Mathematical Experiments by Mupad
by Zebang Li
2015

第五章 曲线曲面积分

Mupad是Matlab内置的符号计算工具箱，是对以数值计算闻名的matlab在符号计算方面的有力补充：
在基本运算方面，具有更好的显示界面和更快捷的命令调用；
在作图方面，命令更加简单，图形更加优化，特别在动画制作方面具有强大优势；
在窗口界面方面，Mupad更是内置了文档编辑功能，使得其可以直接如同word和ppt一样编写报告，并进行展示，进一步可直接生成pdf，省去了latex的编译过程（本文就是直接在Mupad Notebook界面编写的）。

Mupad对于数学实验无疑具有更大优势，而目前国内关于Mupad的工具箱介绍得很少，Goole搜索几乎没有，知网鲜有相关论文，这使得人们对于其不甚了解，鉴于此，希望能对于Mupad的应用进行一些探索，作为对于传统Matlab操作的有力补充。

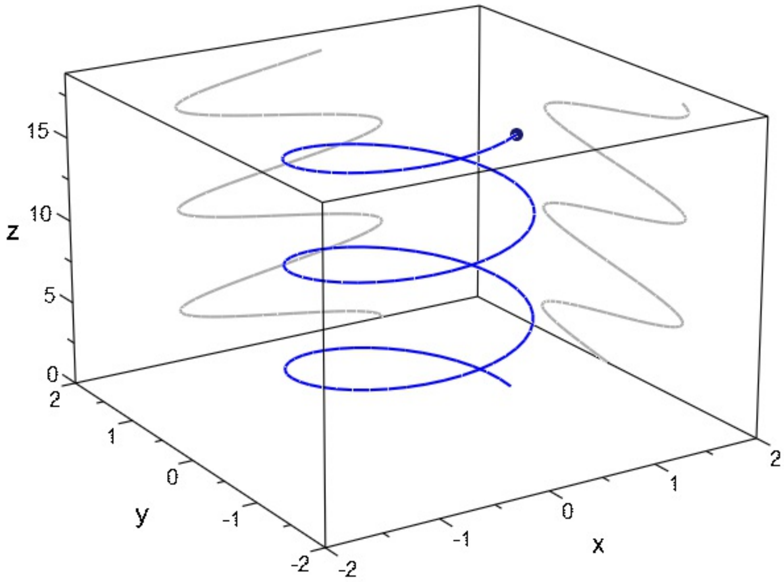
值得一提的是，这并不是一个全新的软件，它是Matlab中的符号计算工具箱，本文不是力图对于一种新软件的简绍，而是区别于传统基于Matlab主界面的数学实验，基于这种Matlab自带的符号计算引擎的一些优势，着重于数学直观的展示，介绍一种探究直观，理解概念的平台，给出一些例子，期望读者能够在此基础上，自行挖掘和研究更多数学直观，作为抛砖引玉，相信这对于教学，理解，探索都是有益的。

下面基于曲线曲面积分讲义里的一些例题和概念，对Mupad的一些应用进行展示。

例1.　基于计算螺旋线 $\sigma(t) = \begin{pmatrix} x = a\cos(t) \\ y = a\sin(t) \\ z = b\,t \end{pmatrix}$, $t = 0..\pi,\ 0 < a,\ 0 < b$ 的弧长引出的一系列展示.

首先，给出螺旋线的几何直观

```
x :=a*cos(t):
y :=a*sin(t):
z :=b*t:

a :=1://假设a=1
b :=1://假设b=1

plot(plot::Curve3d([x, y, z],t = 0..k, k=0..6*PI),
     plot::Point3d([x, y, z],t = 0..6*PI, PointSize = 3*unit::mm),
     plot::Curve3d([2,y(t),z(t)],  t = 0..k, k=0..6*PI, Color =
RGB::LightGrey),
     plot::Curve3d([x(t),2,z(t)], t = 0..k, k=0..6*PI,  Color =
RGB::LightGrey),
     ViewingBox = [-2..2, -2..2, 0..6*PI])
```



这里是基于Mupad给出的动画演示，且在x=2和y=2的平面上给出了投影展示。在Mupad界面，可以自行调节视角，进程等，进一步查看图形细节。

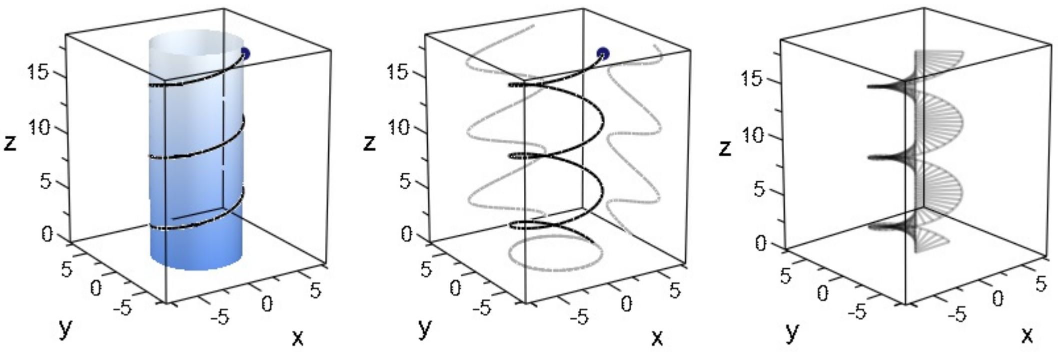
其中，动画的生成是Mupad一个十分便捷的功能，关于动画制作的常见例子及其用法，参见Animation的帮助文档。（注：鉴于目前国内对于这一工具的介绍十分稀少，学会参阅帮助文档显得及其重要）

这里给出的命令并不难，首先给出表达式，然后用plot作图命令分别作图，其中的参数k用于生成动画，其余出现的调节命令是对图形的进一步修饰，例如PointSize调节首端球点的大小,Color调节图形颜色,ViewingBox调节窗口大小，关于这些命令的细节可以在help文档中查看。（注：帮助文档的调出有多种方法，常见的一种是，例如，在命令行输入：? plot 就可以调出plot的帮助文档）

在认真参阅了帮助文档，或是附录的学习指引之后，我们就可以自行尝试着，给出其它表现力丰富的展示了。

```
x :=a*cos(t):
y :=a*sin(t):
z :=b*t:

a :=4:
b :=1:

s1 := plot::Scene3d(plot::Curve3d([x, y, z],t = 0..k, k=0..6*PI,Color=RGB::Black),
                    plot::Point3d([x, y, z],t = 0..6*PI,PointSize = 3*unit::mm),
                    plot::Cylindrical([4, phi, z], phi = 0 .. 2*PI, z = 0..6*PI,FillColor  =RGB::AliceBlue.[0.2],
                                    ULinesVisible = FALSE, VLinesVisible = FALSE),
                    ViewingBox = [-8..8, -8..8, 0..6*PI]):

s2 := plot::Scene3d(plot::Curve3d([x, y, z],t = 0..k, k=0..6*PI,Color=RGB::Black),
                    plot::Point3d([x, y, z],t = 0..6*PI,PointSize = 3*unit::mm),
                    plot::Curve3d([8,y(t),z(t)],t = 0..k,
```

```
k=0..6*PI,Color = RGB::LightGrey),
                    plot::Curve3d([x(t),8,z(t)],t = 0..k,
k=0..6*PI,Color = RGB::LightGrey),
                    plot::Curve3d([x(t),y(t),0],t = 0..k,
k=0..6*PI,Color = RGB::LightGrey),
                    ViewingBox = [-8..8, -8..8,
0..6*PI],Scaling=Constrained):

s3 := plot::Scene3d(plot::Sweep([x,y, z], [0,0,z], t = 0..k, k=0..6*PI,
Mesh = 100,Filled=FALSE,
                    ViewingBox = [-8..8, -8..8,
0..6*PI],Scaling=Constrained)):

plot(s1,s2,s3,Layout = Horizontal,Height = 10*unit::cm, Width =
20*unit::cm)
```



这里给出了圆柱面和螺旋面上的螺旋线。

关于命令的说明：plot::Curve3d是曲线作图，plot::Cylindrical是基于参数的柱面作图，plot::Sweep是基于参数曲线的面作图，plot::Scene3d用于合并作图，Scaling=Constrained用于保持比例尺不变，Layout = Horizontal是控制三个图形水平放置。

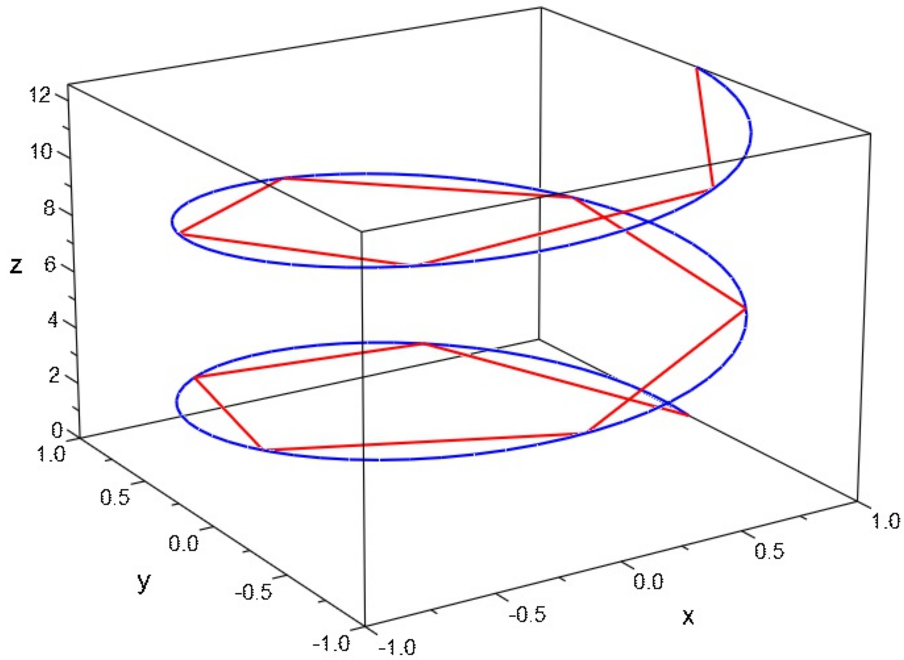
关于这些命令的用法细节，是简单的，且在帮助文档中已经有了详尽的阐释，这里就不再一一做重复的罗列了，大家可以自行查看帮助文档或附录的学习指引。

回顾一下曲线弧长的定义：设 $\sigma:[\alpha,\beta]\to R^n$ 是一条参数曲线 L，对于 $[\alpha,\beta]$ 的任意分法 $T:\alpha=t_0<t_1<\ldots<t_m=\beta$，称 $S(L,T)=\sum_{k=1}^{m}\left|\sigma\left(t_{k-1}\right)-\sigma\left(t_k\right)\right|$ 为依次连接L上的点 $\sigma(t_0),\sigma(t_1),\ldots,\sigma(t_m)$ 形成的折线的长度，称 $s=|L|=\sup\{S(L,T):T\text{为}[\alpha,\beta]\text{的分法}\}$为曲线L的弧长。

下面我们给出这种基于分法加细的几何直观（在Mupad界面可以查看动态展示）

```
x:=a*cos(t):
y :=a*sin(t):
```

```
z :=b*t:

a :=1:
b :=1:

Myframe :=  plot::Curve3d([x, y, z],t = 0..4*PI):

for i from 0 to 61 do
    u[i] := i/3;
end_for:
for i from 0 to 60 do
  myframe[i] :=  plot::Curve3d([x, y, z],t = 0..4*PI,UMesh =4+i,
LineColor = RGB::Red,VisibleFromTo = u[i]..u[i + 1]);
end_for:

plot(Myframe,myframe[i] $ i = 0..60,
     Height = 9*unit::cm, Width = 16*unit::cm)
```



这是一个基本的利用for循环语句制作动画的例子，是常见的“frame by frame”的动画制作方式，虽然这不是Mupad动画制作的最优方式，但作为Matlab的传统动画制作方式，在Mupad中有时也使用（关于动画制作的几种基本方法，参见Animation的帮助文档）其中UMesh表示参数曲线上取点的个数。

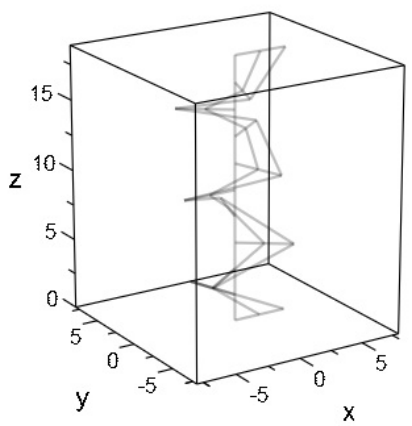
有了这些直观理解，运用弧长公式 $|L| = \int_{\alpha}^{\beta} \left|(\sigma(t))'\right| \, \mathrm{d}t$ 计算原题目中螺旋线的弧长就不是难事了。

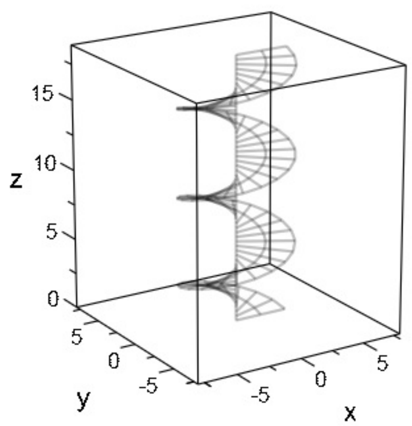
```
x :=a*cos(t):
y :=a*sin(t):
z :=b*t:

L = int(sqrt(a^2*sin(t)^2+a^2*cos(t)^2+b^2), t = 0..2*PI)
```

$$L = 2\,\pi\,\sqrt{a^2 + b^2}$$

进一步的基于螺旋线，给出螺旋面的展示及其逼近是类似的，读者可以类似的自行尝试，下面给出一种动画展示。

```
x := 4*cos(t):
y := 4*sin(t):
z := 1*t:
for i from 0 to 11 do
  myframe[i]:=(plot::Surface([4*r*cos(t),4*r*sin(t), t],r=0..1, t =
0..6*PI,
                              UMesh=3,VMesh = 6+5*i,Filled=FALSE,
                              VisibleFromTo = i..i + 1,ViewingBox
=[-8..8,-8..8,0..6*PI],
                              Scaling=Constrained)):
end_for:
S1 := plot::Scene3d(myframe[i] $ i = 0..11):
S2 := plot::Scene3d(plot::Surface([4*r*cos(t),4*r*sin(t), t],r=0..1,
t=0..6*PI,
                         TimeRange=0..6, UMesh = 3,VMesh=60,Filled=FALSE,
     ViewingBox = [-8..8, -8..8, 0..6*PI],Scaling=Constrained)):
plot(S1,S2, Layout=Horizontal)
```





例2. 计算三叶玫瑰线 L: $r = a \sin(3\theta)$, $\theta \in [0, 2\pi]$ 弧长。

首先推导出极坐标系下的弧长计算公式。

对于参数表达 r : -> ,

对于参数表达 $r: \theta \to \begin{pmatrix} \cos(\theta)\, r(\theta) \\ r(\theta) \sin(\theta) \end{pmatrix}$,

有弧长微元

$$ds = |r'(\theta)|\,d\theta = \sqrt{(r \cos(\theta) + \sin(\theta)\, r'_\theta)^2 + (r \sin(\theta) - \cos(\theta)\, r'_\theta)^2}\,d\theta = \sqrt{r'(\theta)^2 + r(\theta)^2}\,d\theta$$

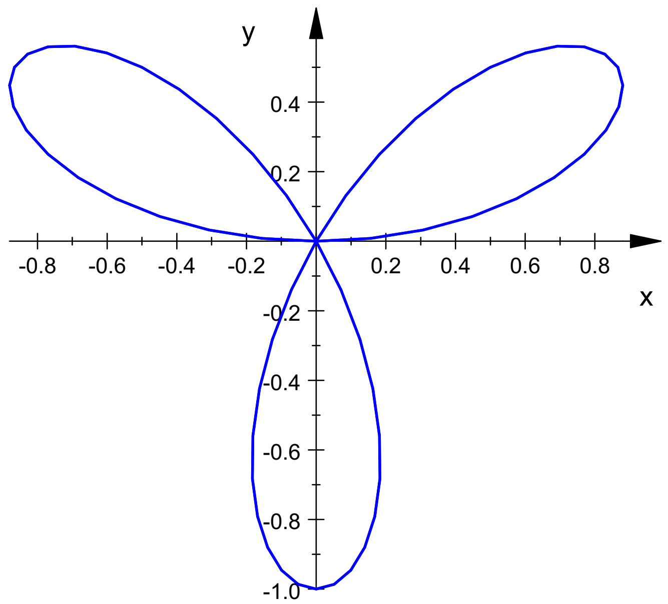
于是，可以计算出三叶玫瑰线弧长

```
r:=a*sin(3*t):
s := int(sqrt(diff(r,t)^2+r^2),t=0..2*PI)
```

$$\int_0^{2\pi} \sqrt{9\, a^2 \cos(3\, t)^2 + a^2 \sin(3\, t)^2}\, \mathrm{d}\, t$$

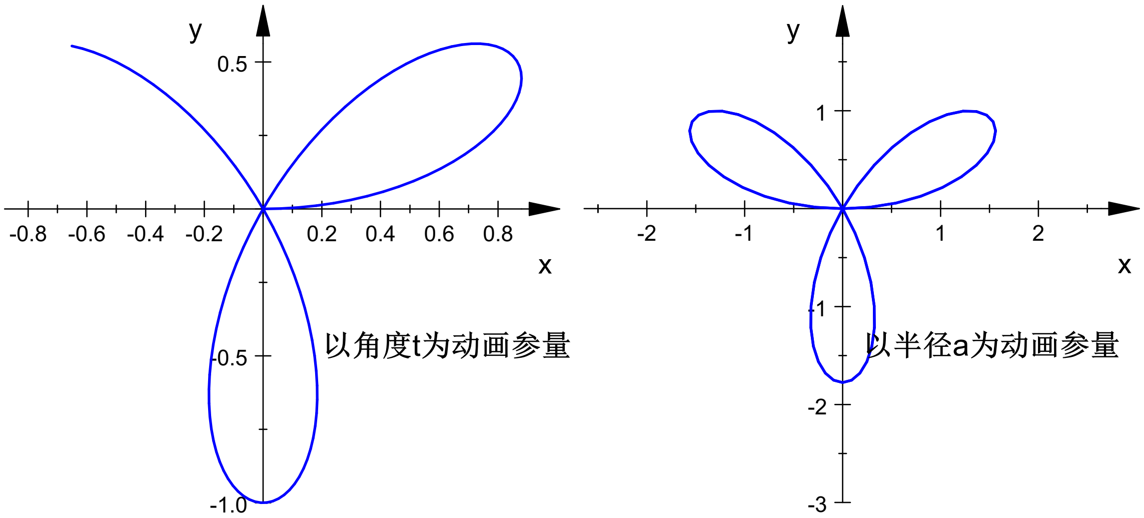
然后做出三叶玫瑰线图形（取a=1）。

```
plot(plot::Polar([sin(3*t), t], t = 0..2*PI))
```



稍作改变就可以生成基于不同参数的动画，从而理解各个参数对于图形的影响。

```
s1:= plot::Scene2d(plot::Polar([sin(3*t), t], t = 0..k,k=0..2*PI),
                   plot::Text2d("以角度t为动画参量",[0.2,-0.5])):
s2:= plot::Scene2d(plot::Polar([a*sin(3*t),t],t=0..2*PI,a=1..3),
                   plot::Text2d("以半径a为动画参量",[0.2,-1.5])):
plot(s1,s2,Layout=Horizontal)
```



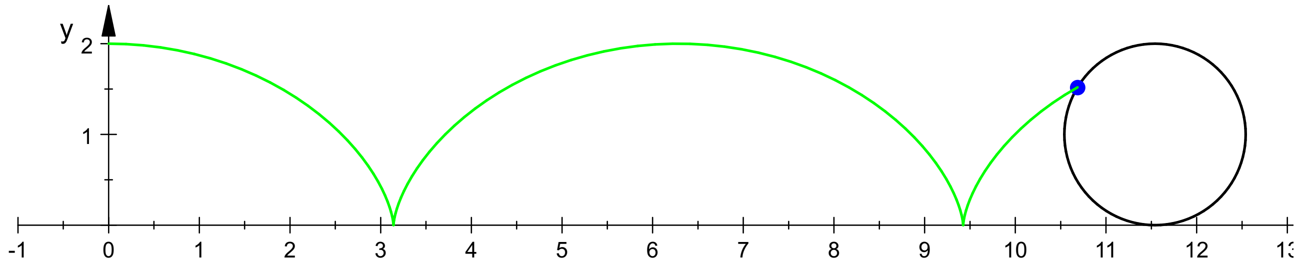
可以看见，Mupad对于动画制作具有强大功能，很简单的命令，就可以实现动画要求。

例3.摆线的展示

刚才介绍了一般动画制作，类似的，现在再运用到摆线的展示上。

首先给出基于Mupad的摆线动画命令。

```
R := 1:
C := [x, R]:
Rim := plot::Circle2d(R, C, x = 0..4*PI,
                              LineColor = RGB::Black):

Point := plot::Point2d([x + sin(x),R + cos(x)],
                              x = 0..4*PI,
                              PointColor = RGB::Blue,
                              PointSize = 2.0*unit::mm):

Cycloid := plot::Curve2d([y + sin(y), R + cos(y)],
                              y = 0..x, x = 0..4*PI,
                              LineColor = RGB::Green):

plot(Rim,Point,Cycloid):
```

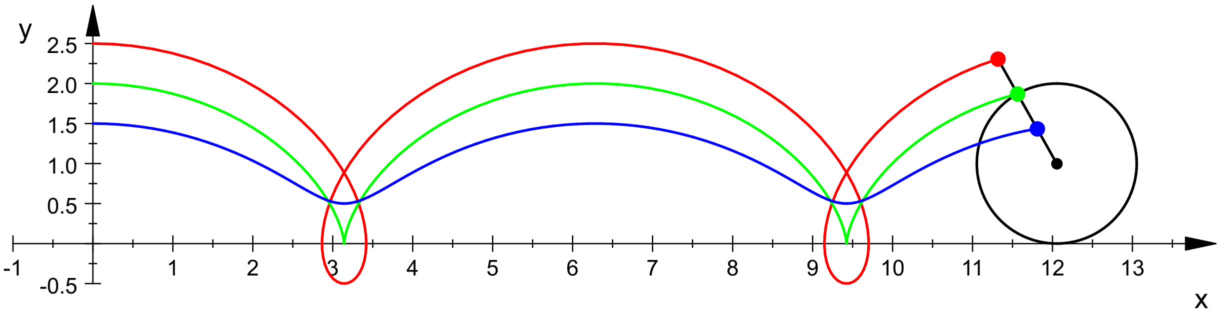


这个例子可以看出Mupad对于制作多对象动画仍然有很强大的功能。
它的几个命令十分简洁（事实上还可以更加简洁）。其中LineColor，PointColor，PointSize都是附加命令，为了图形美观而已，其核心命令就是于三个对象：摆线，点，动圆。三个带动画参量的plot就直接完成。

有兴趣的读者还可以在展示动画中增添一些对象，得到表现力更加丰富的动画，下面是“帮助文档”中的一个例子。

```
WheelRadius := 1:
WheelCenter := [x, WheelRadius]:
WheelRim := plot::Circle2d(WheelRadius, WheelCenter,
                           x = 0..4*PI,
                           LineColor = RGB::Black):
WheelHub := plot::Point2d(WheelCenter, x = 0..4*PI,
                          PointColor = RGB::Black):
WheelSpoke := plot::Line2d(WheelCenter,
   [WheelCenter[1] + 1.5*WheelRadius*sin(x),
    WheelCenter[2] + 1.5*WheelRadius*cos(x)],
   x = 0..4*PI, LineColor = RGB::Black):
color:= [RGB::Red, RGB::Green, RGB::Blue]:
r :=  [1.5*WheelRadius, 1.0*WheelRadius, 0.5*WheelRadius]:
for i from 1 to 3 do
 Point[i] := plot::Point2d([WheelCenter[1] + r[i]*sin(x),
                            WheelCenter[2] + r[i]*cos(x)],
                            x = 0..4*PI,
                           PointColor = color[i],
                           PointSize = 2.0*unit::mm):
 Cycloid[i] := plot::Curve2d([y + r[i]*sin(y),
                              WheelRadius + r[i]*cos(y)],
                             y = 0..x, x = 0..4*PI,
                             LineColor = color[i]):
```

```
end_for:
plot(WheelRim, WheelHub, WheelSpoke,
     Point[i] $ i = 1..3,
     Cycloid[i] $ i = 1..3,
     Scaling = Constrained,
     Width = 120*unit::mm, Height = 60*unit::mm):
```

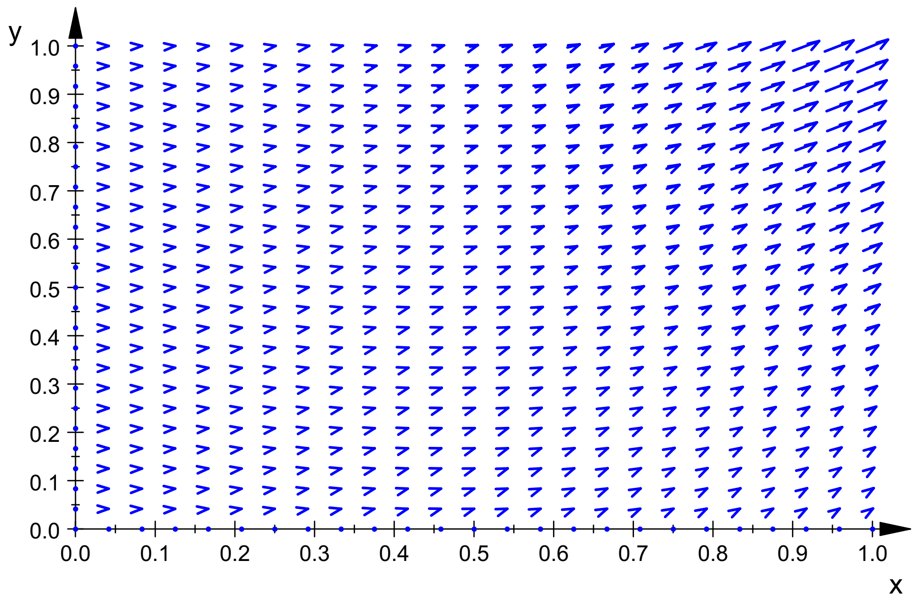


例4. 第二型线积分的一个例子： $F=(3x^2y^2, 2x^3y)$， L 从 $A(0, 1)$ 到 $B(1, 1)$ ，

(i)沿曲线 $y=x^3$； (ii)沿曲线 $y=x^2$； (iii)沿折线AC,CB,其中C(0,1)

首先作出力场 $F=(3x^2y^2, 2x^3y)$ 在区间 $[0, 1]\times[0, 1]$ 上的向量场

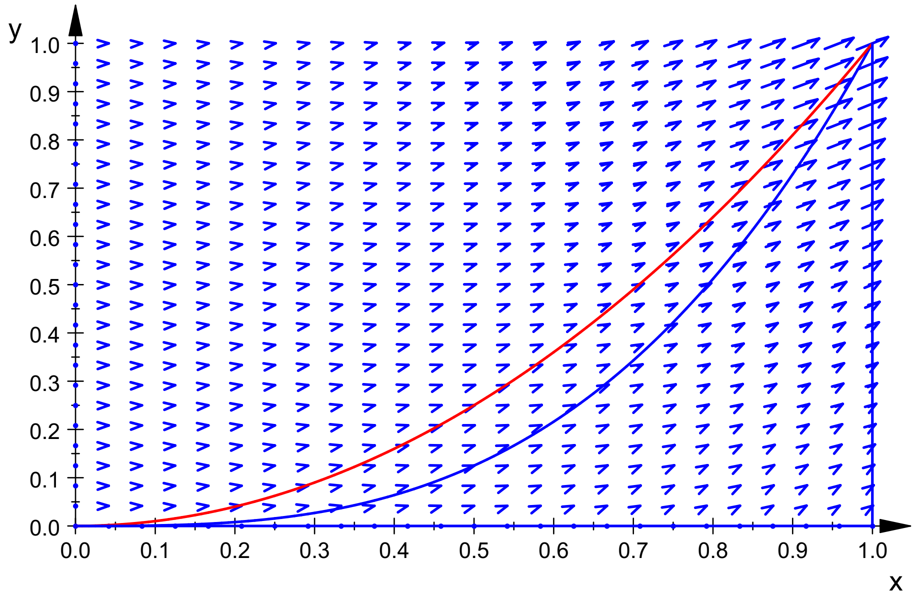
```
plot(plot::VectorField2d([3*x^2*y, 2*x^3*y], x = 0..1, y = 0..1,Mesh =
[25, 25]))
```



题目中不同路径下的第二型线积分分别有

```
y1 :=x^3:
y2 :=x^2:
y3 := plot::Polygon2d([[0,0],[1,0],[1,1]]):
```

```
plot(plot::VectorField2d([3*x^2*y, 2*x^3*y], x = 0..1, y = 0..1,Mesh =
[25, 25]),
     y1,y2,y3,ViewingBox=[0..1,0..1])
```



回忆一下第二型线积分的做法：设 $\sigma:[\alpha,\beta]->R^n$ 是一条参数曲线 L，对于 $[\alpha,\beta]$ 的任意分法 $T:\alpha=t_0<t_1<...<t_m=\beta$

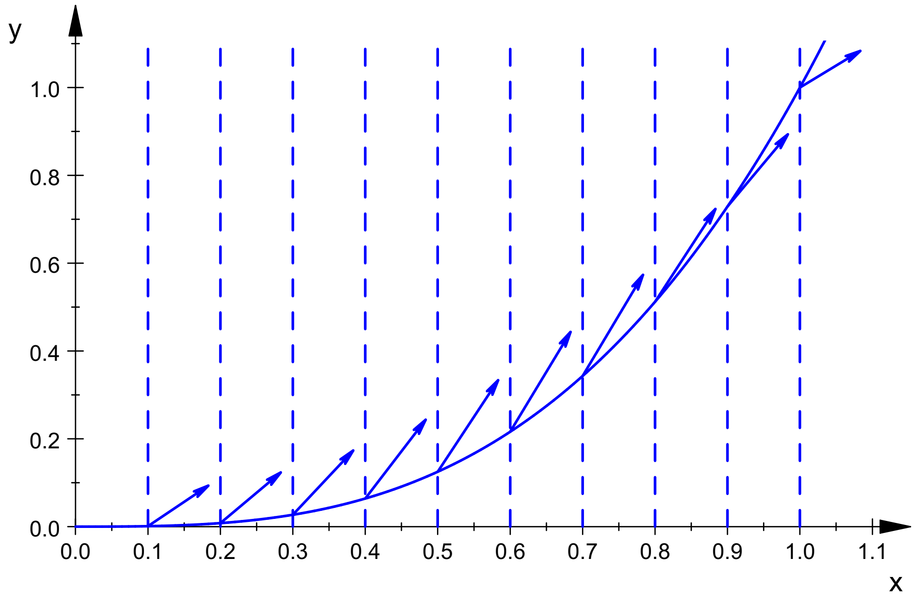
任取 $T\in\left[t_{j-1},t_j\right]$,则 $x_j=\sigma(t_j)$, $\eta_j=\sigma(\tau_j)$ $\in$ L .用常力 $F(\eta_j)$ 在有向线段

$x_{j-1}x_j$ 上做的功，近似变力在有向线段上做的功.

尝试把这一过程直观化。下面展示了这种在划分的每一段中，取常力代替变力的思想。

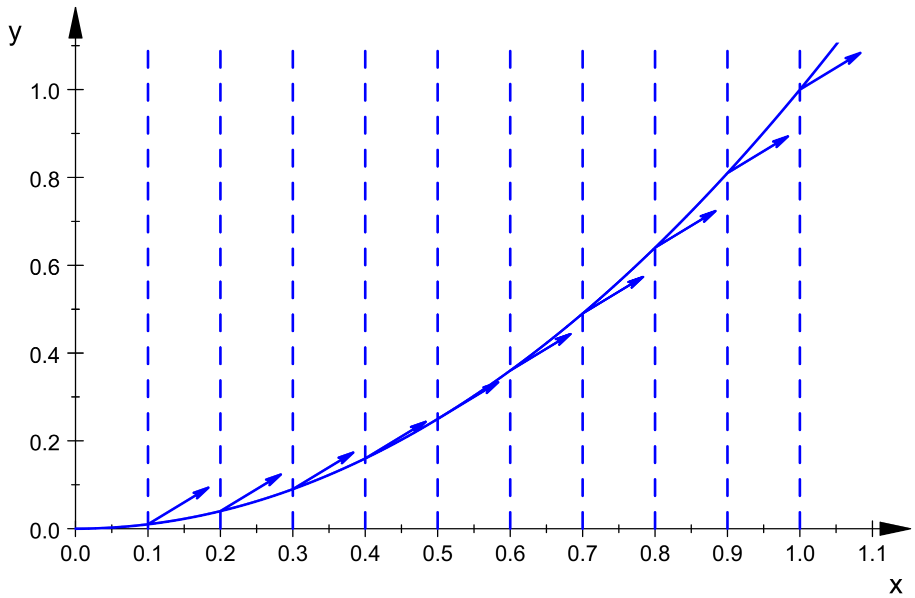
对于 $y=x^3$

```
y1 :=x^3:
for i from 1 to 10 do
   Arrow1[i] := plot::Arrow2d([i/10,(i/10)^3],
[i/10+((3*(i/10)^5)/sqrt((3*(i/10)^5)^2))/12,
                              (i/10)^2+((2*(i/10)^6)/sqrt((2*(i/10)^6)^
2))/12],
                               TipLength = 2*unit::mm);
   Division[i] := plot::Line2d([i/10, 0], [i/10, 1.1],LineStyle =
Dashed)
end_for:
plot(Arrow1[i]$i=1..10,
     Division[i]$i=1..10,
     y1,ViewingBox=[0..1.1,0..1.1])
```



对于 $y = x^2$

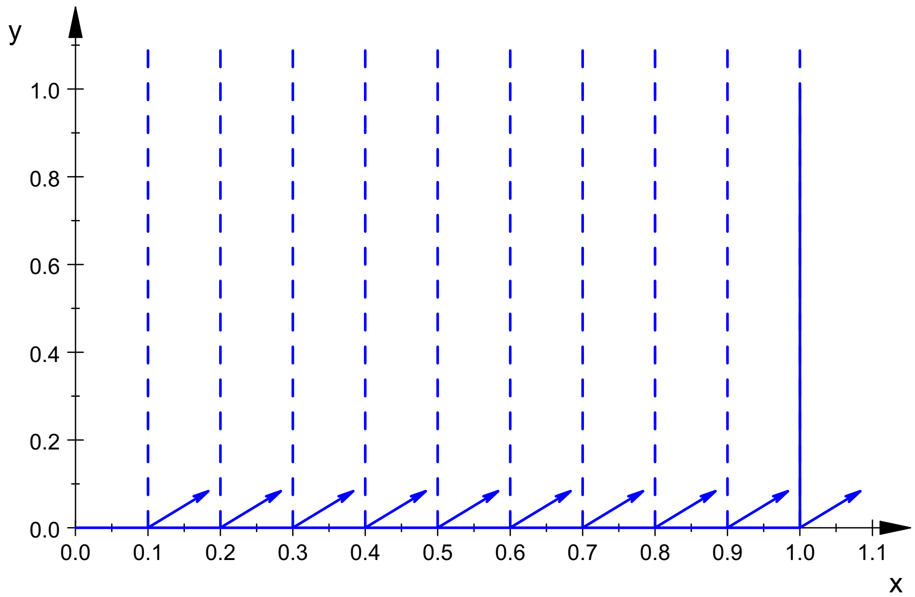
```
y2 :=x^2:
for i from 1 to 10 do
    Arrow2[i] := plot::Arrow2d([i/10,(i/10)^2],
[i/10+((3*(i/10)^4)/sqrt((3*(i/10)^4)^2))/12,
                              (i/10)^2+((2*(i/10)^5)/sqrt((2*(i/10)^5)^
2))/12],
                               TipLength = 2*unit::mm);
    Division[i] := plot::Line2d([i/10, 0], [i/10, 1.1],LineStyle =
Dashed)
end_for:
plot(Arrow2[i]$i=1..10,
     Division[i]$i=1..10,
     y2,ViewingBox=[0..1.1,0..1.1])
```



对于沿折线AC,CB,其中C(0,1)

```
y3 :=plot::Polygon2d([[0,0],[1,0],[1,1]]):
for i from 1 to 10 do
 Arrow3[i] := plot::Arrow2d([i/10,0],
[i/10+((3*(i/10)^4)/sqrt((3*(i/10)^4)^2))/12,
                              ((2*(i/10)^5)/sqrt((2*(i/10)^5)^2))/12],
```

```
                                   TipLength = 2*unit::mm);
Division[i] := plot::Line2d([i/10, 0], [i/10, 1.1],LineStyle = Dashed)
end_for:
plot(Arrow3[i]$i=1..10,
     Division[i]$i=1..10,
     y3,ViewingBox=[0..1.1,0..1.1])
```



我们发现该例中积分与路径无关

```
y1 := x^3:
y2 := x^2:
output::mathText
 (L[1],"=",L[2],"=",L[3],"=",
  int(3*x^2*y1^2+2*x^3*y1*diff(y1,x), x=0..1),"=",
  int(3*x^2*y2^2+2*x^3*y2*diff(y2,x), x=0..1),"=",
  int(2*x,x=0..1))
```

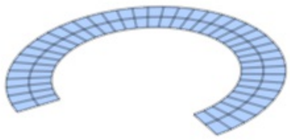
$L_1=L_2=L_3=1=1=1$

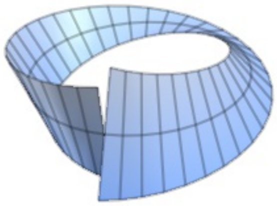
例5. 单侧曲面的例子（莫比乌斯带）

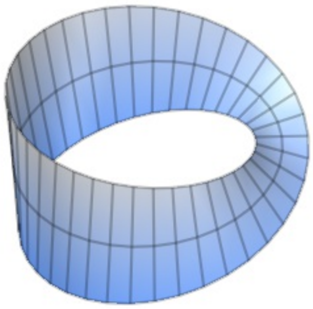
我们给出莫比乌斯带的例子，莫比乌斯带由矩形带子扭转折叠而成
关于折叠过程，给出动画：

```
c := a -> 1/2 *(1 - 1/sin(PI/2*a)):

rectangle2annulus := plot::Surface(
   [c(a) + (u - c(a))*cos(PI*v), (u - c(a))*sin(PI*v), 0],
   u = 0.8..1.2, v = -a..a, a = 1/10^10..1,
   Color = RGB::Grey, Mesh = [3, 40], Frames = 40):

annulus2moebius := plot::Surface(
   [((u - 1)*cos(a*v*PI/2) + 1)*cos(PI*v),
    ((u - 1)*cos(a*v*PI/2) + 1)*sin(PI*v),
     (u - 1)*sin(a*v*PI/2)],
   u = 0.8..1.2, v = -1..1, a = 0..1,
   Color = RGB::Grey, Mesh = [3, 40], Frames = 20):

rectangle2annulus::VisibleFromTo := 0..2:
annulus2moebius::VisibleFromTo := 2..5:
```

```
plot(rectangle2annulus, annulus2moebius, Axes = None,
     CameraDirection = [-11, -3, 3]):
```





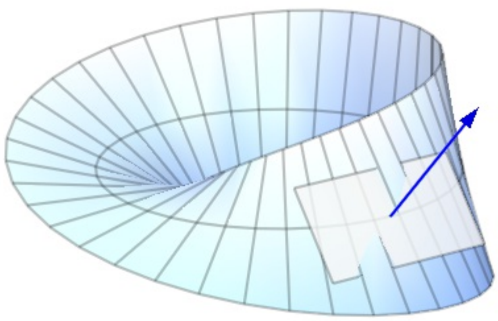


可以通过动画展示莫比乌斯带不可定向的性质，动画中，一个切平面及其法向量，在轴线上一直环绕，周而复始，连续遍历了之前矩形纸袋的每一面。

读者还可以精简下面的代码，其实，无论代码形式的繁简，其基本命令只有那几条，掌握起来十分便捷。

```
moebius := plot::Surface(
    [(2+x*sin(y/2))*cos(y),
     (2+x*sin(y/2))*sin(y),
      x*cos(y/2)],
    x = -1..1, y = 0..2*PI,
    Color=RGB::White, Mesh = [3, 40]):

for i from 1 to 101 do
  t[i] := i/10;
```

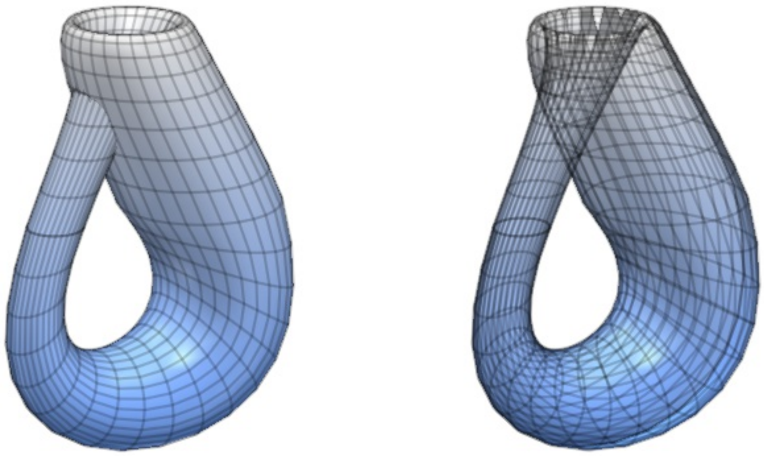
```
end_for:

for i from 1 to 100 do
  v := (i-1)*2*PI/(100);
  myframe[i] := plot::Group3d(
                        plot::Parallelogram3d([2*cos(v), 2*sin(v),0],
[sin(v/2)*cos(v),sin(v/2)*sin(v),cos(v/2)]/2,
                         [-2*sin(v),2*cos(v),0]/2,Color = RGB::Grey.[0.8]),
                        plot::Arrow3d([2*cos(v), 2*sin(v),0],
                         [2*cos(v)-2*cos(v)*cos(v/2),2*sin(v)-2*sin(v)*cos(v
/2), 2*sin(v/2)]),
                        VisibleFromTo = t[i]..t[i + 1]);
end_for:

for i from 101 to 201 do
  t[i] := i/10;
end_for:

for i from 101 to 200 do
  v := (i-101)*2*PI/(100);
  myframe[i] := plot::Group3d(
                        plot::Parallelogram3d([2*cos(v), 2*sin(v),0],
[sin(v/2)*cos(v),sin(v/2)*sin(v),cos(v/2)]/2,
                         [-2*sin(v),2*cos(v),0]/2,Color = RGB::Grey.[0.8]),
                        plot::Arrow3d([2*cos(v), 2*sin(v),0],
                         [2*cos(v)+2*cos(v)*cos(v/2),2*sin(v)+2*sin(v)*cos(v
/2), -2*sin(v/2)]),
                        VisibleFromTo = t[i]..t[i + 1]);
end_for:

plot(moebius,myframe[i] $ i = 1..200,Axes = None):
```



这样有趣的不可定向的例子还有很多，例如“克莱因瓶”，下面给出克莱因瓶的展示：

```
bx := u -> -6*cos(u)*(1 + sin(u)):
by := u -> -14*sin(u):
r :=  u -> 4 - 2*cos(u):
x := (u, v) -> piecewise([u <= PI, bx(u) - r(u)*cos(u)*cos(v)],
                         [PI < u,  bx(u) + r(u)*cos(v)]):
```

```
y := (u, v) -> r(u)*sin(v):
z := (u, v) -> piecewise([u <= PI, by(u) - r(u)*sin(u)*cos(v)],
                         [PI < u,  by(u)]):
KleinBottle:= plot::Surface(
      [x, y, z], u = 0 .. 2*PI, v = 0 .. 2*PI,
      Mesh = [35, 31], LineColor = RGB::Black.[0.2],
      Color = RGB::Grey.[0.8]):
plot(KleinBottle, Axes = None, Scaling = Constrained,
     Width = 60*unit::mm, Height = 72*unit::mm):
```



值得一提的是，Mupad中透明度等图形属性，并不需要通过编辑命令修改，可直接在窗口中就可以修改（后者透明度为0.8）。

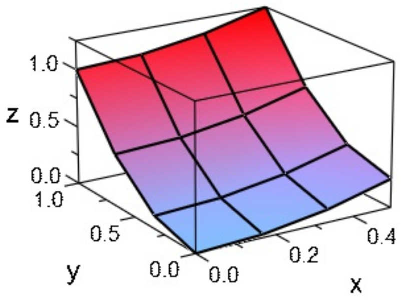
例6.　曲面的划分，以柱面为例，及一个反例

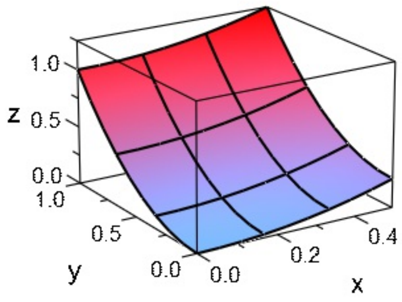
回顾一下正则曲面面积定义：

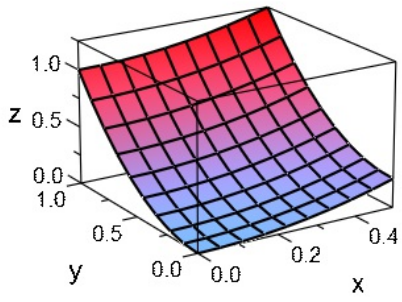
设 $S=\sigma([a,b])\subset R^3$ 是正则曲面，区间 $[a,b]\subset R^2$ 上的一个分法 T，将 $[a,b]$ 分成有限个小区间 $\{\mathrm{i}_{ij}\}$，小区间（矩形）i_{ij} 以 (u_{i-1},v_{j-1}) 为一个顶点，以 Δu_i，Δv_j 为边长，用曲面在点 $p_{ij}=\sigma\left(u_{i-1},v_{j-1}\right)$ 切平面上的平行四边形面积近似小曲面片 $S_{ij}=\sigma\left(\mathrm{i}_{ij}\right)$ 的面积 S_{ij}，这个平行四边形是以 p_{ij} 为一个顶点，由向量 $\sigma'_u\Delta u_i$，$\sigma'_v\Delta v_j$ 张成，因此 $|S_{ij}|\approx|\sigma'_u\Delta u_i\times\sigma'_v\Delta v_j|$

利用Mupad图形的Mesh值，可以轻松的展示这一曲面划分。

```
S1 := plot::Scene3d(plot::Function3d(
        x^2 + y^2, x = 0..1/2, y = 0..1, Mesh = [4, 4])):
S2 := plot::Scene3d(plot::Function3d(
        x^2 + y^2, x = 0..1/2, y = 0..1, Mesh = [4, 4],
       Submesh = [2, 2])):
S3 := plot::Scene3d(plot::Function3d(
        x^2 + y^2, x = 0..1/2, y = 0..1, Mesh = [10, 10])):
plot(S1, S2, S3, Layout = Horizontal,
     Height = 5*unit::cm, Width = 12*unit::cm,
     LineColor = RGB::Black):
```

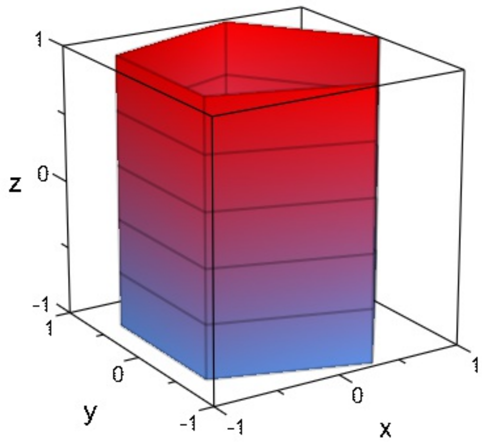


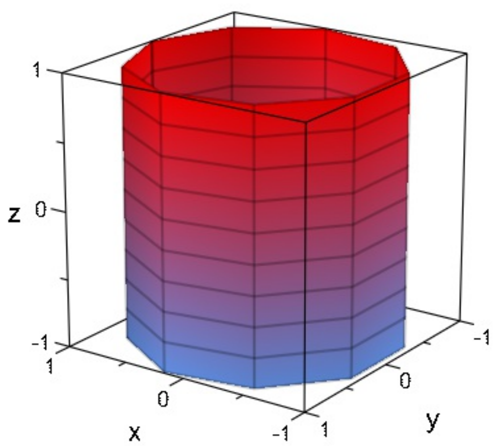


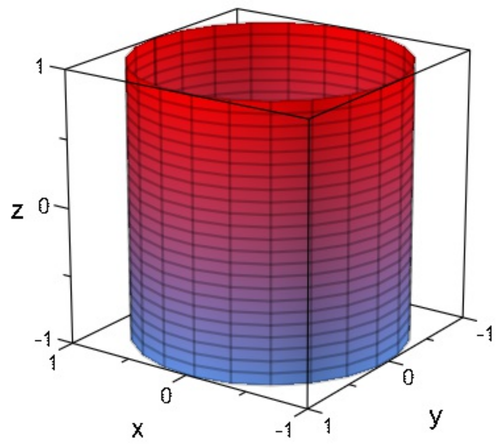


柱面的逼近用这种划分来近似，其过程可以制作成动画：

```
for i from 0 to 21 do
    u[i] := i;
end_for:
for i from 0 to 20 do
  myframe[i] :=  plot::Cylindrical([1, phi, z], phi = 0..2*PI, z =
-1..1,
                  Mesh =[4+i,4+i],VisibleFromTo = u[i]..u[i + 1]);
end_for:


plot(myframe[i] $ i=1..20):
```







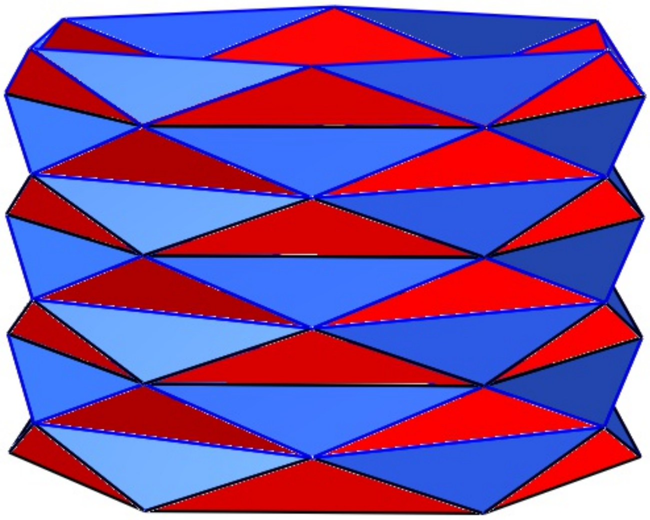
下面展示一个反例：用割面近似曲面

将圆柱面 $S=\{(x,y,z):x^2+y^2=1,z\in[0,\ 1]\}$ 的高m等分，分成m+1个圆周，在每个圆周上取n个等分点，使每个圆周上取n个等分点，是每个圆周上n个等分点为上个圆周n等分点的中点，用这些点的连线构成三角形割面，得到2mn个全等三角形

这个例子的作图有些繁琐，鉴于基本作图命令是简单的，有兴趣的读者可以尝试给出更加简洁的命令（下面仅供参考）。

```
for j from 0 to 2 do
   for i from 1 to 6 do
     S[i,j] := plot::Polygon3d([[cos((i/6)*2*PI), sin((i/6)*2*PI),
(2*j+1)/10] ,
                                [cos(((i+1)/6)*2*PI),
sin(((i+1)/6)*2*PI), (2*j+1)/10],
                                [cos(((0.5+i)/6)*2*PI),
sin(((0.5+i)/6)*2*PI), 2*j/10]],
                                Closed = TRUE,Filled=TRUE,FillColor
=RGB::LightBlue):
   end_for:
end_for:
delete i,j:
////////////////////////////////////////////////////////////////////
for j from 1 to 3 do
   for i from 1 to 6 do
     s[i,j] := plot::Polygon3d([[cos(((0.5+i)/6)*2*PI),
sin(((0.5+i)/6)*2*PI), 2*j/10],
                                [cos(((0.5+i+1)/6)*2*PI),
sin(((0.5+i+1)/6)*2*PI), 2*j/10],
                                [cos(((i+1)/6)*2*PI),
sin(((i+1)/6)*2*PI),(2*j-1)/10]],
                                Closed=TRUE,Color=RGB::Black,Filled=TRU
E,FillColor=RGB::LightBlue):
   end_for:
end_for:
delete i,j:
/////////////////////////////////////////////////////////////////////
for j from 0 to 2 do
   for i from 1 to 6 do
     T[i,j] := plot::Polygon3d([[cos((i/6)*2*PI), sin((i/6)*2*PI),
(2*j+1)/10],
                                [cos(((i+1)/6)*2*PI),
```

```
sin(((i+1)/6)*2*PI),(2*j+1)/10],
                                [cos(((0.5+i)/6)*2*PI),
sin(((0.5+i)/6)*2*PI),(2*(j+1))/10]],
                                Closed = TRUE,Filled=TRUE,FillColor =
RGB::Red):
   end_for:
end_for:
delete i,j:
//////////////////////////////////////////////////////////////////////
for j from 1 to 3 do
   for i from 1 to 6 do
     t[i,j] := plot::Polygon3d([[cos(((0.5+i)/6)*2*PI),
sin(((0.5+i)/6)*2*PI), 2*j/10] ,
                                [cos(((0.5+i+1)/6)*2*PI),
sin(((0.5+i+1)/6)*2*PI), 2*j/10] ,
                                [cos(((i+1)/6)*2*PI),
sin(((i+1)/6)*2*PI),(2*(j+1)-1)/10]],
                                Closed =
TRUE,Color=RGB::Black,Filled=TRUE,FillColor=RGB::Red):
   end_for:
end_for:
delete i,j:
//////////////////////////////////////////////////////////////////////
for i from 1 to 6 do
    B[i] := plot::Polygon3d([[cos(((0.5+i)/6)*2*PI),
sin(((0.5+i)/6)*2*PI), 0] ,
                             [cos(((0.5+i+1)/6)*2*PI),
sin(((0.5+i+1)/6)*2*PI), 0] ,
                             [cos(((i+1)/6)*2*PI),
sin(((i+1)/6)*2*PI),0.1 ]],
                              Closed =
TRUE,Color=RGB::Black,Filled=TRUE,FillColor = RGB::Red):
end_for:
delete i,j:
//////////////////////////////////////////////////////////////////////
for i from 1 to 6 do
     b[i] := plot::Polygon3d([[cos((i/6)*2*PI), sin((i/6)*2*PI),0.7] ,
                              [cos(((i+1)/6)*2*PI),
sin(((i+1)/6)*2*PI), 0.7] ,
                              [cos(((0.5+i)/6)*2*PI),
sin(((0.5+i)/6)*2*PI), 0.6 ]],
                               Closed = TRUE,Filled=TRUE,FillColor =
RGB::LightBlue):
   end_for:
delete i,j:

plot(S[i,j] $ j=0..2 $i=1..6,
     s[i,j] $ j=1..3 $i=1..6,
     T[i,j] $ j=0..2 $i=1..6,
     t[i,j] $ j=1..3 $i=1..6,
     B[i]   $ i=1..6,
     b[i]   $ i=1..6, AxesVisible=FALSE  )
```



Mupad界面，可以旋转放缩，得到更好的观察，这里pdf仅仅是一个初步展示。

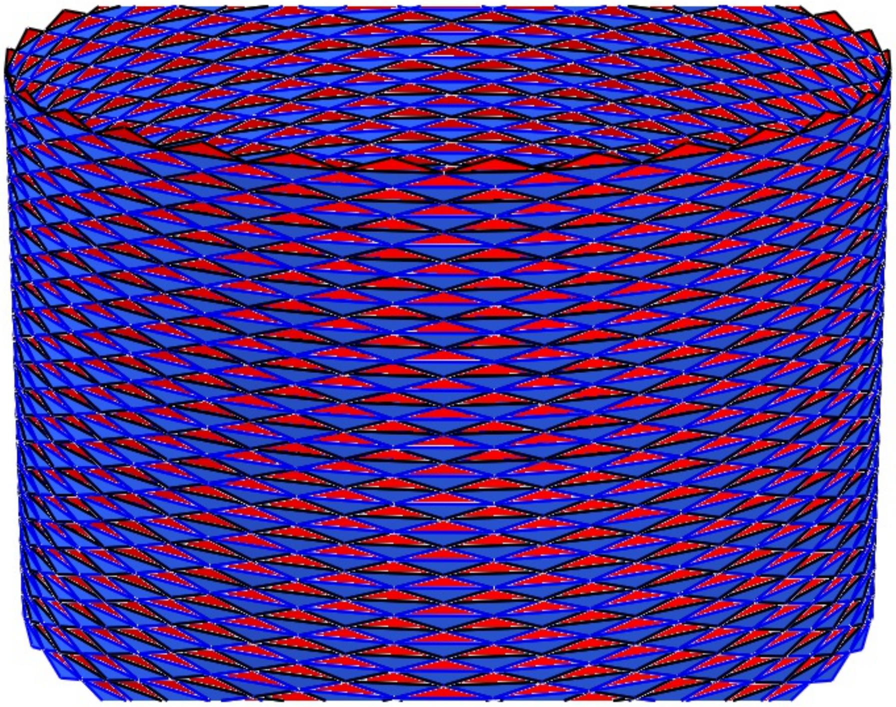
这些割面三角形底边长 $2\sin\left(\frac{\pi}{n}\right)$，高为 $\sqrt{\left(\cos\left(\frac{\pi}{n}\right)-1\right)^2+\frac{1}{m^2}}$，

则割面的总面积 $S=2\,m\,n\sin\left(\frac{\pi}{n}\right)\sqrt{\left(\cos\left(\frac{\pi}{n}\right)-1\right)^2+\frac{1}{m^2}}$，稍加计算就可以知道这个结果发散

```
m :=n^3:
limit(2*m*n*sin(PI/n)*sqrt((1/m)^2+(1-cos(PI/n))^2), n=infinity)
```

∞

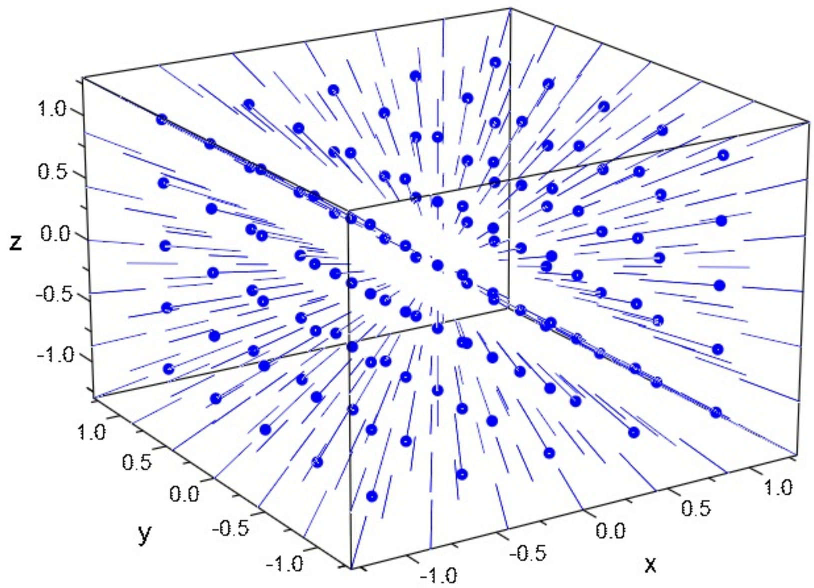
进一步，我们把上面的割面划分加细，就可以看见这种发散背后的直观情形（程序如上，只是更改一些参数）



例7.（流量问题）设 $F=(x, y, z)$，计算 $I=\iint_S F\, n\, \mathrm{ds}$，S是球面 $x^2+y^2+z^2=a^2$ 的外侧

可以利用`plot::VectorField3d`做出流量场，遗憾的是我没有发现这个函数能够改成带箭头的图

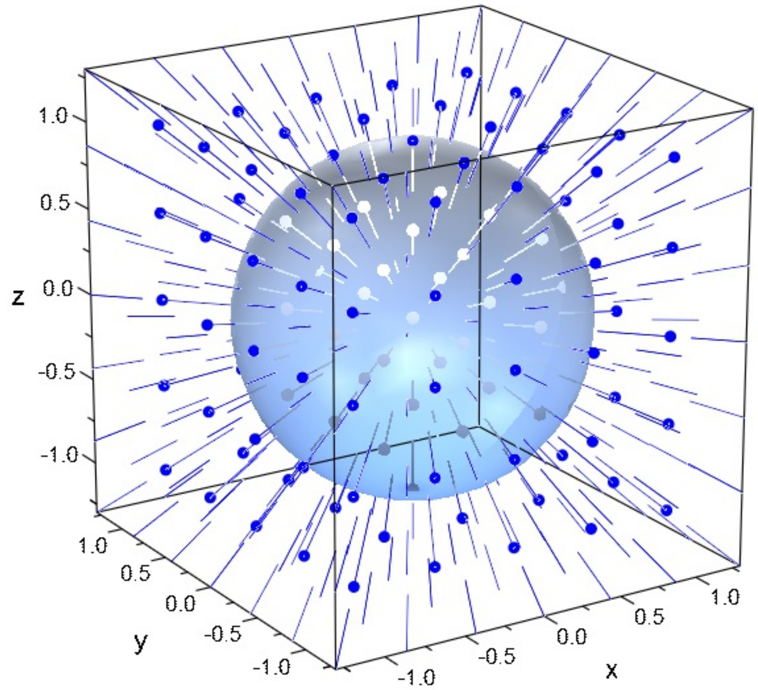
```
plot(plot::VectorField3d([ x,  y,z],x = -1.3..1.3, y = -1.3..1.3, z =
-1.3..1.3))
```



这个流量流过球面的示意图可以表示为：

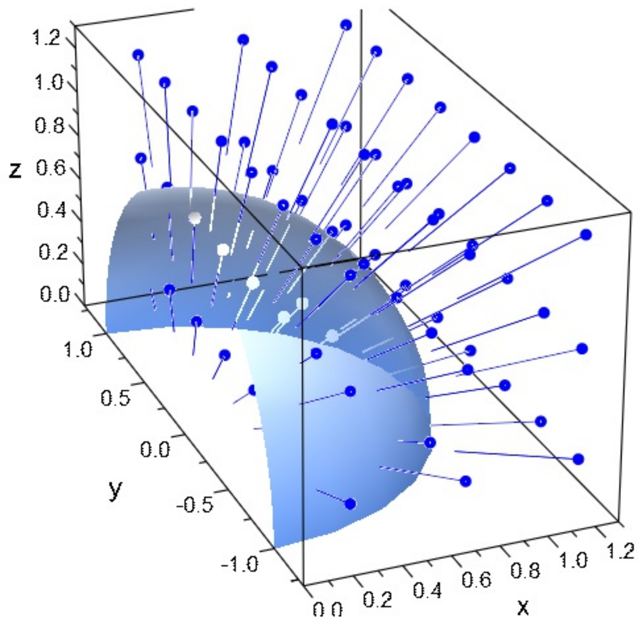
```
plot(plot::VectorField3d([ x,  y,z],x = -1.3..1.3, y = -1.3..1.3, z =
```

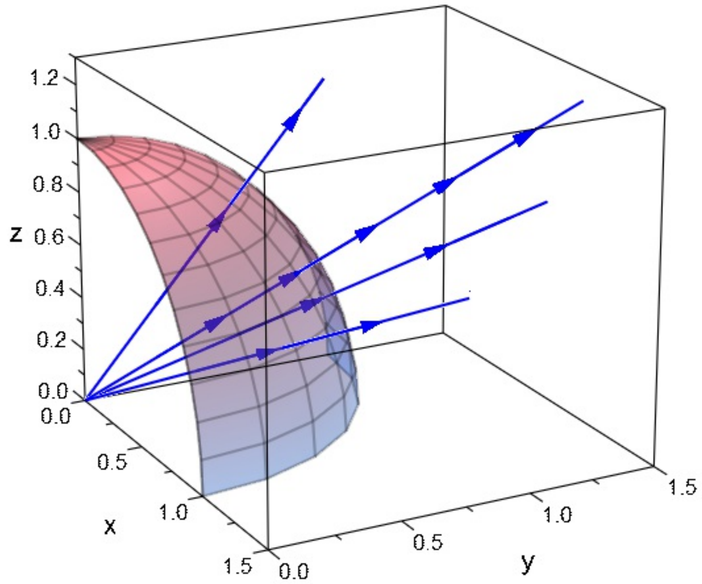
```
-1.3..1.3),
       plot::Spherical([1, u, v], u = 0..2*PI, v = 0..PI,FillColor =
RGB::Black5.[0.1],
       ULinesVisible=FALSE,VLinesVisible=FALSE,Scaling=Constrained))
```



取四分之一球面做一个透视剖析观察

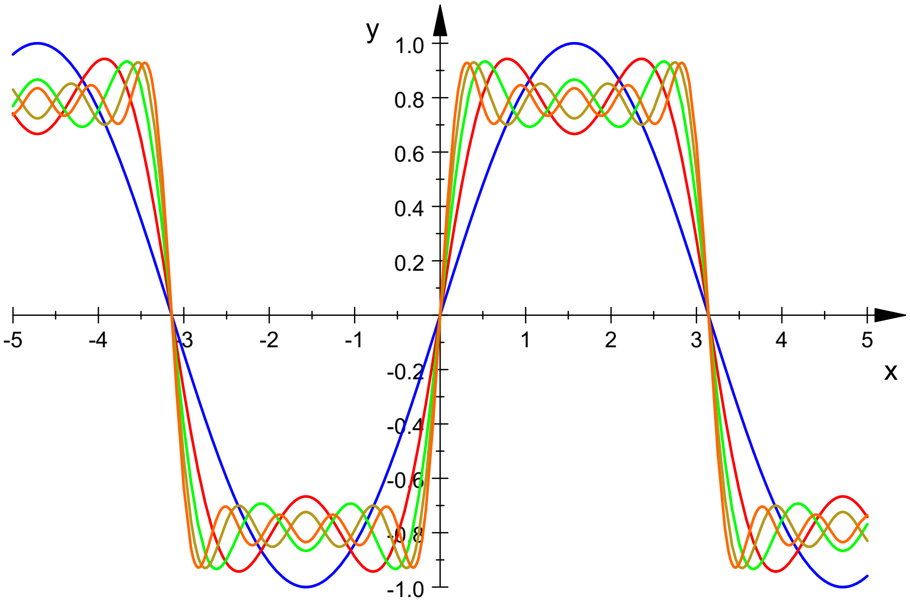
```
plot(plot::VectorField3d([ x, y,z],x = -1..1, y = -1..1, z =
-1..1,Mesh=[6,8,6]),
       plot::Spherical([1, u, v], u = 0..2*PI, v = 0..PI,FillColor =
RGB::Black5.[0.1],
       ULinesVisible=FALSE,VLinesVisible=FALSE,Scaling=Constrained,Viewi
ngBox=[0..1.3,-1.3..1.3,0..1.3]))
```



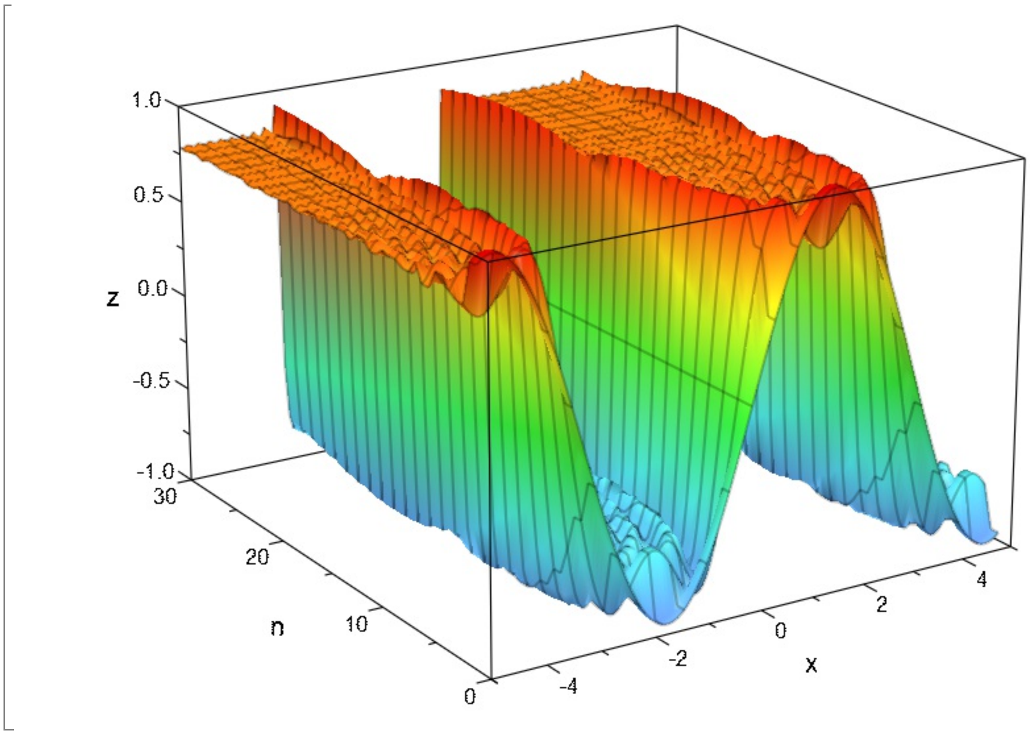


最后给出一个傅里叶逼近的例子，以显示Mupad的便捷，在Mupad只需一行命令就可以展示，甚至是三维视角。

```
f_n := sum(sin((2*k-1)*x)/(2*k-1), k = 1..floor(n)):
plotfunc2d(f_n $ n = 1..5, LegendVisible = FALSE)
```




```
plotfunc3d(f_n, x = -5..5, n = 1..30,
           Submesh = [5,1], FillColorType = Rainbow)
```



值得注意的是以上大多数是动画，PDF版中仅仅给出静图。

以上是基于线面积分的一些Mupad应用展示，由于网上关于Mupad的讨论较少，我也是根据帮助文档一步步探索，总结起来，写在这里，根据我对帮助文档的学习，我认为以上的例子涵盖了利用Mupad图形制作的常用方法，特别是两种基本动画制作方法。

本文本身就是在Mupad note（一个文字和命令交互的界面）中直接生成的PDF，其实这些内容在Mupad note将得到更好的展示。