xlo..xhi,z=subs(y=ylo,zlo..zhi),etc);
> catch:  print("2min surface", lasterror);
> end;
> try ps:=ps,
> plot3d(P(x,yhi,z),x=xlo..xhi,z=subs(y=yhi,zlo..zhi),etc);
> catch:  print("2max surface", lasterror);
> end;
> try ps:=ps,
> plot3d(P(xlo,y,z),y=subs(x=xlo,ylo..yhi),z=subs(x=xlo,zlo..zhi),etc);
> catch:  print("1min surface", lasterror);
> end;
> try ps:=ps,
> plot3d(P(xhi,y,z),y=subs(x=xhi,ylo..yhi),z=subs(x=xhi,zlo..zhi),etc);
> catch:  print("1max surface", lasterror);
> end;
> if (nops([ps])>0) then
> plots[display](ps);
> else
> error("no surface worked out");
> fi;
> end:
> plot123:=proc() plotbody((x,y,z)->[x,y,z],args); end:
> plot132:=proc() plotbody((x,y,z)->[x,z,y],args); end:
> plot213:=proc() plotbody((x,y,z)->[y,x,z],args); end:
> plot231:=proc() plotbody((x,y,z)->[z,x,y],args); end:
> plot312:=proc() plotbody((x,y,z)->[y,z,x],args); end:
> plot321:=proc() plotbody((x,y,z)->[z,y,x],args); end:
```

2 Oppgave 1: Mer om 3D-plott for flater

2.1 Variable grenser

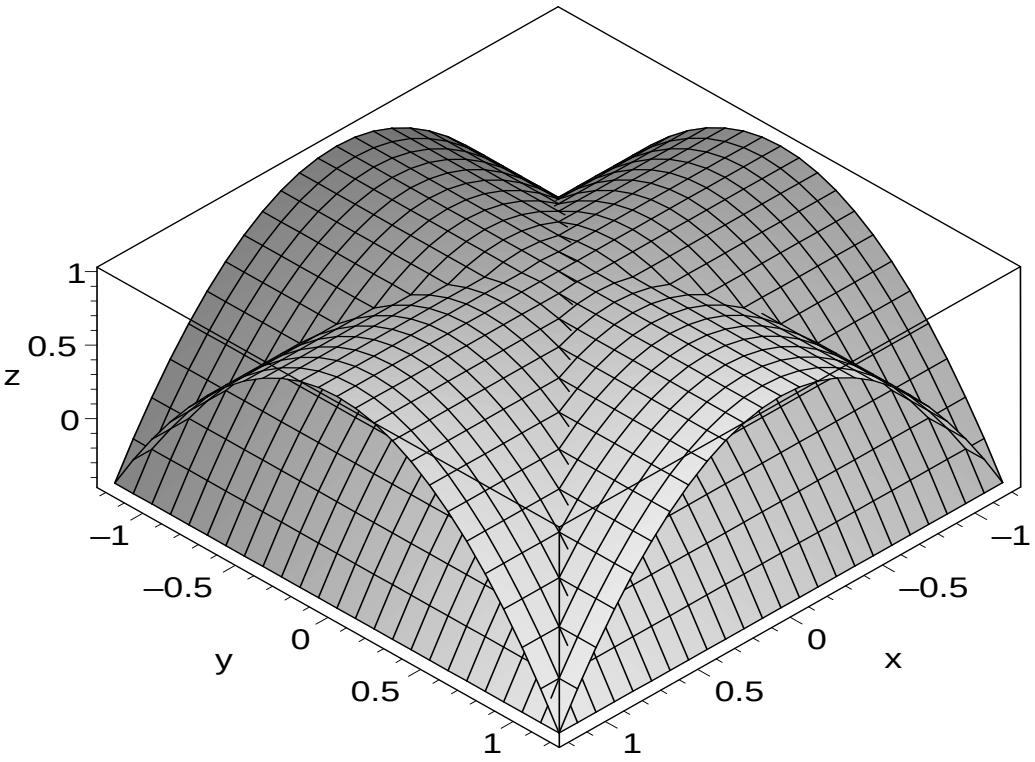
```
> plot3d(2*x*y,x=0..1,y=0..1-x,
> scaling=constrained,axes=normal,orientation=[-30,65]);
```



2.1.1 To parabolske sylindre

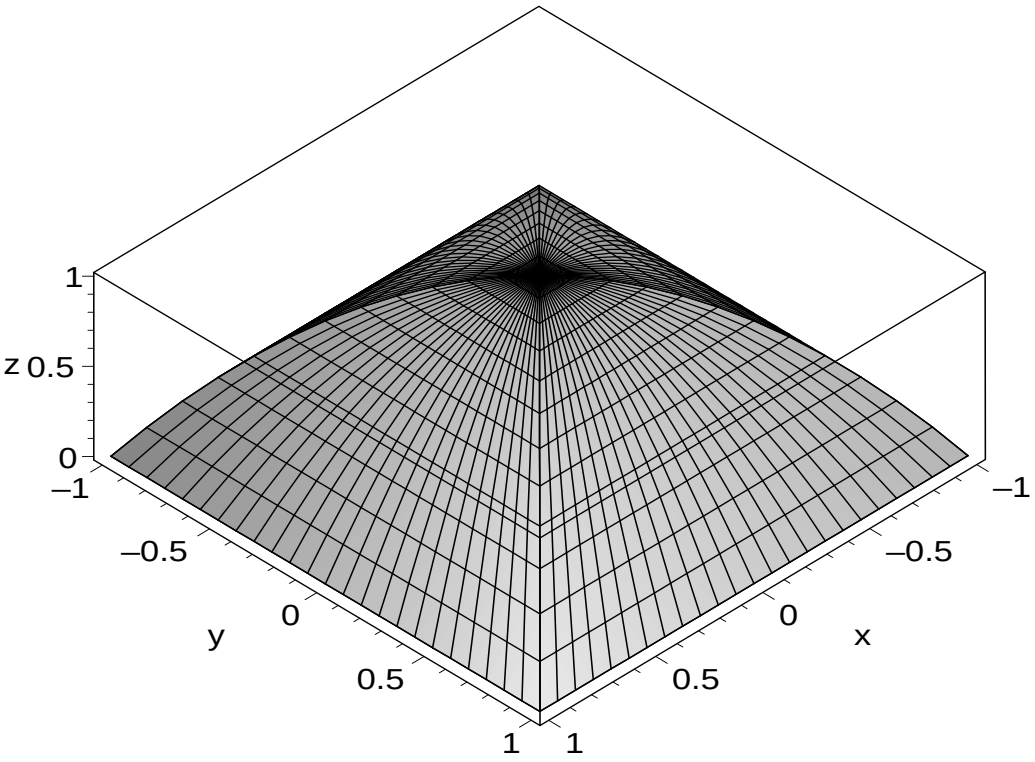
Dette visualiserer et eksempel fra E&P, hvor vi skulle beregne volumet av området over xy-planet og under flatene $z = 1 - x^2$ og $z = 1 - y^2$: Først blåser vi i skjæringen mellom de to flatene, og prøver å få oversikt:

```
> display3d(
> plot3d(1-x^2,x=-1.2..1.2,y=-1.2..1.2),
> plot3d(1-y^2,x=-1.2..1.2,y=-1.2..1.2),
> axes=boxed, scaling=constrained, labels=[x,y,z]);
```



Og her er resultatet etter at vi har regnet litt og tenkt oss om:

```
> display3d(
> plot3d(1-x^2,x=-1..1,y=-abs(x)..abs(x)),
> plot3d([x,y,1-y^2],y=-1..1,x=-abs(y)..abs(y)),
> plot3d(0,x=-1..1,y=-1..1),
> axes=boxed, scaling=constrained, labels=[x,y,z]);
```



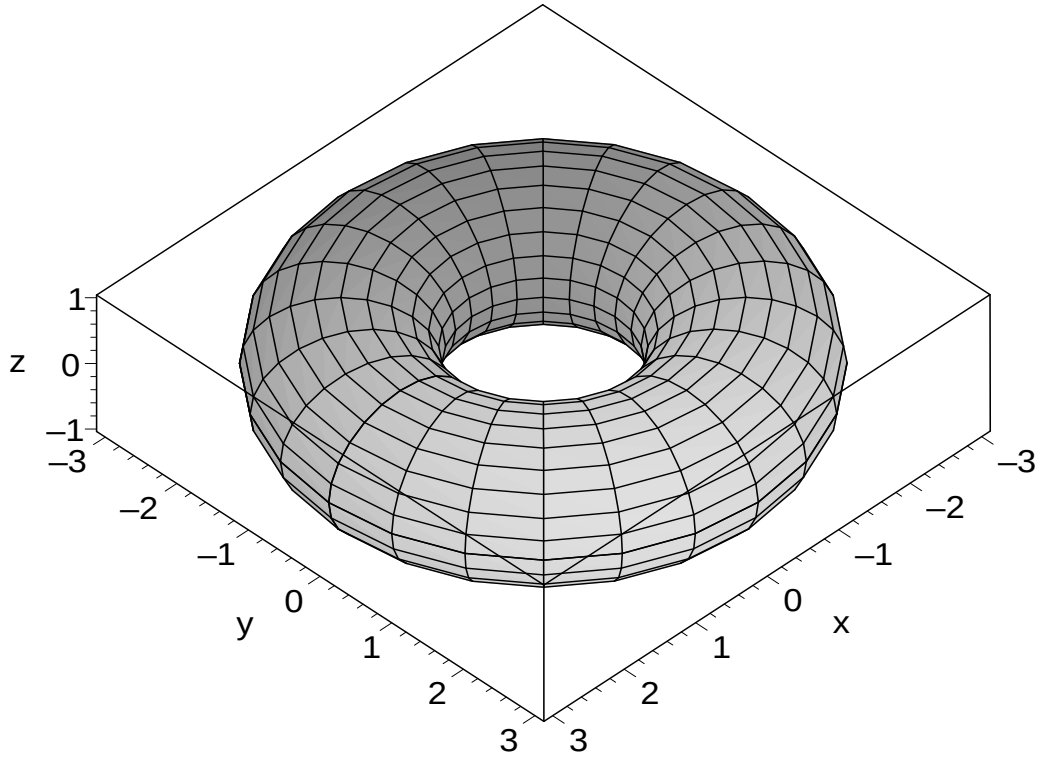
2.2 Parametriske flater

2.2.1 Torus

```
> a:=2:  b:=1:
> F:=(u,v)->[(a+b*cos(u))*cos(v),(a+b*cos(u))*sin(v),b*sin(u)];
```

$$F := (u,\, v) \to [(a + b\cos(u))\cos(v),\ (a + b\cos(u))\sin(v),\ b\sin(u)]$$

```
> plot3d(F(u,v),u=0..2*Pi,v=0..2*Pi,
> scaling=constrained,axes=boxed,labels=["x","y","z"]);
```

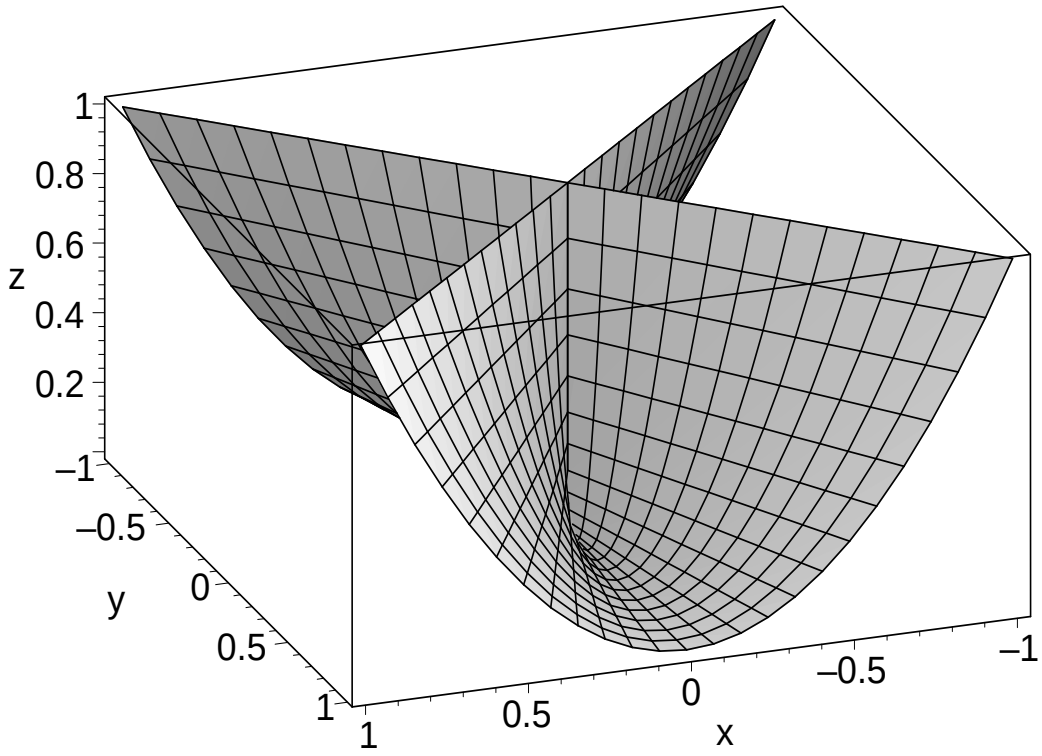


2.2.2 Whitneys paraply

```
> F:=(u,v)->[u*v,u,v^2];
```

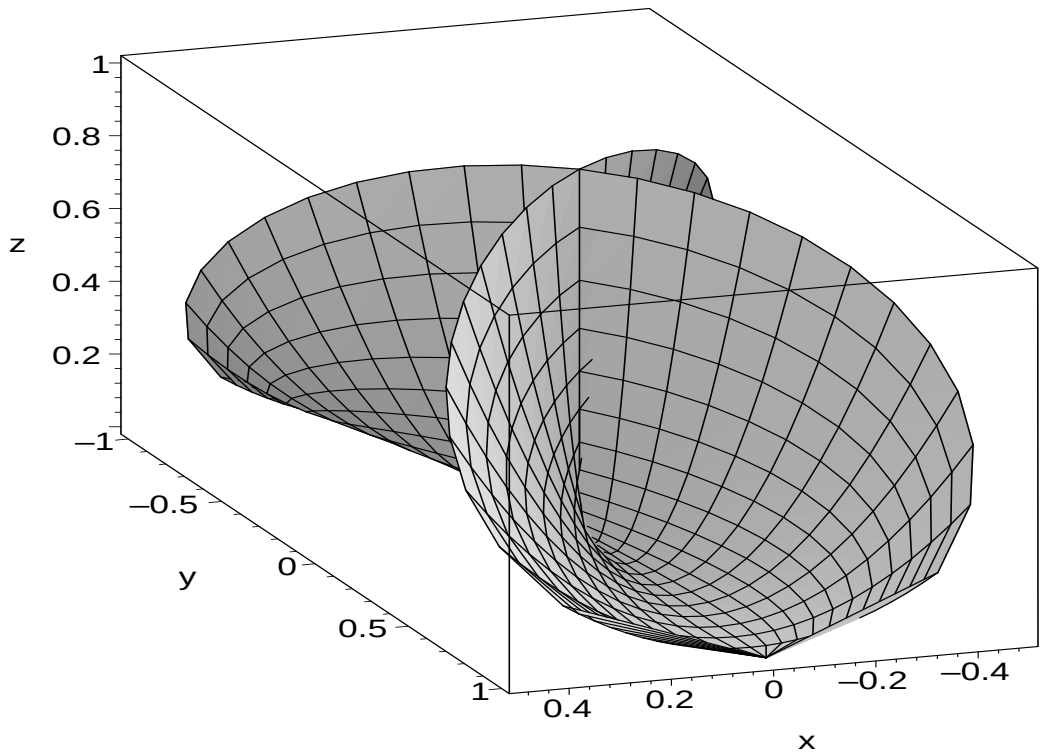
$$F := (u,\, v) \to [u\,v,\ u,\ v^2]$$

```
> plot3d(F(u,v),u=-1..1,v=-1..1,
> scaling=constrained,axes=boxed,orientation=[70,70],
> labels=["x","y","z"]);
```



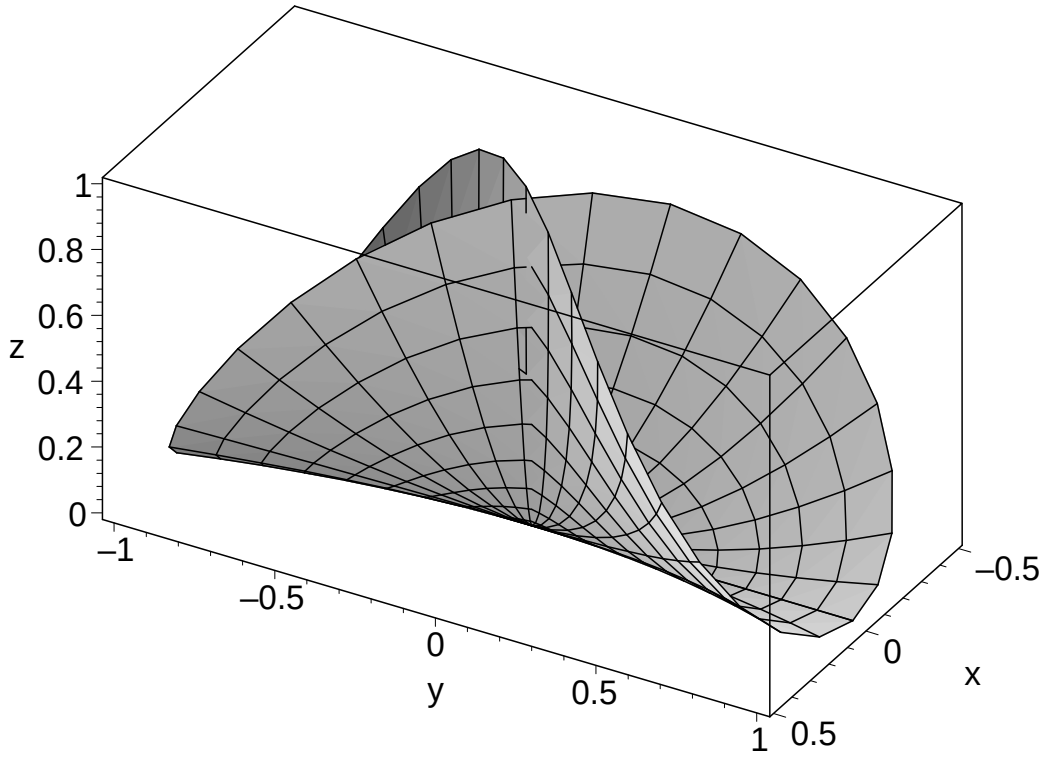
Hvis vi ønsker å begrense parametrene til enhetssirkelen (for eksempel), så kan også parametriske plott ha variable grenser:

```
> plot3d(F(u,v),u=-1..1,v=-sqrt(1-u^2)..sqrt(1-u^2),
> scaling=constrained,axes=boxed,orientation=[70,70],
> labels=["x","y","z"]);
```



Og vil vi ha det riktig fancy, kan vi bruke reparametrisere ved å polarkoordinater i uv-planet!

```
> plot3d(F(r*cos(t),r*sin(t)),r=0..1,t=0..2*Pi,grid=[10,40],
> scaling=constrained,axes=boxed,orientation=[30,60],
> labels=["x","y","z"]);
```



3 Oppgave 2: Fra hjemmeøving 5

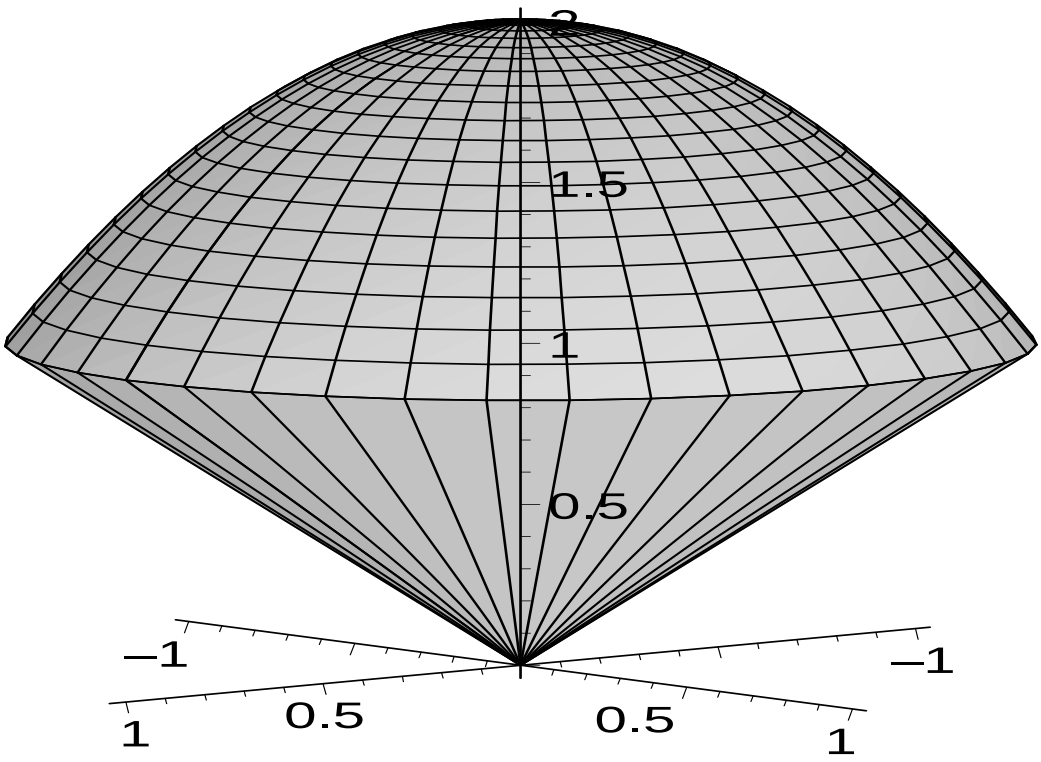
Noen poeng her:

Oppgave 3: Variable grenser i 3D-plott.

Oppgave 5: Mer av det samme. Dessuten parametriske plott.

3.1 Oppgave 2

```
> display3d(
> plot3d([r,theta,r],r=0..1,theta=0..2*Pi,coords=cylindrical,
> grid=[2,40]),
> plot3d([r,theta,2-r^2],r=0..1,theta=0..2*Pi,coords=cylindrical,
> grid=[20,40]),
> scaling=constrained,axes=normal,orientation=[50,80]);
```

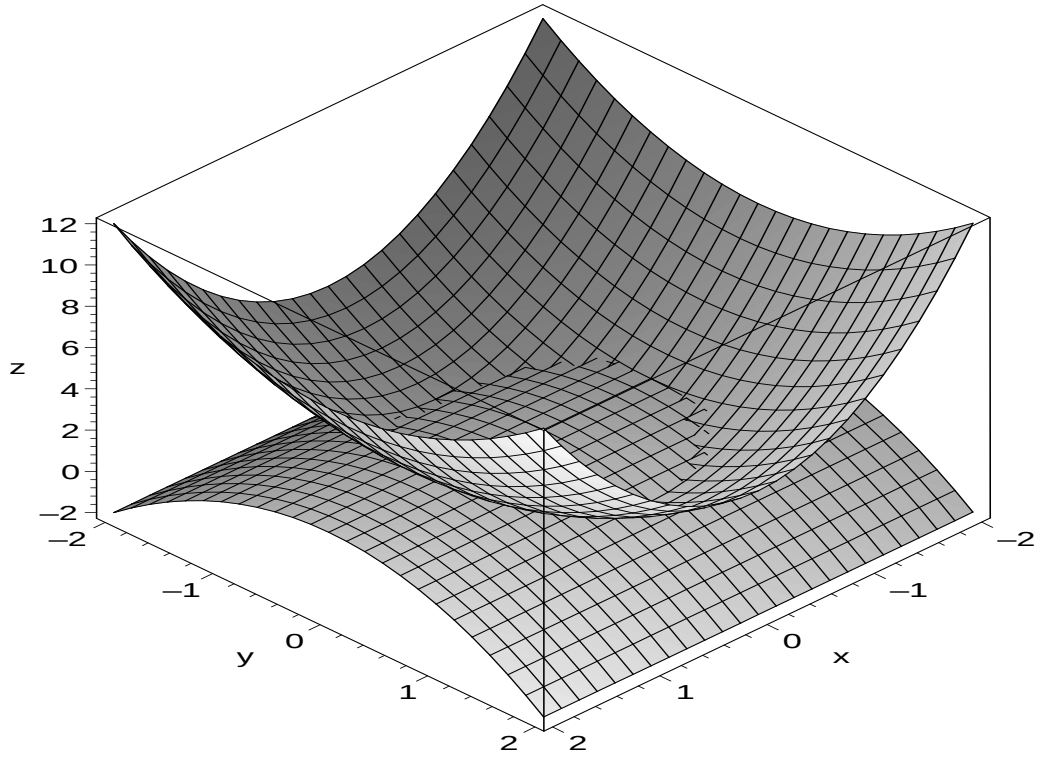


3.2 Oppgave 3

Grovutkast uten hensyn til hvor flatene skjærer hverandre:

```
> omr:=x=-2..2,y=-2..2;
> display3d(
> plot3d(2*x^2+y^2,omr),
> plot3d(2-y^2,omr),
> axes=boxed,labels=["x","y","z"]);
```

$$omr := x = -2..2,\ y = -2..2$$

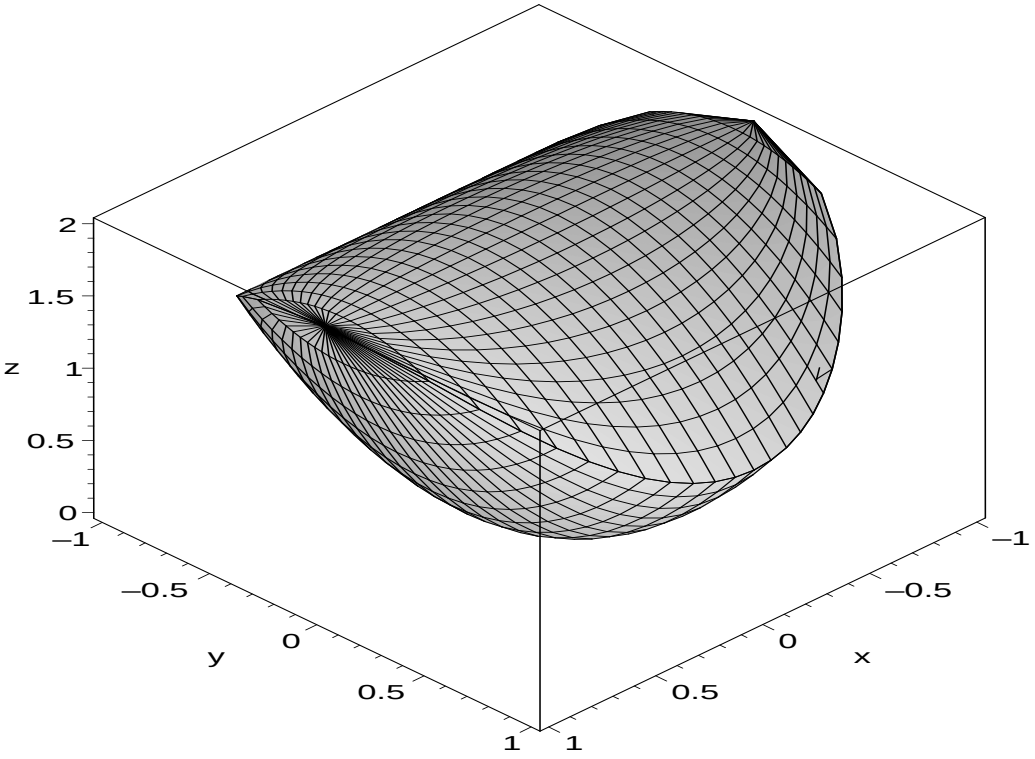


Litt håndregning viser at de to flatene skjærer hverandre i sylinderen
$x^2 + y^2 = 1$
.

Vi kan plotte de to flatene innenfor, på en av to måter. Det enkleste er å gjøre en enkel modifikasjon av det over:

```
> omr:=x=-1..1,y=-sqrt(1-x^2)..sqrt(1-x^2);
> display3d(
> plot3d(2*x^2+y^2,omr),
> plot3d(2-y^2,omr),
> axes=boxed,labels=["x","y","z"]);
```

$$omr := x = -1..1, y = -\sqrt{1-x^2}..\sqrt{1-x^2}$$

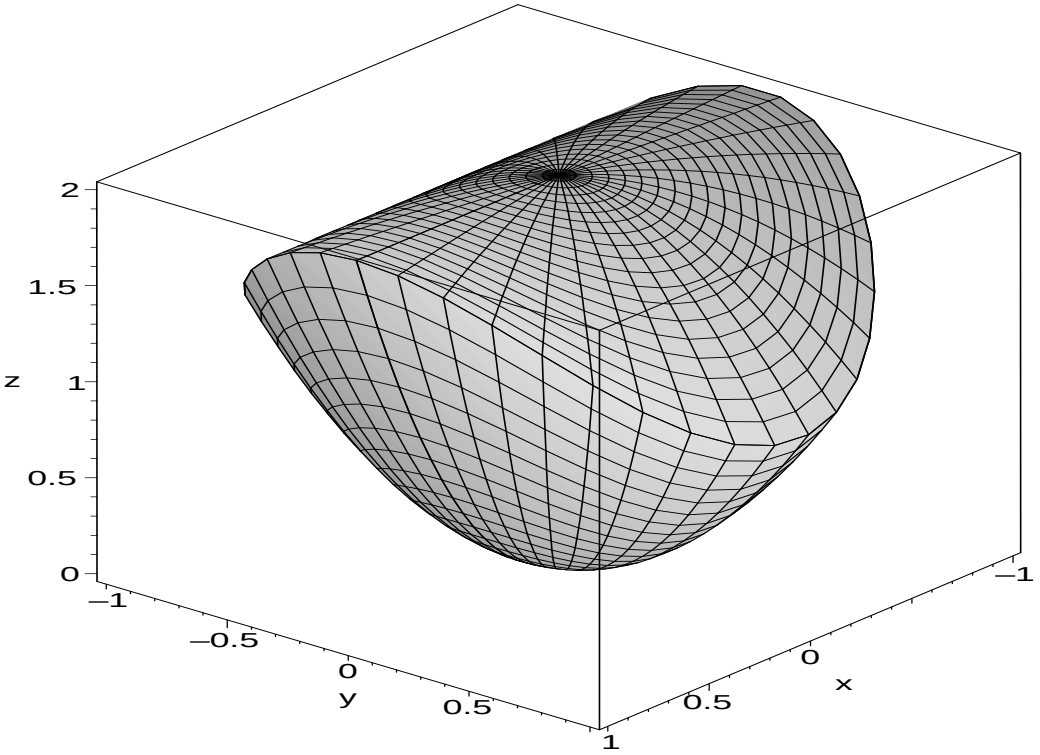


Hvis man føler for å lage et penere bilde, kan man ty til sylinderkoordinater, men det er kanskje ikke verdt anstrengelsen:

```
> omr:=r=0..1,theta=0..2*Pi,coords=cylindrical;
> gridopt:=grid=[20,40];
> display3d(
> plot3d([r,theta,r^2*(1+cos(theta)^2)],omr,gridopt),
> plot3d([r,theta,2-(r*sin(theta))^2],omr,gridopt),
> axes=boxed,labels=["x","y","z"],orientation=[40,60]);
```

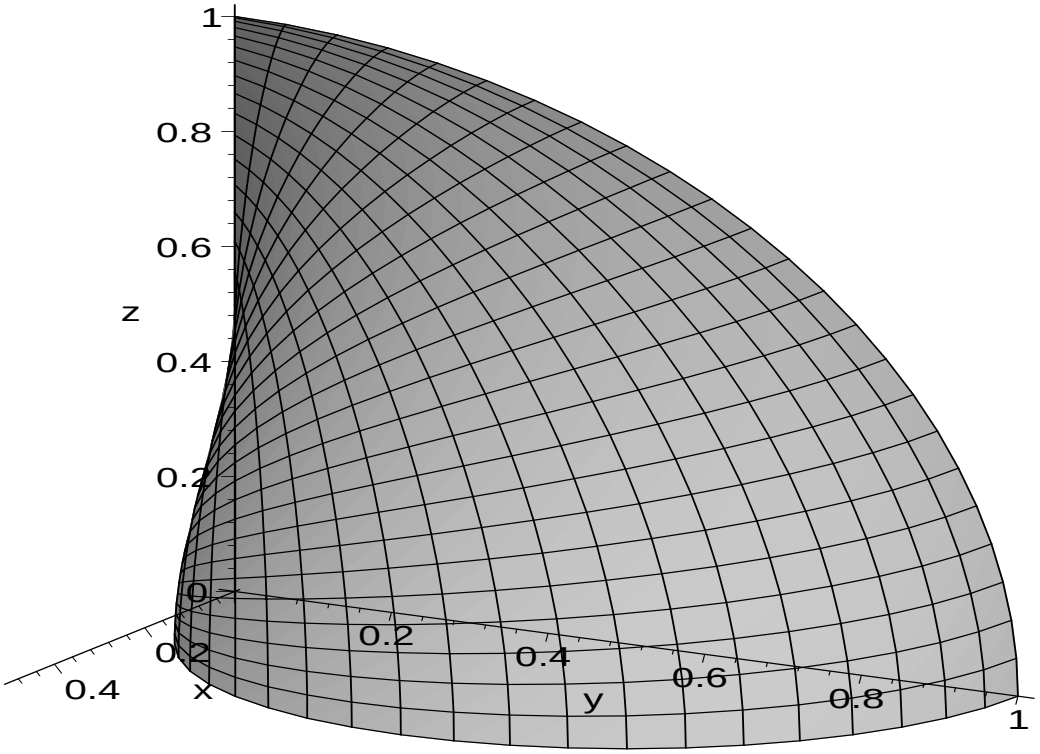
$$omr := r = 0..1,\ \theta = 0..2\,\pi,\ coords = cylindrical$$

$$gridopt := grid = [20,\ 40]$$



3.3 Oppgave 4

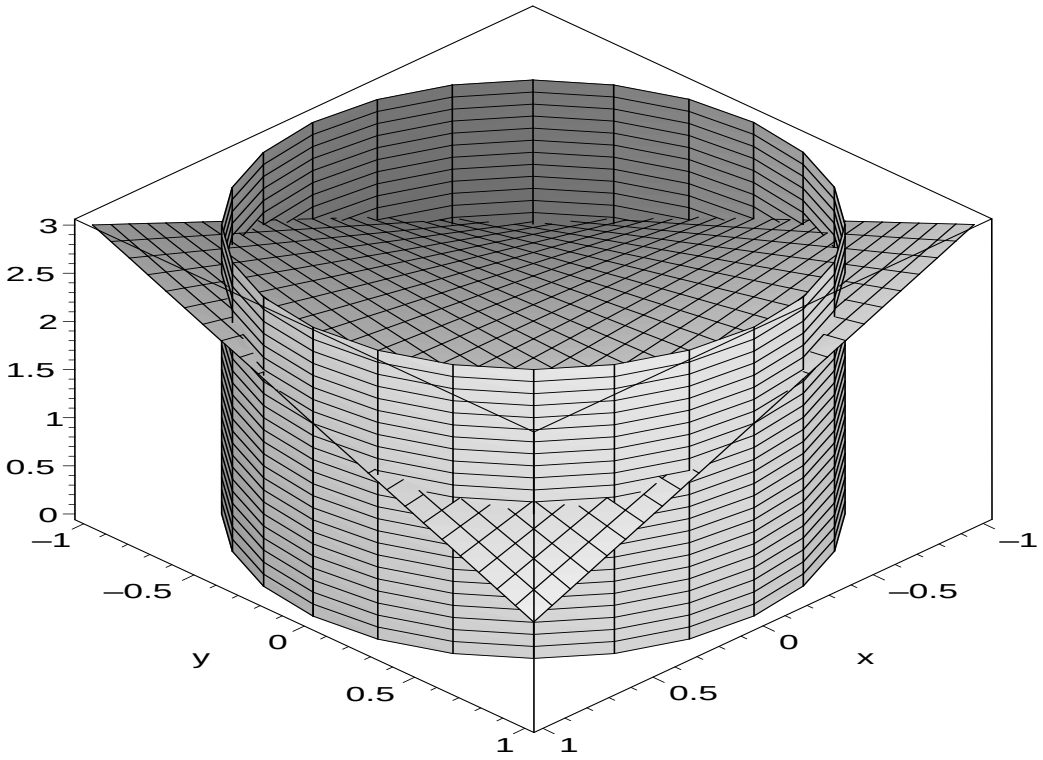
```
> plot3d(sin(theta),theta=0..Pi/2,phi=0..Pi/2,coords=spherical,
> scaling=constrained,axes=normal,labels=["x","y","z"],
> orientation=[30,70]);
```



3.4 Oppgave 5

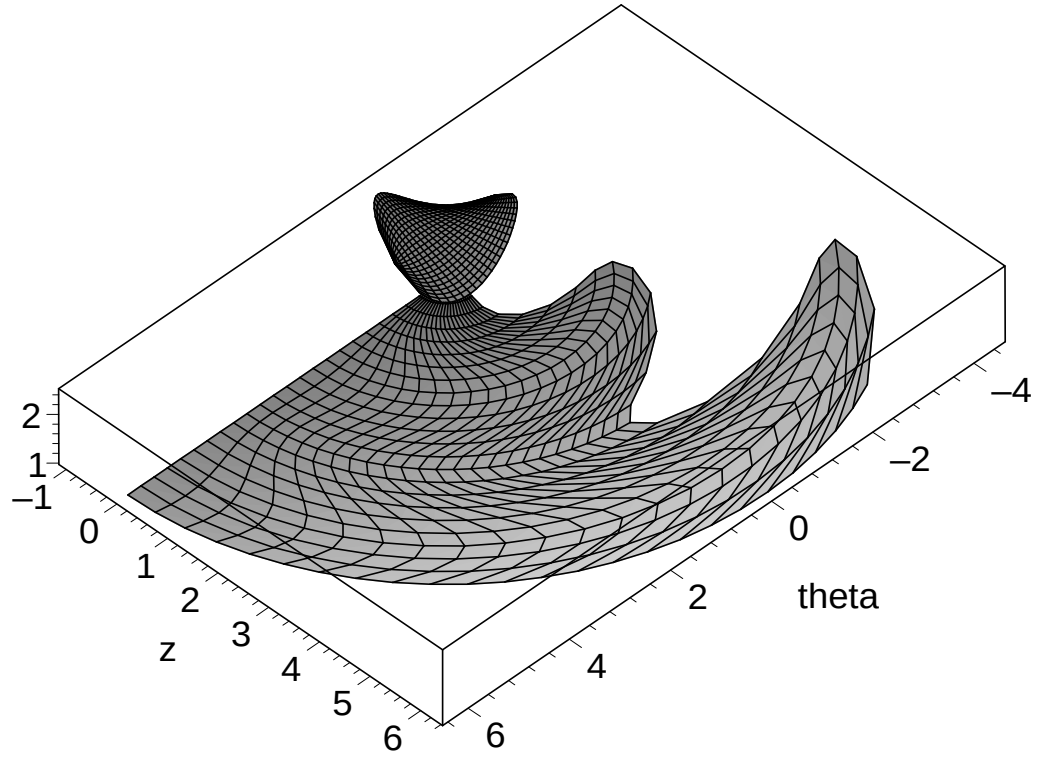
Første forsøk, bare for å få litt oversikt over situasjonen:

```
> display3d(
> plot3d(1,theta=0..2*Pi,z=0..3,coords=cylindrical),
> plot3d(2-x*y,x=-1..1,y=-1..1),
> axes=boxed);
```



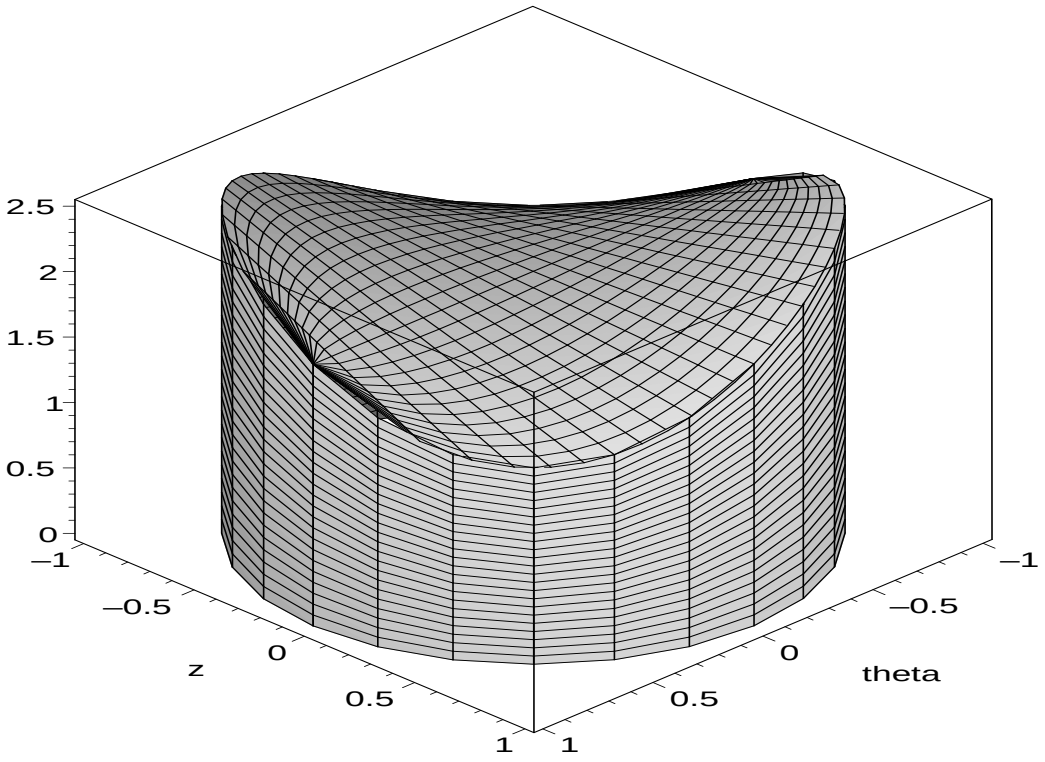
Følgende forsøk på en bedre figur avslører en **bug** i ***plot3d***:

```
> display3d(
> plot3d(1,theta=0..2*Pi,z=0..(2-cos(theta)*sin(theta)),
> coords=cylindrical),
> plot3d(2-x*y,x=-1..1,y=-sqrt(1-x^2)..sqrt(1-x^2)),
> scaling=constrained,axes=boxed);
```



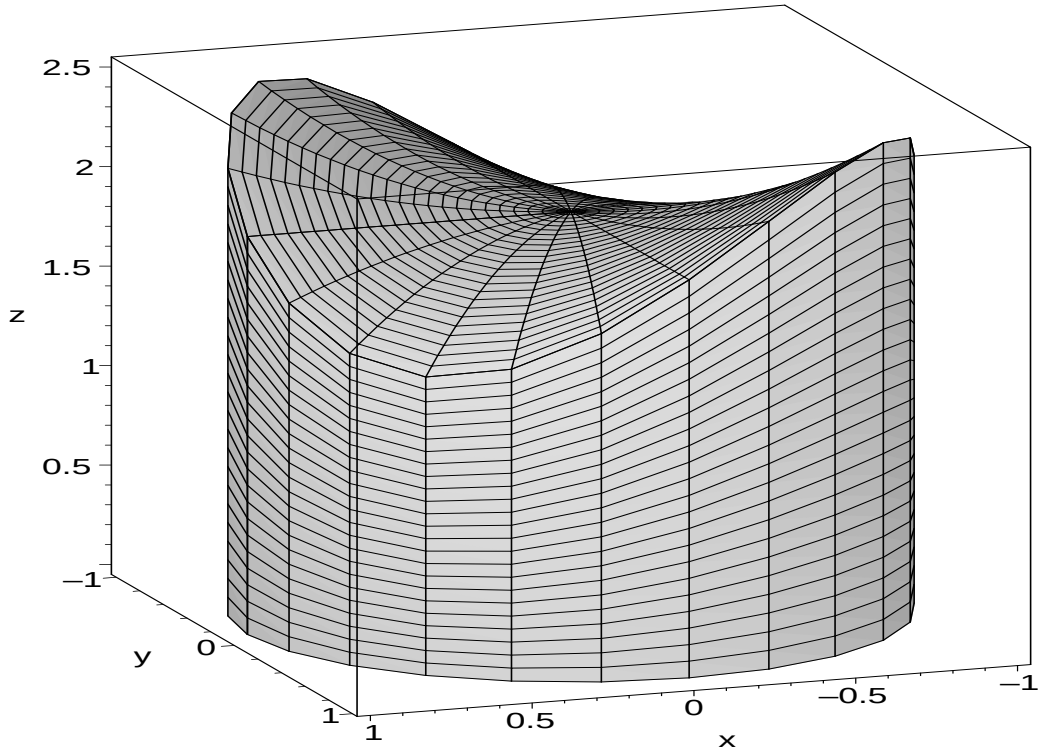
Heldigvis er det lett å omgå bugen:

```
> display3d(
> plot3d([1,theta,z],theta=0..2*Pi,z=0..(2-cos(theta)*sin(theta)),
> coords=cylindrical),
> plot3d(2-x*y,x=-1..1,y=-sqrt(1-x^2)..sqrt(1-x^2)),
> scaling=constrained,axes=boxed);
```



Igjen kan vi streve litt hardere og få en litt penere figur i sylinderkoordinater:

```
> display3d(
> plot3d([1,theta,z],theta=0..2*Pi,z=0..(2-cos(theta)*sin(theta)),
> coords=cylindrical),
> plot3d([r,theta,2-r^2*cos(theta)*sin(theta)],r=0..1,theta=0..2*Pi,
> coords=cylindrical),
> axes=boxed,labels=["x","y","z"],scaling=constrained,
> orientation=[70,70]);
```



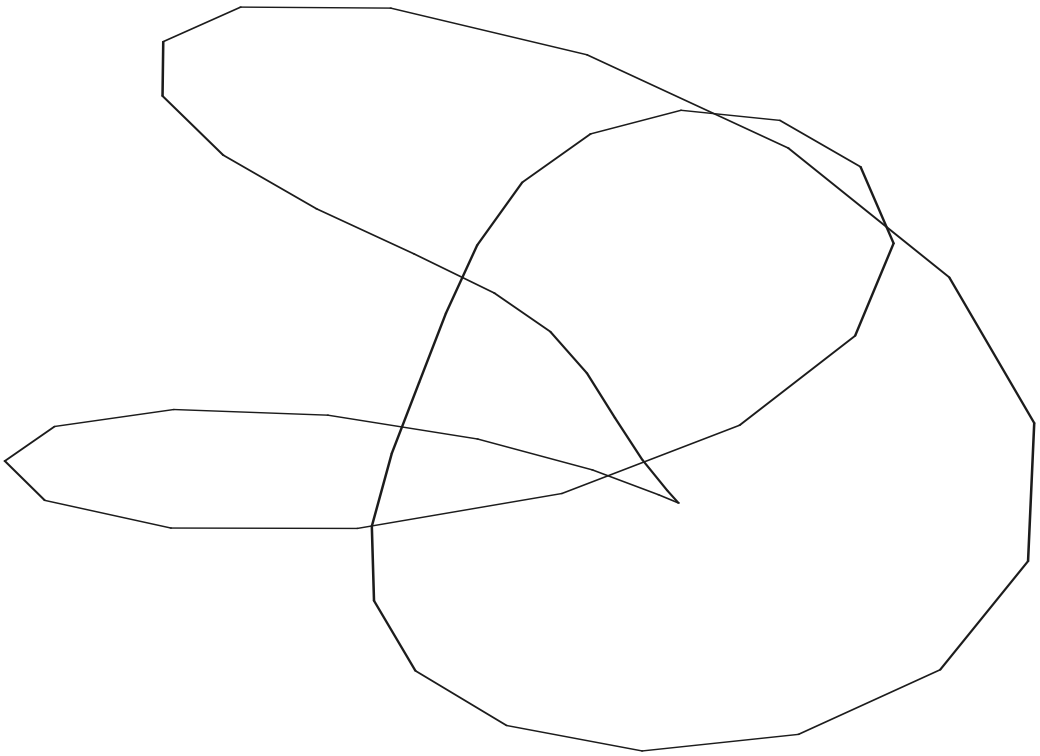
4 Oppgave 3: Visualisering av romkurver

```
> a:=3:  b:=2:
> kurve:=
> ((a+b*cos(2*t))*cos(3*t),
> (a+b*cos(2*t))*sin(3*t),
> b*sin(2*t)),
> t=0..2*Pi;
```

$$kurve := (3 + 2\cos(2\,t))\cos(3\,t),\ (3 + 2\cos(2\,t))\sin(3\,t),\ 2\sin(2\,t),\ t = 0..2\,\pi$$

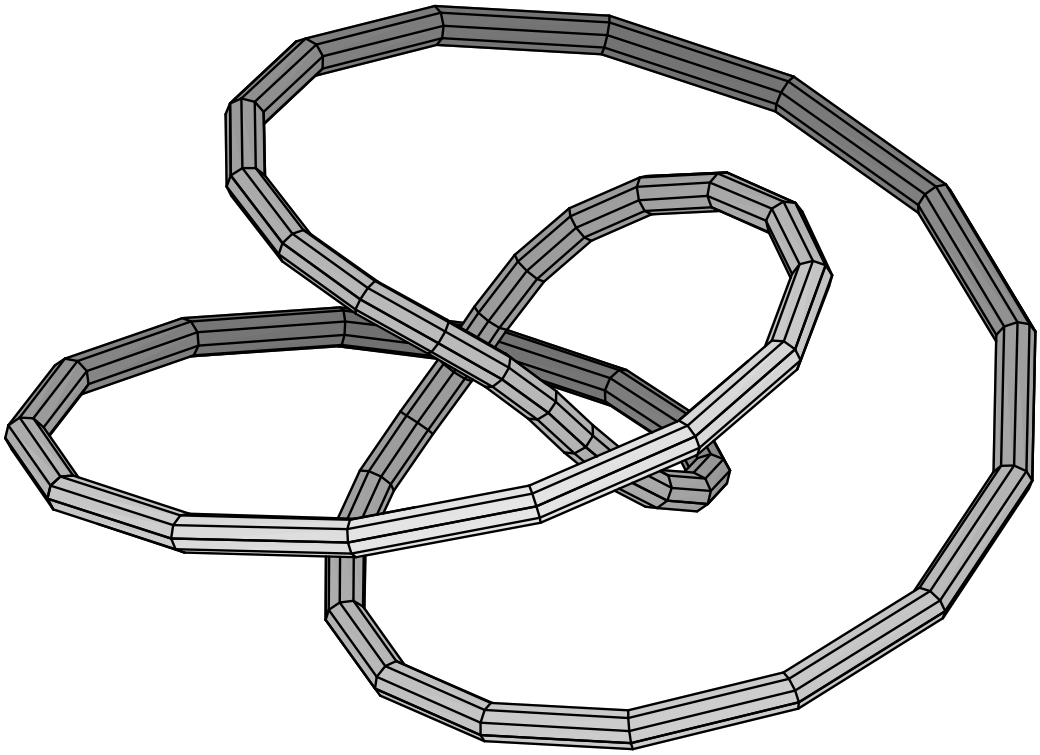
Vi prøver med ***spacecurve***:

```
> spacecurve([kurve],color=blue);
```



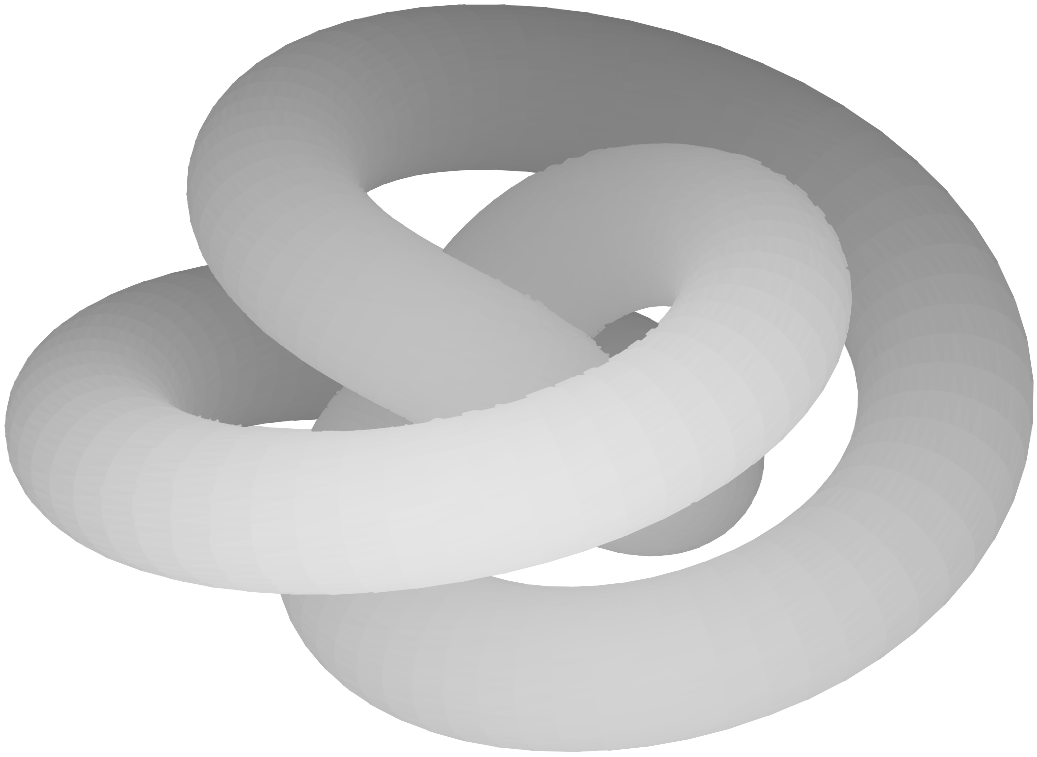
Det var jo ikke så lett å bli klok på! Litt mer skjønner man om man tar tak i figuren og dreier den litt med musa, men det er best å bruke ***tubeplot*** for å få litt tykkelse på kurven:

```
> tubeplot([kurve],radius=0.2,orientation=[65,57],
> scaling=constrained);
```



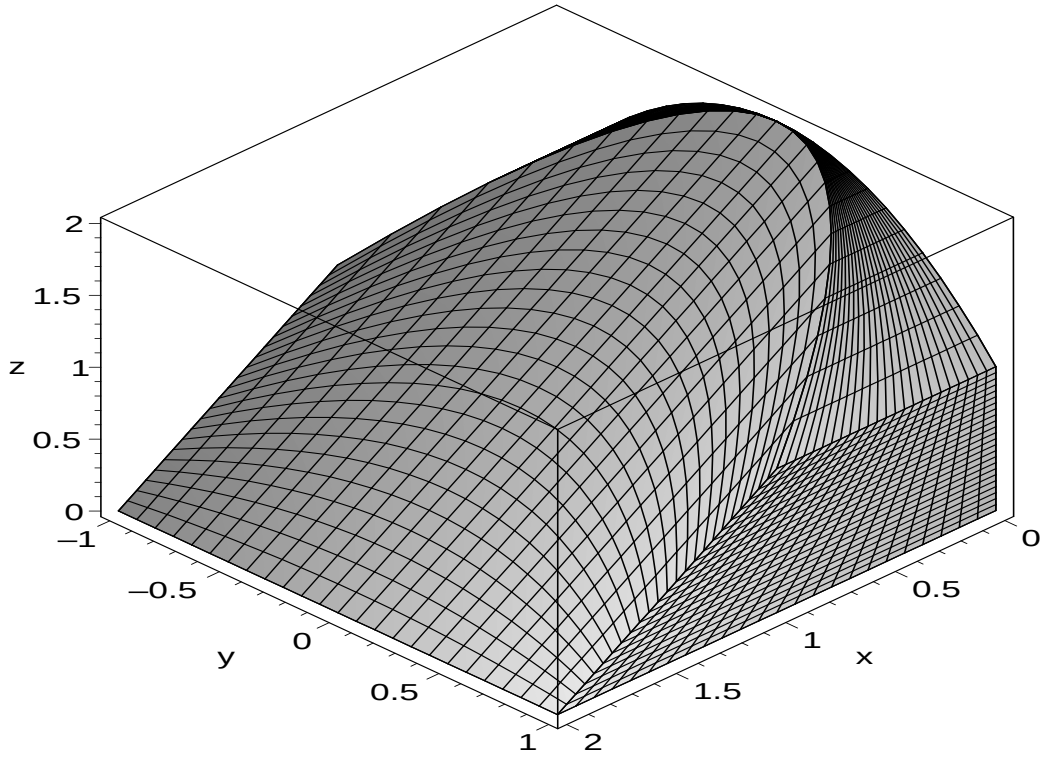
Her kommer en tykkere, mer fancy utgave:

```
> tubeplot([kurve],numpoints=200,tubepoints=50,
> style=patchnogrid,orientation=[65,57],
> scaling=constrained);
```



5 Oppgave 4: Forenklet visualisering av legemer

```
> display3d(
> plot231(y=-1..1,z=0..2-y^2,x=0..2-z),
> scaling=constrained,axes=boxed,labels=["x","y","z"]);
```

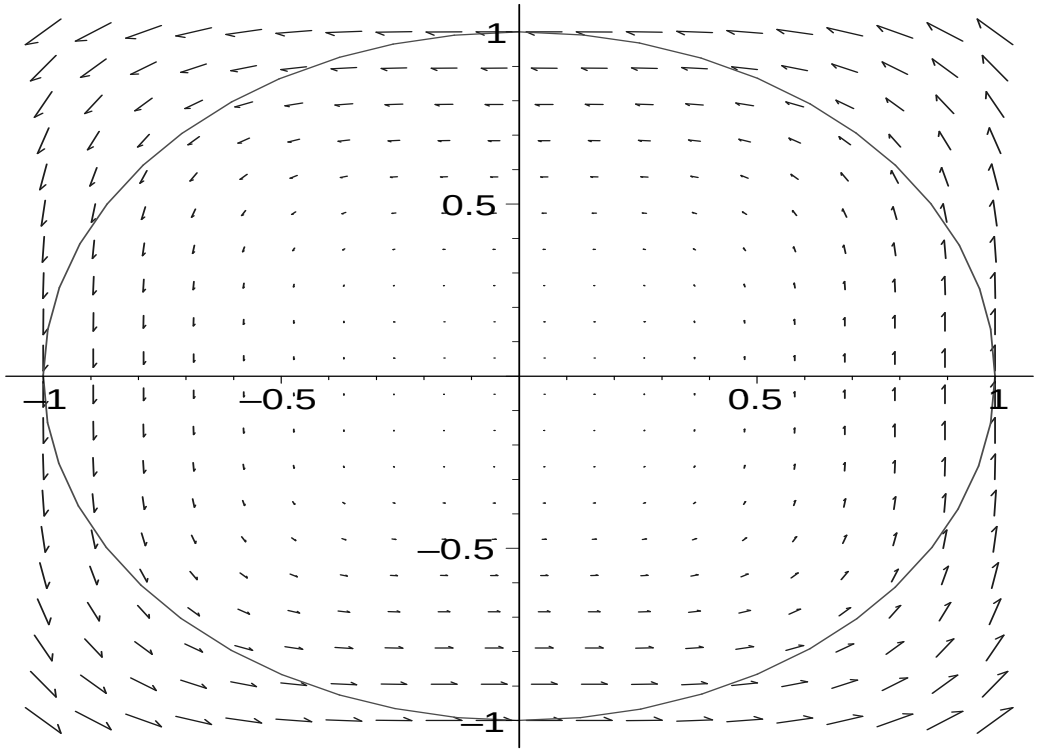


6 Oppgave 5: Visualisering av vektorfelt

6.1 Hjemmeøving 6, oppgave 2

Vektorfeltet og kurven vi integrerer rundt:

```
> display(
> plot([cos(t),sin(t),t=0..2*Pi],color=red),
> fieldplot(<-y^3,x^3>,x=-1..1,y=-1..1,color=blue),
> scaling=constrained,axes=normal);
```



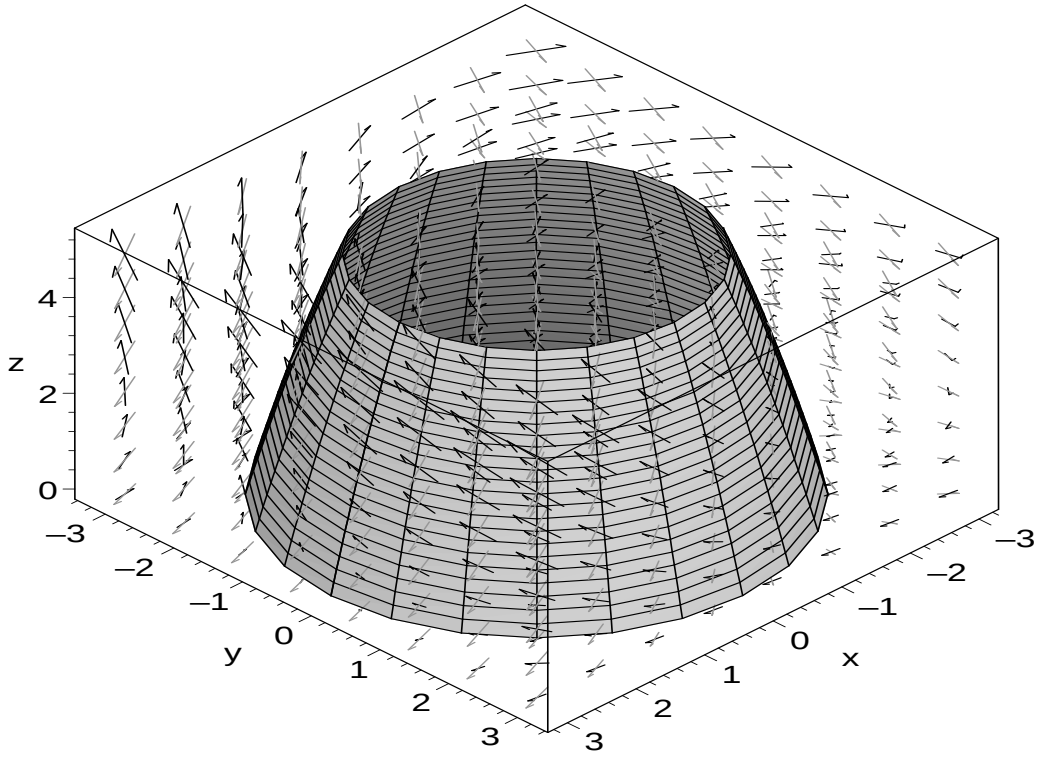
6.2 Hjemmeøving 6, oppgave 4

```
> F:=[2*y,-x*z,2*z+1];
> curlF:=linalg[curl](F,[x,y,z]);
```

$$F := [2\,y,\ -x\,z,\ 2\,z+1]$$

$$curlF := [x,\ 0,\ -z-2]$$

```
> display3d(
> plot3d([r,theta,9-r^2],r=2..3,theta=0..2*Pi,coords=cylindrical),
> fieldplot3d(F,x=-3..3,y=-3..3,z=0..5,color=black),
> fieldplot3d(curlF,x=-3..3,y=-3..3,z=0..5,color=green),
> scaling=constrained,axes=boxed);
```



Om man dreier litt på figuren ser man godt at vektorfeltet (i svart) roterer med klokka sett ovenfra, som stemmer godt med at curlen (i grønt) er rettet nedover. Av figuren ser det ut til at curl F stort sett er rettet tilnærmet parallelt med flaten. Det er vanskelig å se om det går mest inn eller ut.