Sections connues

© Pierre Lantagne
Enseignant retraité du Collège de Maisonneuve

Ce document présente plusieurs exemples de tracés de solides de sections connues. La création de ces tracés exige une assez bonne connaissance de la programmation qu'offre Maple. Ces tracés ont donc été créés dans le but d'illustrer mes documents pédagogiques seulement et non pas dans un but d'apporter un complément de formation en Maple à mes élèves du cours NYB (Calcul intégral). Tout de même, je pense que la lecture de ce document pourrait intéresser certains lecteurs désireux d'explorer les possibilités de programmation en Maple.

Bonne lecture à tous !

[*] Ce document Maple est exécutable avec la version 2020.1

Initialisation

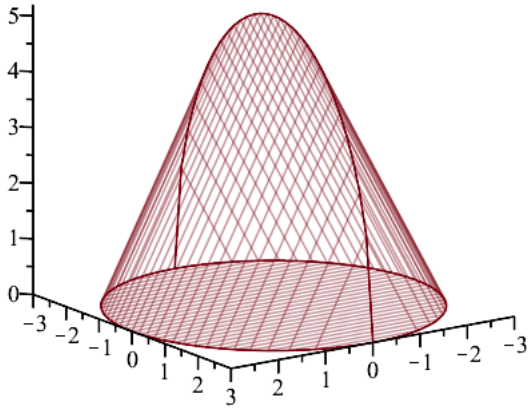
```
> restart;
> with(plots,display,polygonplot3d,spacecurve,setoptions,setoptions3d,
  surfdata):
  with(plottools,curve,line,translate,rotate):
  with(codegen,makeproc):

> setoptions3d(size=[400,400],style=line,color="Burgundy",
               scaling=constrained,
               font=[TIMES,ROMAN,9],
               axes=framed,
               orientation=[55,75]);
```

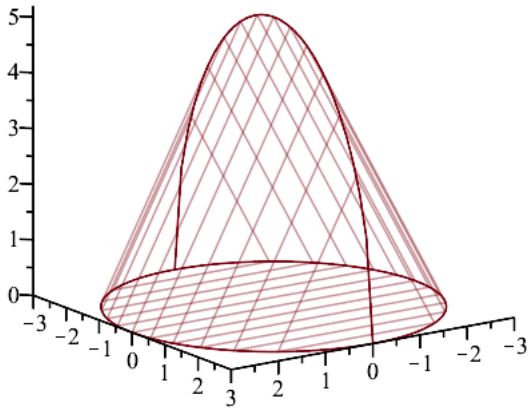
Solide de base circulaire de rayon 3 dont toute section plane perpendiculaire à l'axe des *y* est un triangle équilatral.

```
> Base_du_solide:=x^2+y^2=9:
  Abscisses:=[solve(Base_du_solide,x)]:
  Abscisse:=makeproc(Abscisses[1],y):
  Sections:=y->polygonplot3d([[Abscisse(y),y,0],
              [-Abscisse(y),y,0],[0,y,h(Abscisse(y))]]):
  h:=makeproc(sqrt(3)*y,y):
  Base:=spacecurve([3*cos(t),3*sin(t),0],t=0..2*Pi):
  Arêtes:=table():
  n:=30:
```

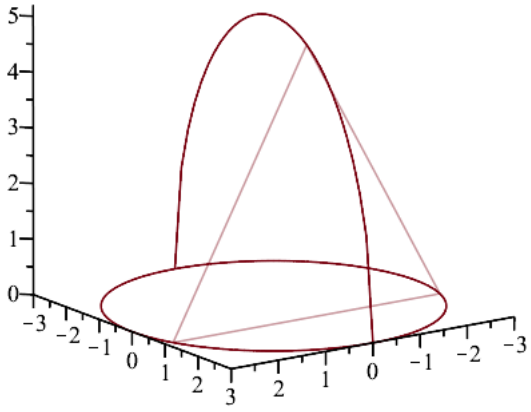
```
for i from 0 by 1 to n do
Arêtes[i]:=evalf([0,-3+6*i*1/n,h(Abscisse(-3+6*i*1/n))]) od:
ordre:=(x,y)->evalb(op(2,x)<op(2,y)): #critre d'ordonnance pour sort
Arête:=curve(sort(convert(Arêtes,list),ordre)):
Section:=table():
n:=30:
for i from 0 by 1 to n do
    Section[i]:=Sections(-3+6*i*1/n) od:
display(convert(Section,list),Base,Arête);
```



```
> Section:=table():
  n:=120:
  for i from 0 by 10 to n do  #(by 10 pour visualiser)
      Section[i]:=Sections(-3+6*i*1/n) od:
  display(convert(Section,list),Base,Arête);
```

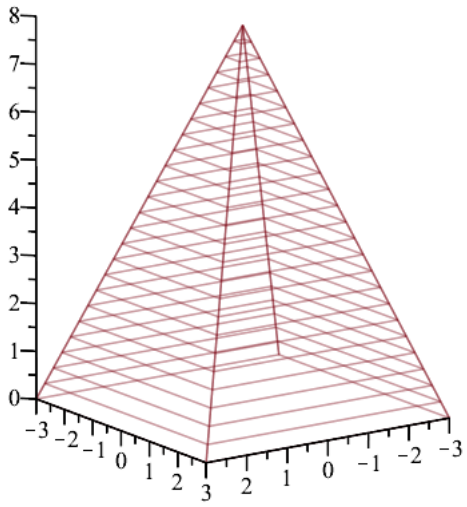


```
> display(Section[80],Base,Arête);
```

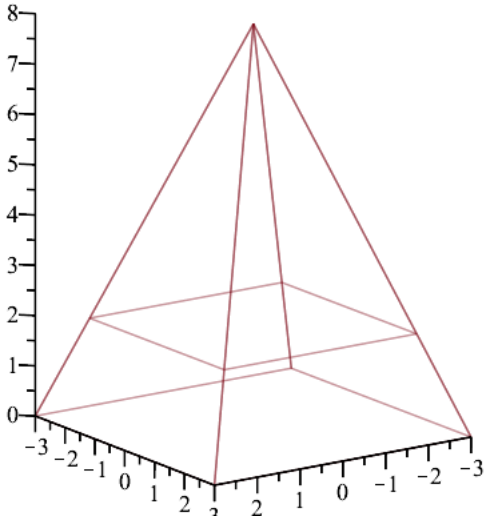


Pyramide de hauteur *h* et de base carrée de côté *c*

```
> h:=8:
  c:=6:
  p1:=[0,h]:
  p2:=[c/2,0]:
  Équation_droite:=z=(p2[2]-p1[2])/(p2[1]-p1[1])*x+h:
  Équation1:=x=-solve(Équation_droite,x):
  xval:=t->eval(rhs(Équation1),z=t):
  L:=[[xval(z),xval(z),z],
  [-xval(z),xval(z),z],
  [-xval(z),-xval(z),z],
  [xval(z),-xval(z),z]]:
  Section:=t->eval(L,z=t):
  Arêtes:=polygonplot3d([[c/2,c/2,0],[-c/2,c/2,0],[-c/2,-c/2,0],[c/2,-
  c/2,0],[c/2,c/2,0],
                [c/2,c/2,0],[0,0,h],[-c/2,c/2,0]  ,
                [-c/2,c/2,0],[0,0,h],[-c/2,-c/2,0],
                [-c/2,-c/2,0],[0,0,h],[c/2,-c/2,0],
                [c/2,-c/2,0],[0,0,h],[c/2,c/2,0]]):
  plt:=table():
  n:=24:
  for i from 1 by 1 to n do
      plt[i]:=polygonplot3d(Section(i*h/n)) od:
  display(convert(plt,list),Arêtes);
```

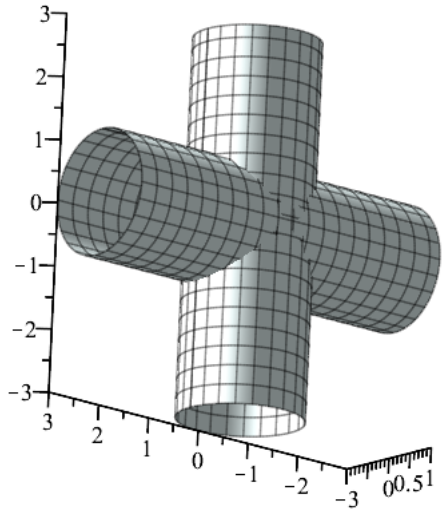


```
> display({plt[6],Arêtes});
```



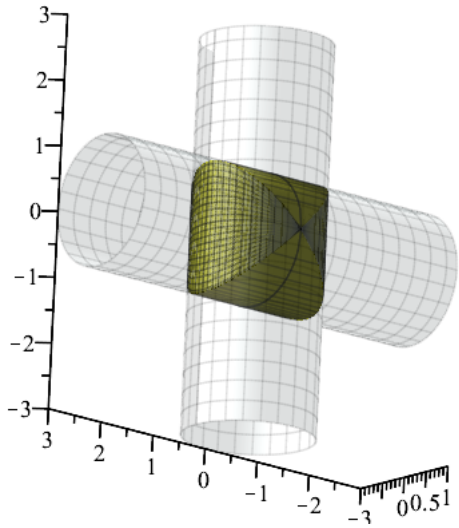
Intersection de deux cylindres de même rayon unité

```
> c1:=plot3d(1,theta=0..2*Pi,z=-3..3,coords=cylindrical,style=patch,
  color="Azure",grid=[120,60]):
> c2:=rotate(c1,0,Pi/2,0):
> display({c1,c2},axes=framed,orientation=[-140,75,170]);
```

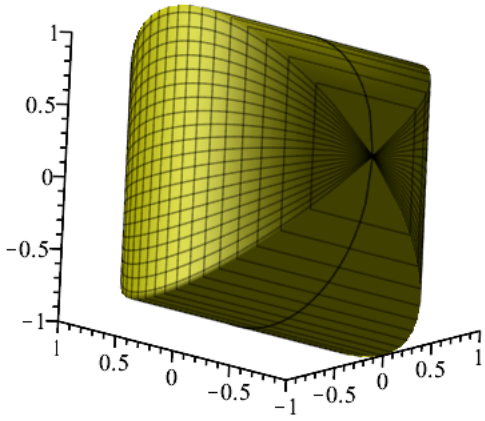


Solide d'intersection de deux cylindres de même rayon unité

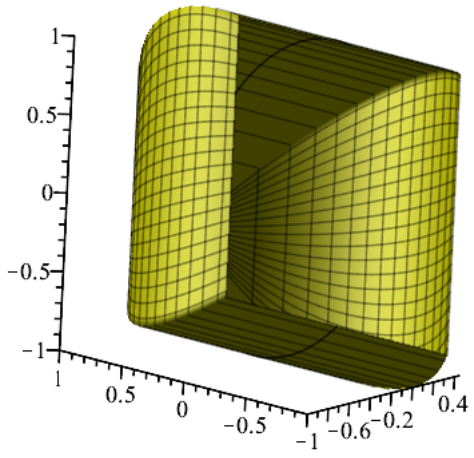
```
> h:=x->sqrt(1-x^2):
  xvals:=[seq(-1+i*2/40,i=0..40)]:
  n:=nops(xvals):
  Baseline:=proc(x) local j:
   [seq([x,0,-h(x)+j*2*h(x)/40],j=0..40)]
  end proc:
  Toplinesup:=proc(x) local j:
    [[x,0,0-h(x)],seq([x,h(x),-h(x)+j*2*h(x)/40],j=0..40),[x,0,h(x)]]
  end proc:
  Toplineinf:=proc(x) local j:
    [[x,0,-h(x)],seq([x,-h(x),-h(x)+j*2*h(x)/40],j=0..40),[x,0,h(x)]]
  end proc:
  Circle:=[seq(Baseline(xvals[i]),i=1..n)]:
  Topsup:=[seq(Toplinesup(xvals[i]),i=1..n)]:
  Topinf:=[seq(Toplineinf(xvals[i]),i=1..n)]:
  Inter:=surfdata([Topsup,Topinf],style=patch,color="Olive"):
  c1:=plot3d(1,theta=0..2*Pi,z=-3..3,coords=cylindrical,style=patch,
  color="Azure",grid=[120,60],transparency=0.75):
  c2:=rotate(c1,0,Pi/2,0):
  display([c1,c2,rotate(Inter,0,Pi,Pi/2)],axes=framed,orientation=
  [-140,75,170]);
```



```
> display(rotate(Inter,0,Pi,Pi/2),axes=framed,orientation=[-140,75,170]
  );
```



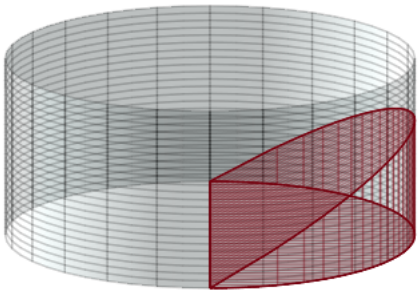
```
> Topsup:=[seq(Toplinesup(xvals[i]),i=1..30)]:
  Topinf:=[seq(Toplineinf(xvals[i]),i=1..30)]:
  Inter_bis:=surfdata({Topsup,Topinf},style=patch,color="Olive"):
  display(rotate(Inter_bis,0,Pi,Pi/2),axes=framed,orientation=[-140,75,
  170]);
```



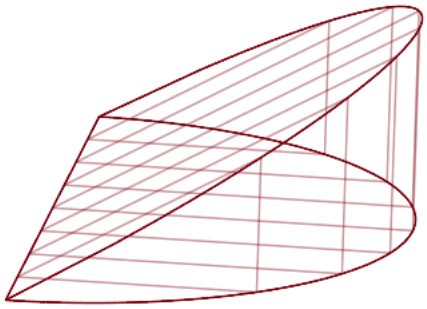
Entaille pratiquée à l'aide de deux plans dans un cylindre circulaire droit de rayon 9 cm.
Le premier plan est paralèlle à la base du cylindre et le second fait un angle de 30° avec la base

```
> Cercle:=x^2+y^2=81:
  Rayon:=sqrt(rhs(Cercle)):
  Sol:=solve(Cercle,y) assuming y>=0:
  yval:=t->subs(x=t,Sol[1]):
  yval(1):
  Pts:=x->evalf([
  [x,0,0],
  [x,yval(x),0],
  [x,yval(x),tan(Pi/6)*yval(x)]]):
  Base:=spacecurve({[9*cos(t),9*sin(t),0],[[-Rayon,0,0],[Rayon,0,0]]},
  t=0..Pi):
  c1:=plot3d(9,theta=0..2*Pi,z=-0..8,coords=cylindrical,style=patch,
  color="Azure",transparency=0.5):
  n:=24:
  Arêtes:=table():
  for i from -n by 1 to n do
      Arêtes[i]:=evalf(op(3,Pts(i*Rayon/n))) od:
  ordre:=(x,y)->evalb(op(1,x)<op(1,y)):
  Arête:=curve(sort(convert(Arêtes,list),ordre)):
  #a:=rotyplot(x->0,x=0..9,x=0);
  n:=18:
  Section:=table():
  for i from -n by 1 to n do   #(by 6 pour visualiser)
```

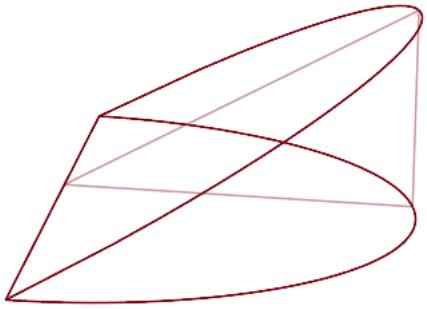
```
    Section[i]:=polygonplot3d(Pts(i*Rayon/n)) od:
display(convert(Section,list),Base,Arête,c1,orientation=[0,75,0],
axes=none);
```



```
> n:=24:
  Section:=table():
  for i from -n by 6 to n do   #(by 6 pour visualiser)
      Section[i]:=polygonplot3d(Pts(i*Rayon/n)) od:
  display(convert(Section,list),Base,Arête,orientation=[14,75,7],axes=
  none);
```

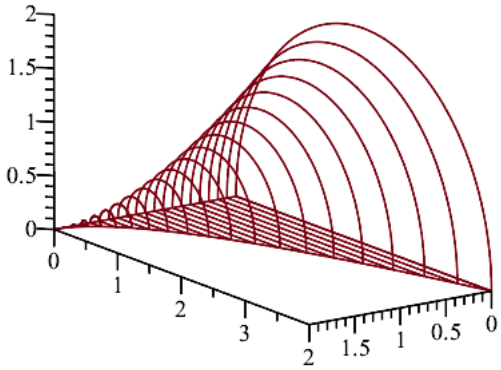


```
> display(Section[-6],Base,Arête,orientation=[14,75,7],axes=none);
```

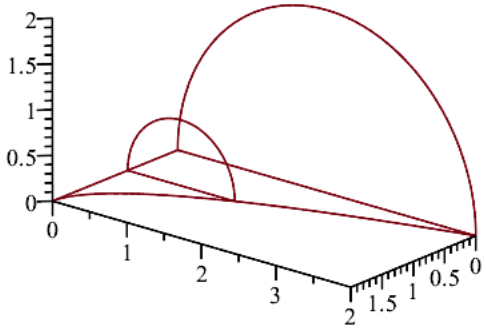


Sections connues Exemple 1 page 229

```
> Rayon:= proc(x) local j:
          (x-2)^2/2
          end:
  n:=20:
  Section:=table():
  for i from 0 to n do
  Section[i]:=spacecurve([0,Rayon(i*2/n)*cos(t),Rayon(i*2/n)*sin(t)],t=
  0..Pi) od:
  Limite:=spacecurve([[t,(t-2)^2,0],[t,0,0],[0,t^2,0]],t=0..2):
  Base:=table():
  for i from 0 to n do
  Base[i]:=line([i*2/n,0,0], [i*2/n,((i*2/n)-2)^2,0]) od:
  display({seq(translate(Section[i],i*2/n,Rayon(i*2/n),0),i=0..n),
  Limite},convert(Base,list));
```



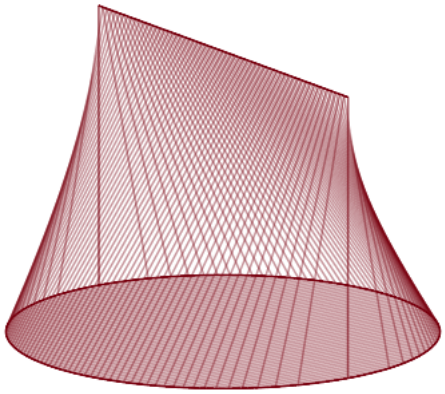
```
> display({translate(Section[0],0,2,0),
            translate(Section[8],8*2/n,Rayon(8*2/n),0),Limite,Base[8]}
  ,orientation=[40,70,0]);
```



Conoïde de cercle

```
> unassing('x','y'):
> Base_du_solide:=x^2+y^2=9:
  Abscisses:=[solve(Base_du_solide,x)]:
  Abscisse:=codegen[makeproc](Abscisses[1],y):
  Sections:=y->polygonplot3d(evalf([
    [Abscisse(y),y,0],
    [-Abscisse(y),y,0],
    [0,y,h(Abscisse(y))]
    ])):
```

```
h:=codegen[makeproc](4,y):
Base:=spacecurve([3*cos(t),3*sin(t),0],t=0..2*Pi):
Arêtes:=table():
n:=60:
for i from 0 by 1 to n do
Arêtes[i]:=evalf([0,-3+6*i*1/n,h(Abscisse(-3+6*i*1/n))]) od:
ordre:=(x,y)->evalb(op(2,x)<op(2,y)):
Arête:=curve(sort(convert(Arêtes,list),ordre)):
Section:=table():
n:=60:
for i from 0 by 1 to n do
    Section[i]:=Sections(-3+6*i*1/n) od:
display(convert(Section,list),Base,Arête,axes=none);
```



```
> Section:=table():
  n:=120:
  for i from 0 by 10 to n do  #(by 10 pour visualiser)
      Section[i]:=Sections(-3+6*i*1/n) od:
  display3d(convert(Section,list),Base,Arête,axes=none);
```