JAPLA 研究会資料 2016/5/21

Ｊグラフィックスを用いて微分幾何学をのぞいてみる
伸開線(Involute)、縮閉線(Evolute)、
包絡線(Envelope)など

西川　利男

非ユークリッド幾何学なる恐ろしげな数学を耳にすることがあるだろう。単にユークリッドのやり方に従わないだけのもので、円や三角形、四角形だけでなく、そのあたりにあるごくありふれた形までを扱おうという、もっとゆるい一般的な幾何学なのである。その奥まった一分野に多様体(Manifold)幾何学というのもある。これはヘビの体の上を這っているアリの動きの幾何学である。一方、実用的な分野として、測地学(Geodesic)というのもある。

微分幾何学とは、このような分野への入り口となる数学である。数学のやり方の常として、図形を表す座標（＝数値の集まり）を関数の式で表す。そこで曲線や曲面などの細かい変化を記述するためには微分の方法が有効であることで、微分幾何学という名前はここから来ている。

頭の中で図形、とくに立体をイメージすることは仲々大変である。そのための数学の道具がベクトルを用いた線形代数である。したがって、微分幾何学は微積分と線形代数の上に乗っかった幾何学といえる。

さらに、それを目に見える形にする現代の強力な道具が、コンピュータ・グラフィックスであることはいうまでもない。

ここでは、Ｊグラフィックスを用いて、微分幾何学の入り口をのぞいてみよう。そして、微分幾何学で出てくるいろいろな曲線を眺めて楽しむというのが目的である。いわんや、数式の誘導や定理の証明を行うつもりは全くなく、いろいろな本がある。[1-4]

コクセターもその書[3]の序文で言っているように、現代数学は幾何学に興味を失って、哀れな学問になってしまった。これを憂えて、大著「幾何学入門」を著したという。

私に言わせれば、コンピュータ・グラフィックスという道具は、幾何学を現代に生き生きとよみがえらせてくれた。特にわれわれ世代の数学愛好家にとっては、幾何学は整数論とならんで、楽しいテーマとなっている。

[1] 一松信、竹之内修編「新数学事典」Ⅲ.11 微分幾何学 大阪書籍 (1991).
[2] 泉信一ら編「共立数学公式」微分幾何学 共立出版(1966).
[3] コクセター、銀林浩「幾何学入門」下　曲線の微分幾何学 p.159-195、
曲面の微分幾何学 p.221-260、測地線 p.261-283 筑摩書房(2009).
[4] 若山正人編「可視化の技術と現代幾何学」岩波書店(2010).

１．微分幾何学の基本の式

１．１　いろいろな数学座標とコンユータ・グラフィック座標

微分幾何学では、次のようないろいろな座標の取り方が行われる。２次元平面の場合は以下のようである。３次元空間でも同様である。

・直交座標　$x,\ y$
・極座標　　$r,\ \theta$
・パラメータ表示　$x(t),\ y(t)$
・曲線に沿っての経過距離 s を用いた表示　$x(s),\ y(s)$

美的なきれいな曲線は、中心対称なものが多いが、それには極座標がやはり便利である。また、最後のパラメータ s は次に示すように、微分幾何学では重要である。つまり、微分幾何学では、接線、法線、曲率、曲率半径など基本的な量は、パラメータ s を用いて極めてエレガントに数式化される。

１．２　曲線に沿っての経過距離、接線、法線、曲率、曲率半径

微分幾何学では、いろいろな式が出てくるが、極座標 $r,\ \theta$ をから曲線に沿っての経過距離 s を用いたもっとも便利な基本の式だけをあげる。

曲線の長さ　$s=\int_{\theta_0}^{\theta}\sqrt{\left(\frac{dr}{d\theta}\right)^2+r^2}\,d\theta$

$x=f(s),\ y=g(s)$ のとき

接線の方程式　$\dfrac{x-x_0}{\left(\frac{df}{ds}\right)_{x=x_0}}=\dfrac{y-y_0}{\left(\frac{dg}{ds}\right)_{y=y_0}}$

法線の方程式　$\dfrac{x-x_0}{\left(\frac{dg}{ds}\right)_{x=x_0}}=\dfrac{y-y_0}{\left(\frac{df}{ds}\right)_{y=y_0}}$

曲率(κ)と曲率半径($1/\kappa$)

$$\kappa=\frac{df}{ds}\frac{d^2g}{ds}-\frac{dg}{ds}\frac{d^2f}{ds^2}$$

また、これらの距離 s を使った式は、z の同様の式を追加することで、３次元空間内の接線、法線に対しても拡張できる。

微分幾何学の基本定理として、Frenet-Seret の式、自然方程式(natural equation)などがあるが、このあたりの式の展開、導出はややこしく、成書[1],[3],[4]などを見ていただきたい。

ところで、実際のコンピュータ・グラフィックスでは、最終的には直交座標の値で行うことになる。さらに、数学の計算値からコンピュータのピクセル値への変換が必要である。

以下には、微分幾何学で現れるいろいろな図形の式について結果だけを示す。

２．円とその伸開線(Involute)

円

$x = a \cos t$

$y = a \sin t$

伸開線

$X = a(\cos t + t * \sin t)$

$Y = a(\sin t - t * \cos t)$

糸巻きの糸の端を指でつまんで、ほぐしていくときにの曲線が伸開線である。また、これは歯車の歯形曲線として使われる。このとき元となる円は次の縮閉線となる。

３．楕円とその縮閉線(Evolute)

楕円

$x = a \cos t$

$y = b \sin t$

縮閉線

$X = * \cos^3 t$

$Y = - * \sin^3 t$

楕円の周上の各点における法線の集合として現れる包絡線(Envelope)は楕円の縮閉線となる。

４．Ｊによるコンピュータグラフィックスの実際

これらの図形のＪによるグラフィックスは、残念ながらＪのplotルーチンでは、表示できない。plotでは、図形はタテヨコを正規化して表示してしまうので、楕円は円となってしまうのである。

したがってここでは、gl2を直接、使用したウィンドウ・グラフィックスとして行った。プログラムは長くなるが、一定の書式で行うだけで、それほどむずかしいものではない。

円とその伸開線および、楕円とその縮閉線のＪプログラムの基本の計算式はつぎのようになる。

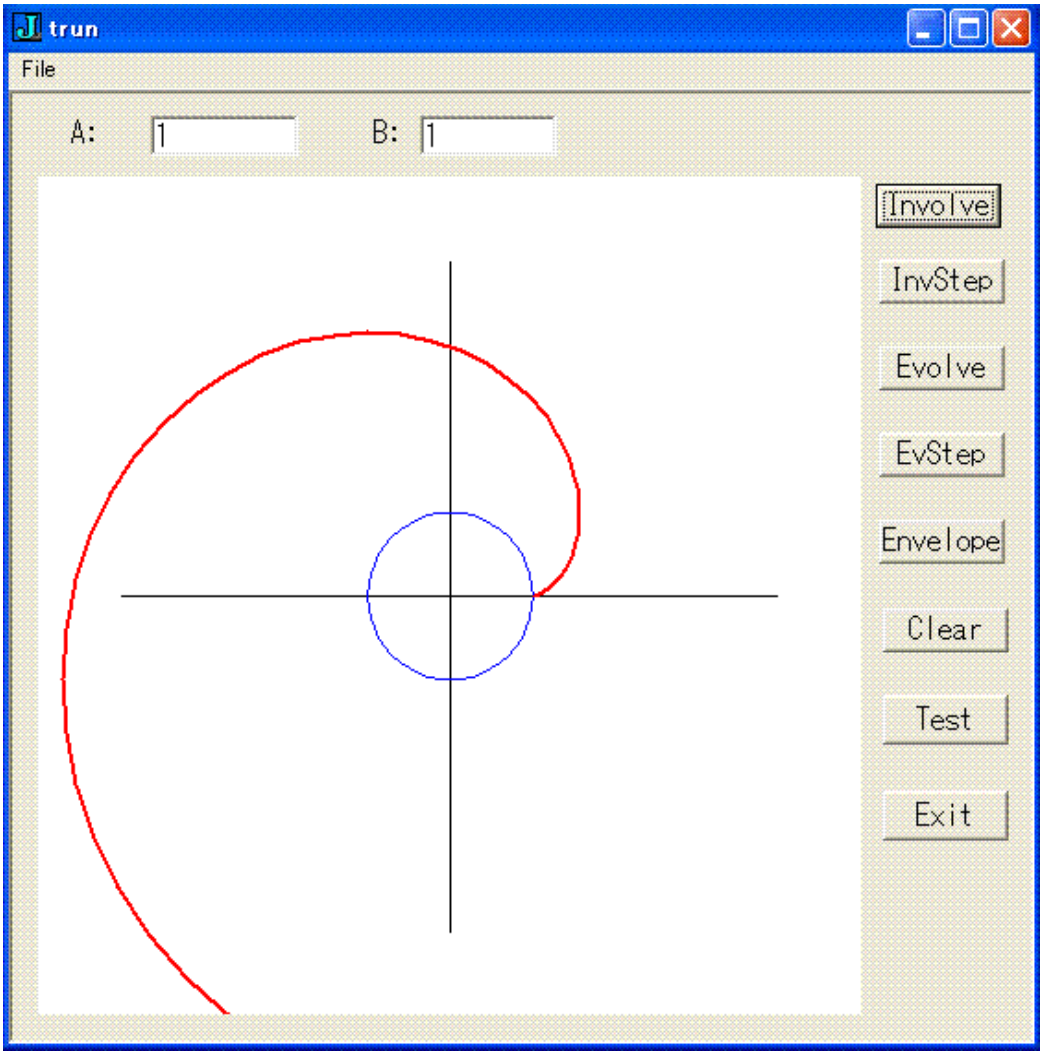
```
t =: i. 49
th =: t * 2r48p1

NB. circle & involute ============================================
NB. A =: 1
NB. P =: (A * cos th);(A * sin th)
NB. Q =: (A *(cos th) + (th * sin th));(A * (sin th) - (th * cos th))

NB. ellipse & evolute ============================================
NB. A =: 1
```

```
NB. B =: 2
NB. AB =: (A^2) - (B^2)
NB. R =: (A * cos th);(B * sin th)
NB. S =: ((AB%A) * (cos th)^3);(-(AB%B) * (sin th)^3)
```

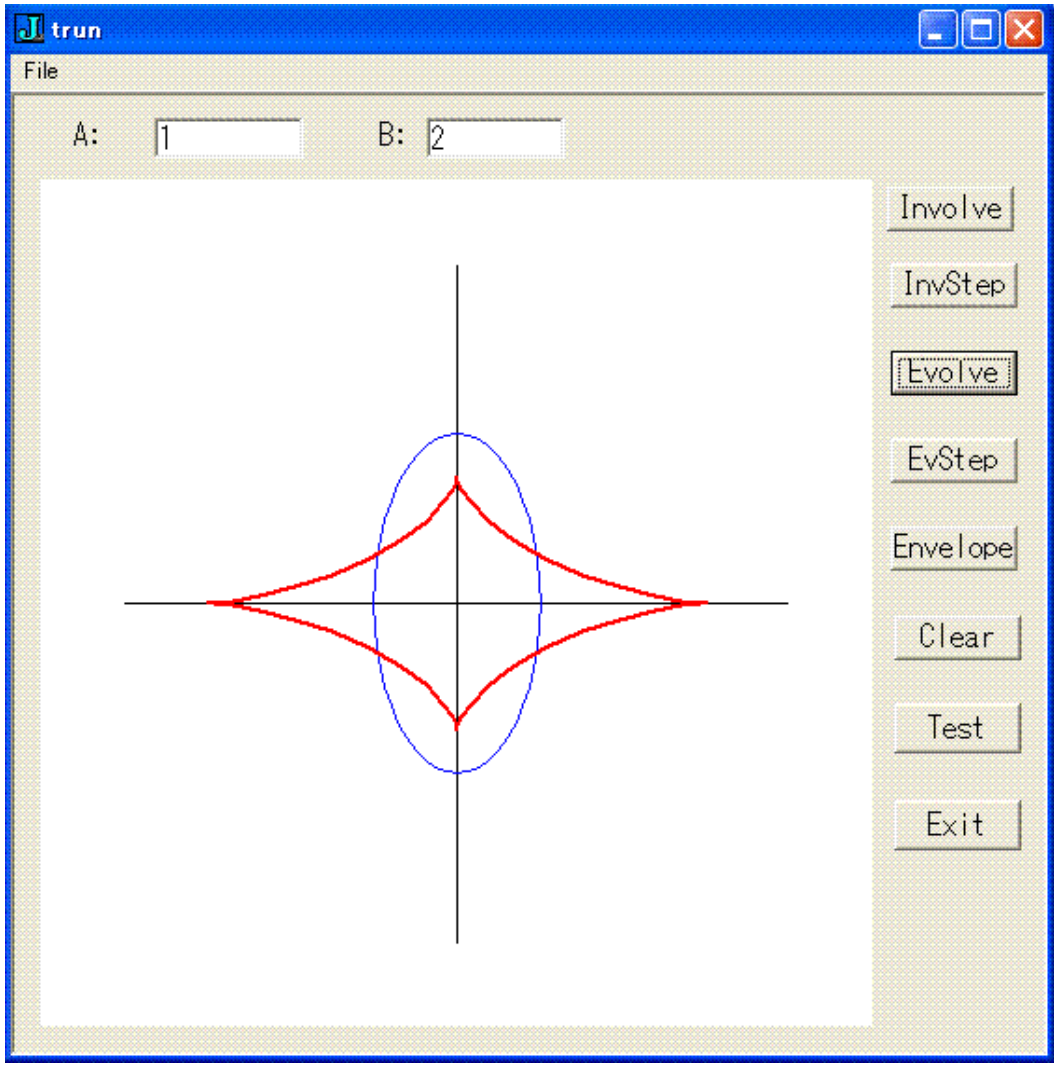
最後に全体のプログラムコーディングをあげる。

実行は、パラメータA, Bを入力し、ボタンの選択により伸開線(Involute)、縮閉線(Evolute)、包絡線(Envelope)を表示する。曲線の一括表示以外に、途中ワンステップずつの表示も選べるようにした。

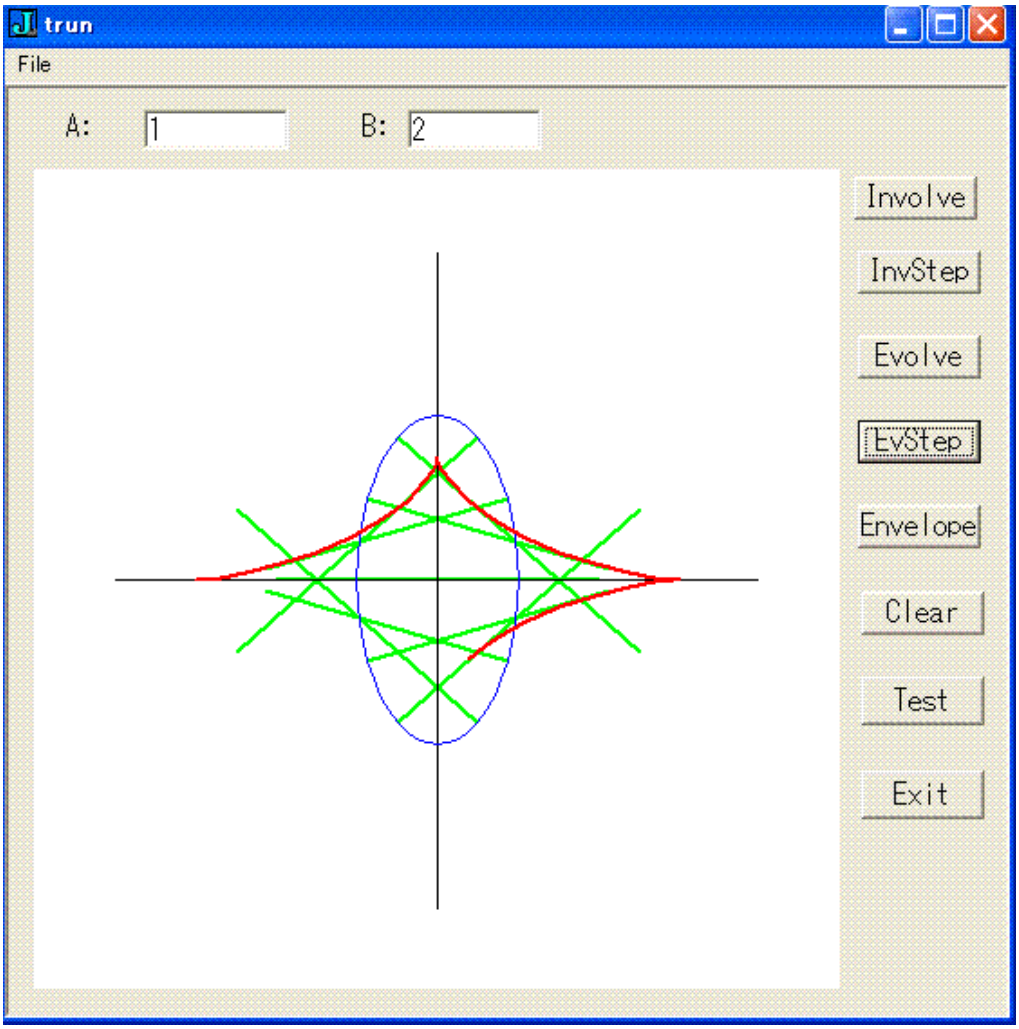
４．１　円とその伸開線(Involute) の実行例



４．２　楕円とその縮閉線(Evolute)の実行例



４．３　楕円とその法線群による包絡線(Envelope)と縮閉線(Evolute)
途中ステップの段階的表示の実行例



５．微分幾何学とは、曲線と曲面の幾何学である

　微分幾何学のほんの入り口だけをのぞいてきたが、その最も基本となる考えは、曲線（曲面については、触れてこられなかったが）、曲率がいろいろ変わる線　＝　ぐにゃぐにゃしたへびのような線、までを対象とした幾何学であり、最も一般的な現実にある図形の幾何学である。微分は、それを行う操作に過ぎず、微分幾何学というより、曲線、曲面幾何学と言ったほうがよい。そして、その仲間として、測地幾何学という実用的にも役に立つ幾何学がある。

　Ｊとそのグラフィックスにより、いずれ挑戦したいと思っている。

NB. involve & evolve curves
NB. from differential geometry

```
require 'trig'
require 'plot'

t =: i. 49
th =: t * 2r48p1

NB. involute ========================================================

NB. A =: 1
NB. P =: (A * cos th);(A * sin th)
NB. Q =: (A *(cos th) + (th * sin th));(A * (sin th) - (th * cos th))

NB. evolute =========================================================

NB. A =: 1
NB. B =: 2
NB. AB =: (A^2) - (B^2)
NB. R =: (A * cos th);(B * sin th)
NB. S =: ((AB%A) * (cos th)^3);(-(AB%B) * (sin th)^3)

NB. display graph on gl2 ============================================
require 'gl2'

TRUN=: 0 : 0
pc trun;
menupop "File";
menu new "&New" "" "" "";
menu open "&Open" "" "" "";
menusep ;
menu exit "&Exit" "" "" "";
menupopz;
xywh 236 145 34 12;cc ok button;cn "Test";
xywh 236 168 34 12;cc cancel button;cn "Exit";
xywh 7 20 223 202;cc disp isigraph;
xywh 234 22 34 11;cc Involve button;
xywh 235 61 34 11;cc Evolve button;
xywh 236 124 34 11;cc Clear button;
xywh 235 40 34 11;cc InvStep button;
xywh 235 82 34 11;cc EvStep button;
xywh 235 103 34 11;cc Envelope button;
xywh 16 6 16 10;cc label static;cn "A:";
```

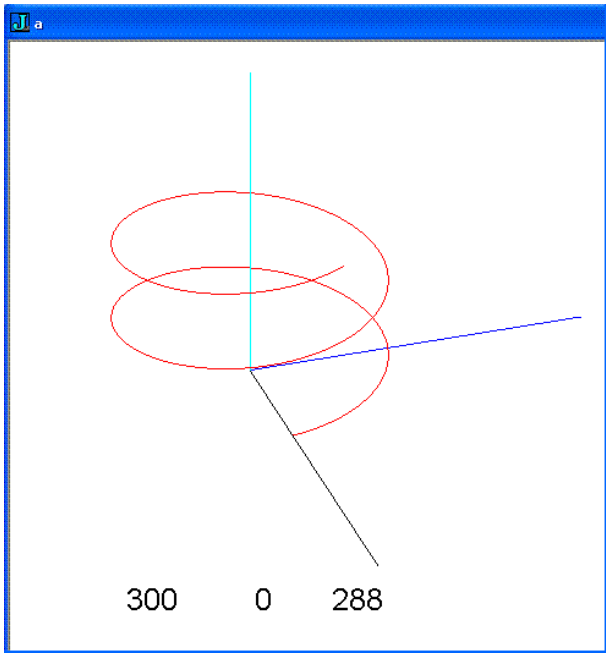
```
xywh 37 5 41 11;cc edA edit ws_border es_autohscroll;
xywh 98 6 17 10;cc label static;cn "B:";
xywh 110 5 38 11;cc edB edit ws_border es_autohscroll;
pas 6 6;pcenter;
rem form end;
)


run =: trun_run
trun_run=: 3 : 0
wd TRUN
NB. initialize form here
ISTEP =: 0
JSTEP =: 0
wd 'set edA', ' "2" ', ';'
wd 'set edB', ' "1" ', ';'
A =: 2
B =: 1
AB =: (A^2) - (B^2)
wd 'pshow;'
)

trun_close=: 3 : 0
wd'pclose'
)

trun_cancel_button=: 3 : 0
trun_close''
)

adj =: 3 : 0
500 + , 100 * |: > y.
)

wr =: 1!:2&2
trun_Involve_button=: 3 : 0
gllines 100 500 900 500
gllines 500 100 500 900
glrgb 0 0 255
glbrush ''
glpen 1, 0
gllines 500 + , 100 * |: > P
```

```
glrgb 255 0 0
glbrush ''
glpen 4, 0
gllines 500 + , 100 * |: > Q
glshow ''
)

trun_Evolve_button=: 3 : 0
AB =: (A^2) - (B^2)
gllines 100 500 900 500
gllines 500 100 500 900
glrgb 0 0 255
glbrush ''
glpen 1, 0
gllines 500 + , 100 * |: > R
glrgb 255 0 0
glbrush ''
glpen 4, 0
gllines 500 + , 100 * |: > S
glshow ''
)

trun_Clear_button=: 3 : 0
ISTEP =: 0
JSTEP =: 0
glclear ''
)

trun_InvStep_button=: 3 : 0
ISTEP =: ISTEP + 4
tt =. i. ISTEP
stth =: tt * 2r48p1
NB. A =: 1
PP =. (A * cos th);(B * sin th)
QQ =. (A *(cos stth) + (stth * sin stth));(B * (sin stth) - (stth *
cos stth))

glrgb 0 0 0
glbrush ''
glpen 1, 0
gllines 100 500 900 500
gllines 500 100 500 900
```

```
glrgb 0 0 255
glbrush ''
glpen 1, 0
gllines 500 + , 100 * |: > PP

glrgb 255 0 0
glbrush ''
glpen 4, 0
gllines 500 + , 100 * |: > QQ
glshow ''
)


trun_EvStep_button=: 3 : 0
AB =: (A^2) - (B^2)

JSTEP =: JSTEP + 1
tt =. i. JSTEP
ssth =. tt * 2r48p1

glrgb 0 0 0
glbrush ''
glpen 1, 0
gllines 100 500 900 500
gllines 500 100 500 900

glrgb 0 0 255
glbrush ''
glpen 1, 0
gllines 500 + , 100 * |: > R
SS =. ((AB%A) * (cos ssth)^3);(-(AB%B) * (sin ssth)^3)
glrgb 255 0 0
glbrush ''
glpen 4, 0
gllines 500 + , 100 * |: > SS
glshow ''
)

trun_Envelope_button=: 3 : 0
glrgb 0 255 0
glbrush ''
```

```
glpen 4, 0
ith =. 0
while. ith < 360
  do.
  if. (ith = 90) +. (ith = 270) do. goto_skip. end.
  EXY =. (A * cosd ith), (B * sind ith)
  EXY1 =. (_3, _3 * ((A*sind ith)%(B*cosd ith)) ) +"(0) EXY
  if. (ith > 90) *. (ith < 270)
  do.
    EXY1 =. (3, 3 * ((A*sind ith)%(B*cosd ith)) ) +"(0) EXY
  end.
  gllines (500 + 100 * EXY), (500 + 100 * EXY1)
  label_skip.
  ith =. ith + 30
end.
glshow ''
)

trun_edA_button=: 3 : 0
A =: ". edA
P =: (A * cos th);(A * sin th)
Q =: (A *(cos th) + (th * sin th));(A * (sin th) - (th * cos th))
)


trun_edB_button=: 3 : 0
B =: ". edB
P =: (A * cos th);(A * sin th)
Q =: (A *(cos th) + (th * sin th));(A * (sin th) - (th * cos th))
AB =: (A^2) - (B^2)
R =: (A * cos th);(B * sin th)
S =: ((AB%A) * (cos th)^3);(-(AB%B) * (sin th)^3)
)
```

```
load'f:¥j402¥user¥eigen_3dview.ijs'
run 13
```




```
helix =: 3 : 0
glLineWidth 1
glBegin GL_LINES
  glColor 0 0 0 1  NB. X-axis
  glVertex L:0 (0 0 0);(6 0 0)
  glColor 0 0 1 1  NB. Y-axis
  glVertex L:0 (0 0 0);(0 6 0)
  glColor 0 1 1 1  NB. Z-axis
  glVertex L:0 (0 0 0);(0 0 7)
glEnd ''
glBegin GL_LINE_STRIP
  glColor 1 0 0 1
  i =. 0
  while. i < 150
    do.
      t =. (1r36p1) * i
      X =. 2 * cos t
      Y =. 2 * sin t
      Z =. 0.2 * t
      glVertex X, Y, Z
      i =. i + 1
    end.
glEnd ''
)
```