Bending for the logarithmic spiral

■ Introduction

The logarithmic spiral is a curve in the complex plane of the form $exp\ (2\pi\ t\ (i + c))$. Here t is the variable which changes from $-\infty$to $+\infty$ as we move along the spiral, and c is a real constant. Since all our considerations will be preserved by isometries of hyperbolic 3−space,and these include $z\to 1/z$ and $z\to\bar{z}$, we may as well assume that $c>0$. When analyzing the hyperbolic convex hull of the logarithmic spiral, and the complement of the logarithmic spiral in the complex plane, the important thing to realize is that there is a 1−parametergroup preserving the subset.
In fact, the group of orientation preserving isometries of hyperbolic 3−space preserving the logarithmic spiral is isomorphic to the group of isometries of the real line. An orientation−preservingisometry of hyperbolic 3−space preserving the logarithmic spiral has either
1) the form $z\to exp\ (2\pi\ \tau\ (i + c))\ z$, where τ is a real constant, or
2) the form $z\to \dfrac{exp\,(2\,\pi\,\tau\,(i+c))}{z}$.
In terms of isometries of the real line, these correspond to $t\to t+\tau$ and to $t\to\tau-t$.

The second of these will be of most use to us. Note that it is an elliptic involution, with fixed points at $z\to \pm exp(\pi\ \tau\ (i + c))\ z$. Let $\alpha(\tau,c)$ be the geodesic in upper half 3−space joining these two points.

The original group acts in a loxodromic manner. There are two hyperbolic planes which are relevant. One is the complement Ω of the logarithmic spiral, endowed with the hyperbolic metric coming from the Riemann mapping. The other is the convex hull boundary of the logarithmic spiral. In each of these the group acts with a fixed geodesic as the axis. These geodesics travel from one end of the spiral to the other, either inside Ω, or on the convex hull boundary. The two geodesics correspond under the nearest point retraction from the floor Ω to the convex hull boundary. The convex hull boundary has the shape of half a snail shell. We denote by γ the geodesic on the snail shell.

From the fact that the situation is invariant under a group, we see that the bending lines all make the same fixed angle $\theta(c)$ with γ. Then $\theta(c)$ varies from $\pi/2$ to 0 as c varies from 0 to ∞. It would be easy to plot the curve of $\theta(c)$ against c but I haven't done so. The bending measure for the logarithmic spiral induces a multiple of Lebesgue measure on γ. Any smaller measure with the same foliation by geodesics at angle $\theta(c)$ gives rise to a spiral region whose complement is a second spiral region. Any larger measure will give a pleated surface which is not embedded.

The orbit of any point in the domain Ω is a spiral. In the hyperbolic plane these become lines of constant curvature, at a fixed distance from the geodesic γ.

Finding the bending measure (some constant multiple of the Lebesgue measure) along γ will enable us to compute any type of measurement of bending which interests us. In order to find γ explicitly, we proceed as follows. The key observation is that if an elliptic involution preserves the spiral, then its set of fixed points forms a geodesic α which meets γ on the snail shell. One endpoint of α lies in Ω on the spiral which is a geodesic in the hyperbolic metric on Ω. The other end can't lie in Ω because an elliptic has only one fixed point in the hyperbolic plane, and so the other end must lie on the main logarithmic spiral.

The general elliptic involution with these properties is $z\to e^{2\pi k(i+c)}/z$. From the maximal circle which is tangent at 1 to the logarithmic spiral, we obtain a single bending line, with endpoints at $t=0$ and at $t=time[c]$. The involution which interchanges these two endpoints has $k=time[c]$. This involution interchanges the two endpoints of the bending line, 1 and $e^{2\pi k(i+c)}$. Its fixed points are $\pm e^{\pi k(i+c)}$. The geodesic in upper half 3−spaceconnecting these two fixed points is the set of fixed points of the elliptic involution in 3−space.

We want to study the bending of the convex hull boundary. This means we need to compute the maximal circles in the complement of the logarithmic spiral.

■ Compute the maximal circles

Because of the group action, we need only find a single maximal circle in the complement of the logarithmic spiral. We will find the circle passing through the point 1. We need to compute the other tangent point.

```
Off[Remove::rmnsm](*no error message
  if there is nothing to remove*)
Remove["`*"];(*remove the values of all existing variables*)
On[Remove::rmnsm];
```

The spiral is defined by the formula

```
sp[t_] := Exp[2 Pi (c + I) t];
```

We will find the maximal circle by looking for the equation of the centre, as determined by the unit normal at two different points on the spiral. The unit normal is

```
n[t_] := -I sp'[t]/Abs[sp'[t]];
```

The following clever way of finding our circle was shown to me by Allan Hayes.
If r is positive, the following equation says that we have a circle of radius r which is tangent at the two points on the spiral corresponding to $t = 0$ and to some general value of t.

```
equ=sp[0]+r*n[0] == sp[t]-r*n[t];
rsol=Solve[equ, r]
    (*solve for r in terms of the other variables*)
```

$$\left\{\left\{r \to \frac{-1+e^{2\,(i+c)\,\pi\,t}}{-\frac{i\,(i+c)}{\text{Abs}[i+c]}-\frac{i\,(i+c)\,e^{2\,(i+c)\,\pi\,t-2\,\pi\,\text{Re}[(i+c)\,t]}}{\text{Abs}[i+c]}}\right\}\right\}$$

The general solution found by *Mathematica* is a complex number. We want to find t such that r is real. So we direct our attention to the imaginary part of the above solution.
ComplexExpand expands an expression assuming that all symbols stand for real variables, converting to use of the real and imaginary part functions wherever possible. After doing this, we simplify the results.

```
num=ComplexExpand[Im[r]/.rsol[[1]],
    TargetFunctions->{Re,Im}]//Simplify
```

$$\frac{c\,(-1+e^{2\,c\,\pi\,t})+(1+e^{2\,c\,\pi\,t})\,\text{Tan}[\pi\,t]}{2\,\sqrt{1+c^2}}$$

We want this to be zero, so we only need to look at the numerator. % means "the result of the preceding computation". We convert to trigonometric functions and then multiply by various non−zero functions, until we get a nice simple condition.

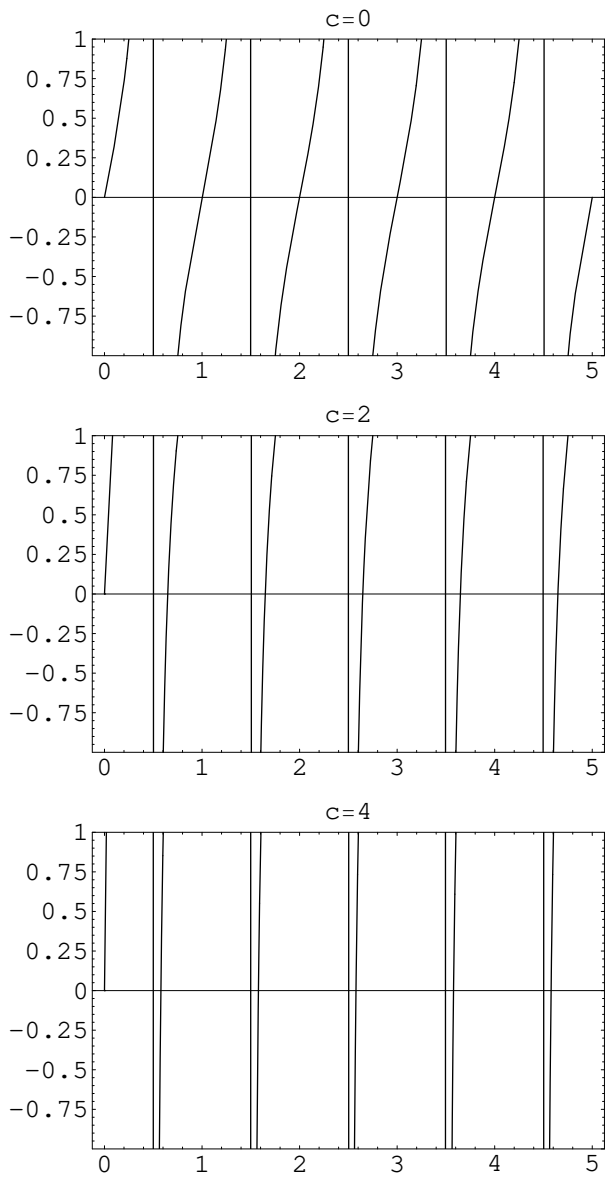
```
num1=ExpToTrig[Numerator[num]];
num2=Simplify[num1,c>0 && t>0];
(*simplify the result of the preceding computation under the assumption that c>0
and t>0*)
num3 = Simplify[num2/((2 Sec[π*t])(Cosh[c*π*t] + Sinh[c*π*t]))];
Expand[num3/(Cosh[c π t]*Cos[π*t])]
Clear["num*"];
```

$$\text{Tan}[\pi\,t]+c\,\text{Tanh}[c\,\pi\,t]$$

```
tst[t_, c_] := Tan[π t] + c Tanh[c π t]
```

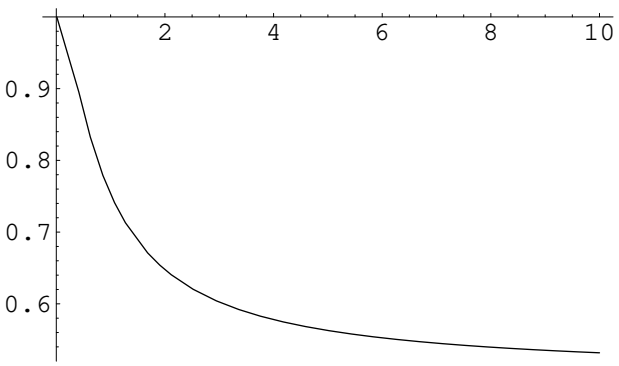
This is a nice simple formula whose smallest positive solution for *t* gives the second point of contact for the maximal circle tangent at 1.
For c>0, in each t–interval [(n–1/2),(n+1/2)], n an integer, it increases from $-\infty$to to ∞; so there is one zero in each interval.
As c$\longrightarrow \infty$ the zeros move from the middle to the left of the intervals.

```
Do[str="c="<>ToString[c];
    Plot[tst[t,c],{t,0,5},
        PlotRange->{-1,1}, Frame->True,PlotLabel->str],
    {c,0,4,2}
]
```



We are interested in the zero which lies in the interval containing 1. FindRoot uses Newton's method. It's not too elegant to start at .501 and it doesn't work for large values of *c*. A more robust technique would be to use bisection. But this would take longer to run. Alternatively, the starting point for Newton's method should be adjusted, depending on *c*. We need to ensure that the starting point is to the right of 0.5 and sufficiently close to 0.5. Using Newton's method as below works up to *c*=10000 and soon after that stops working. Another point about *tst*[*t*,*c*] is that it gets very large when *c* gets large. To obtain meaningful results for use later on, we need to compute the zero with more than normal machine precision for *c* greater than around 7.

```
Clear[time];
time[c_] := t /. FindRoot[tst[t, c], {t, .501}];
Plot[time[c], {c, .01, 10}];
```



The graph above shows the time at which the maximal circle next touches, after t=0. Now we do a careful computation of where the maximal circle tangent at t=0 again touches the spiral when c=1.

```
carefulTime[c_] := t /. FindRoot[tst[t, c], {t, .501},
    WorkingPrecision → 70];
carefulTime[1]
```

```
0.75280936557096988453738713641017070961588000847922264856434345681444 57
```

We next compute the real part of r, the radius of the circle we are looking for.

```
Clear[a, c, d, t]; a =
 Simplify[ComplexExpand[Re[r /. rsol[[1]]], TargetFunctions → {Re, Im}], {t > 0, c > 0}]
```

$$\frac{-1 + e^{2 c \pi t} - c \left(1 + e^{2 c \pi t}\right) \text{Tan}[\pi t]}{2 \sqrt{1 + c^2}}$$

We substitute for Tan[π t], using *tst*[t] == 0.

```
Simplify[a /. Tan[π t] → -c Tanh[c π t]]
```

$$\frac{1}{2} \sqrt{1 + c^2} \left(-1 + e^{2 c \pi t}\right)$$

nv[c] is the normal vector at t=0, with the dependence on c explicit. n[0] was the same thing conceptually, but it was expressed as a complex number, and the dependence on c was implicit. The idea will be to take *k=time[c]*, the time of the first tangency to the maximal circle after t=0, but we don't do this for now. *r[c,k]* is the radius of the maximal circle with the dependence on c and *k=time[c]* made explicit.

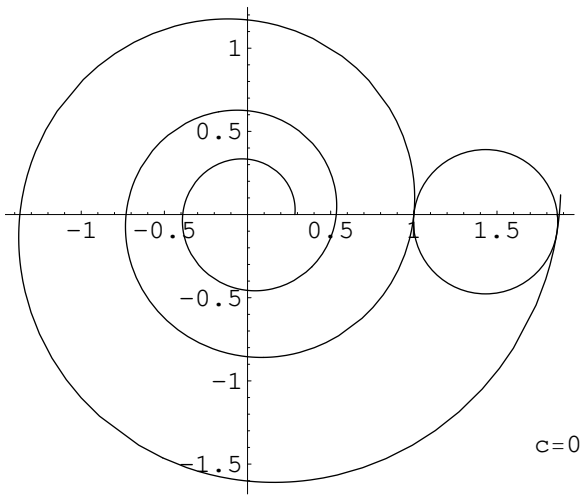
```
Clear[r, c, k, nv];
r[c_, k_] := 1/2 Sqrt[1 + c^2] (-1 + Exp[2 c * π * k]);
nv[c_] = ComplexExpand[{Re[n[0]], Im[n[0]]}, TargetFunctions → {Re, Im}];
```

The next function gives the centre of the maximal circle, provided we put *k=time[c]*.

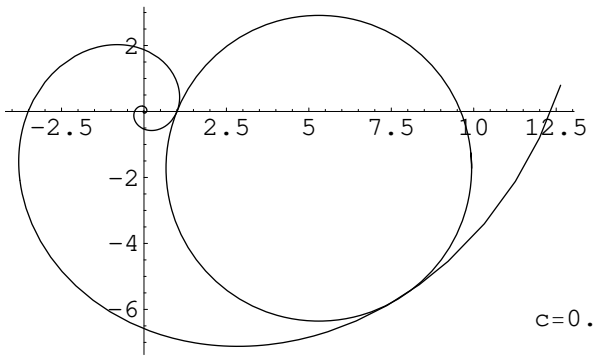
```
Clear[centre] ; (*remove all previous values*)
centre[c_, k_] := {1, 0} + r[c, k] * nv[c] ;
```

Now let's draw some pictures and make sure the computation is correct.

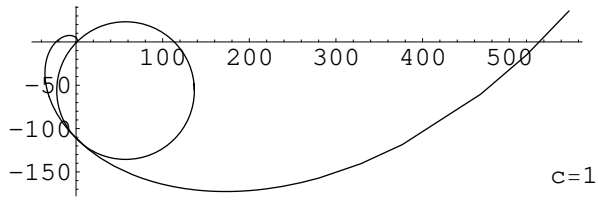
```
Do[str = StringJoin["c=", ToString[c]]; (*make the title*)
 k = time[c];
 spiral = ParametricPlot[{Re[sp[t]], Im[sp[t]]}, {t, -2, 1.01},
   DisplayFunction → Identity];
 circ2 = Graphics[Circle[centre[c, k], r[c, k]]];
 (*the maximal circle*)
 txt = Graphics[Text[str, Scaled[{1, .1}]]];
 Show[{spiral, circ2, txt}, {
   DisplayFunction → $DisplayFunction,
   PlotRange → All,
   AspectRatio → Automatic
  }
 ], {c, .1, 1, .3}]
```



c=0.1



c=0.4



c=1.

■ Find $\alpha \bigcap \gamma$, the unique fixed point of a certain elliptic involution on the hyperbolic axis of the 1–parametergroup, acting on the snail shell dome.

Look for the point on the convex hull boundary where the unique geodesic γ on the convex hull boundary preserved by the 1–parametergroup meets the geodesic α of fixed points of the elliptic involution interchanging the two points of contact of the maximal circle. The two points of contact are {1,0} and *vec*[*c,2k*]. The two fixed points of the involution in the plane are ±*vec*[*c,k*]. The last line of the next computation is the expression which is zero at the intersection of the two intervals connecting these pairs of points. Each such interval is the image of the vertical projection of one of the two geodesics onto the complex plane.

```
Clear[c, k, term, vec, equ2, s, t]; term[c_, k_] = ComplexExpand[Exp[Pi * k * (I + c)],
  TargetFunctions → {Re, Im}]; (*the second tangent point of the maximal circle,
 expressed as a complex number*)
vec[c_, k_] = {term[c, k][[1]], term[c, k][[2]] / I};
(*the second tangent point, expressed as a vector*)
equ2 = (1 - t) {1, 0} + t * vec[c, 2 k] - s * vec[c, k]
```

$$\{1 - t - e^{c k \pi}\, s \operatorname{Cos}[k \pi] + e^{2 c k \pi}\, t \operatorname{Cos}[2 k \pi],\ -e^{c k \pi}\, s \operatorname{Sin}[k \pi] + e^{2 c k \pi}\, t \operatorname{Sin}[2 k \pi]\}$$

The above expression has to equal 0 at the intersection of the two intervals. We now compute the *(x,y)*–coordinates of the intersection $\alpha \bigcap \gamma$ in terms of c and k.

```
Clear[c, k, t, s, xycoords, fact, tssol];
tssol = Solve[{equ2[[1]] == 0, equ2[[2]] == 0}, {t, s}] // Simplify;
s /. tssol
fact = ( s * Cos[k * π] * e^(c k π)) /. tssol [[1]]
vec[c, k]
xycoords[c_, k_] = fact * {1, Tan[k * π]}
```

$$\left\{\frac{2\, e^{c k \pi} \operatorname{Cos}[k \pi]}{1 + e^{2 c k \pi}}\right\}$$

$$\frac{2\, e^{2 c k \pi} \operatorname{Cos}[k \pi]^2}{1 + e^{2 c k \pi}}$$

$$\{e^{c k \pi} \operatorname{Cos}[k \pi],\ e^{c k \pi} \operatorname{Sin}[k \pi]\}$$

$$\left\{\frac{2\, e^{2 c k \pi} \operatorname{Cos}[k \pi]^2}{1 + e^{2 c k \pi}},\ \frac{2\, e^{2 c k \pi} \operatorname{Cos}[k \pi] \operatorname{Sin}[k \pi]}{1 + e^{2 c k \pi}}\right\}$$

The bending line with one endpoint at 1 and the other endpoint at *vec*[*c,2k*] is a vertical semicircle in upper half 3–space. We can determine a point on this semicircle by specifying the angle from the centre of the semicircle, with 0 angle representing 1 and angle π representing *vec*[*c,2k*]. Let *cosAngInter* be the cosine of the angle at the intersection point $\alpha \bigcap \gamma$. This cosine can be read off in the plane from *tssol* above. We will use the sine and cosine to determine $\alpha \bigcap \gamma$ as a point of upper half 3–space.(1–2*t*) is the quantity that tells us how far to go round the semicircle.

```
Simplify[Together[1 - 2 t /. tssol]][[1]]
```

$$\frac{-1 + e^{2 c k \pi}}{1 + e^{2 c k \pi}}$$

It's hard to get *Mathematica* to simplify this. So we give up and type in the values by hand.

```
Clear[cosAngInt, sinAngInt];
cosAngInt = Tanh[c * k * π];
sinAngInt = Sech[c * k * π];
```

The next thing to find is $\alpha \bigcap \gamma$ in terms of c and k. We also check our previous calculations.

```
Clear[c, k, centreSemiCircle, norm, horizVect, nHV, zcoord, alphaInterGamma];
centreSemiCircle = Flatten[{({1, 0} + vec[c, 2 k]) / 2, 0}];
norm[{u_, v_, w_}] := Sqrt[u^2 + v^2 + w^2];
horizVect = Flatten[{({1, 0} - vec[c, 2 k]) / 2, 0}];
nHV = norm[horizVect];
alphaInterGamma[c_, k_] =
  centreSemiCircle + cosAngInt * horizVect + sinAngInt * {0, 0, nHV};
xycoords1[c_, k_] = Drop[alphaInterGamma[c, k], -1];
zcoord[c_, k_] := alphaInterGamma[c, k][[3]];
xycoords[c, k] - xycoords1[c, k] // Simplify
```

```
{0, 0}
```

The formula for a point at angle θ along the bending line is

```
centreSemiCircle + Cos[θ].horizVect + sin[θ].{0, 0, nHV}
```

The tangent vector to the bending line at θ is

```
-sin[θ].horizVect + cos[θ].{0, 0, nHV}
```

Now let's work out a tangent vector (not of unit hyperbolic length) to the bending line at the intersection point $\alpha \bigcap \gamma$, pointing from 1 to *vec*[*c,2k*].

```
Clear[tangVectToBendingLine] ;
tangVectToBendingLine[c_, k_] = -Sech[c * k * π] * horizVect + Tanh[c * k * π] {0, 0, nHV} ;
```

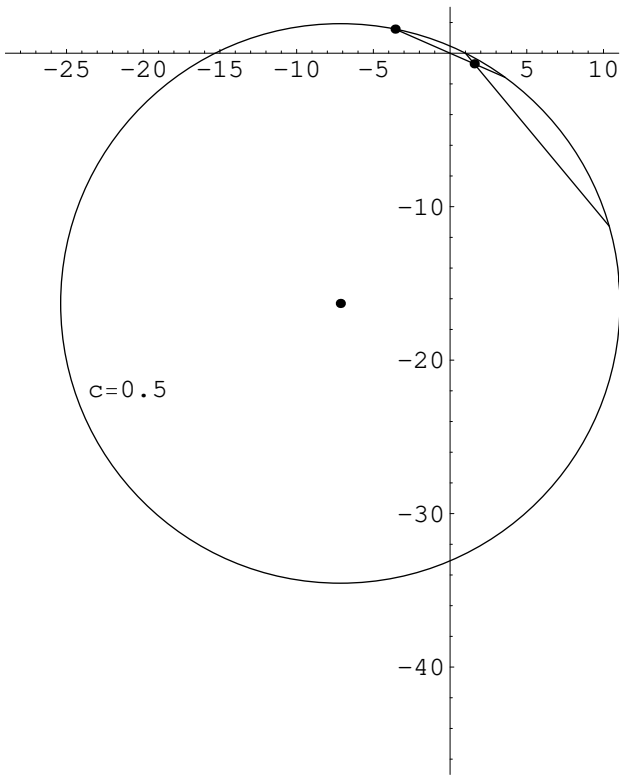
.Now we go back to drawing some pictures, just to make sure that we haven't made some terrible blunder. Below we will show a plot of the circle containing the four endpoints. This is **not** the maximal circle. We show the vertical projection of the bending line and the vertical projection of the fixed points of the elliptic involution. The second of these is the interval through the origin.

We first find the circle that contains the four endpoints 1, *vec*[*c,2k*], *vec*[*c,k*], −*vec*[*c,k*].

```
Clear[circ, circ1, c, k, x, y, x0, y0];
circ[{x_, y_}] := (x - x0) ^2 + (y - y0) ^2 - rSquared; (*the equation of the circle*)
circ1[c_, k_] = {{x0, y0}, Sqrt[rSquared]} /.
   Solve[{circ[{1, 0}] == 0, circ[vec[c, k]] == 0, circ[-vec[c, k]] == 0},
     {x0, y0, rSquared}][[1]]; (*the solution for the centre and
  the radius squared of the circle containing the four points.*)
```

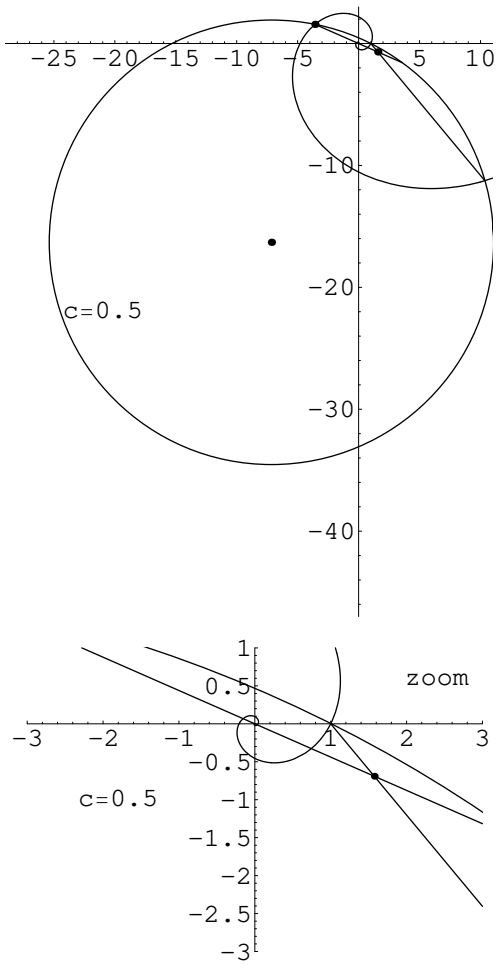
Below we show a plot of the circle containing the four endpoints. This is **not** the maximal circle. We show the vertical projection of the bending line and the vertical projection of the fixed points of the elliptic involution. The second of these is the interval through the origin.

```
Clear[circ2, k, c, l1, l2, pts, txt];
c = .5; k = time[c];
l1 = Line[{{1, 0}, vec[c, 2 k]}];
l2 = Line[{vec[c, k], -vec[c, k]}];
pts = {PointSize[.015], Point[xycoords[c, k]],
   Point[circ1[c, k][[1]]], Point[vec[c, k]]};
 circ2 = Circle @@ circ1[c, k];
str = StringJoin["c=", ToString[c]];
txt = Text[str, Scaled[{.2, .5}]];
spiral = ParametricPlot[{Re[sp[t]], Im[sp[t]]}, {t, -2, 1.01},
   DisplayFunction → Identity];
Show[{Graphics[{l1, l2, pts, circ2, txt}]},
  {
   Axes → True,
   AspectRatio → Automatic,
   DisplayFunction → $DisplayFunction,
   PlotRange → {{-29, 11}, {-47, 3}}
  }
 ];
```



Next we see how the spiral fits in with the circle, and after zooming in.

```
Show[{spiral, Graphics[{l1, l2, pts, circ2, txt}]},
  {
   Axes → True,
   AspectRatio → Automatic,
   DisplayFunction → $DisplayFunction,
   PlotRange → {{-29, 11}, {-47, 3}}
  }
 ];
Show[{spiral, Graphics[{l1, l2, pts, circ2, txt, Text["zoom", Scaled[{.9, .9}]]}]},
  {
   Axes → True,
   AspectRatio → Automatic,
   DisplayFunction → $DisplayFunction,
   PlotRange → {{-3, 3}, {-3, 1}}
  }
 ];
```



Above we see how the spiral fits in with the circle, and after zooming in.

■ Computing bending angles.

In order to work out the derivative of the bending measure with respect to *t*, wewill find the angle at which the maximal circle meets its image after time *t* under the spiral flow. We firstwork out the angle between two arbitrary circles

```
Clear[x, y, x1, y1, x2, y2, r1, r2];
Clear[intersectionPtSoln, intersectionPt, bending];
Simplify[Solve[{
      (x - x1)^2 + (y - y1)^2 == r1^2,
      (x - x2)^2 + (y - y2)^2 == r2^2
    }, {x, y}][[1]]];
{x, y} /. %;
Simplify[%, x1 > x2];
intersectionPt[{{x1_, y1_}, r1_}, {{x2_, y2_}, r2_}] = %;
```

Note that the simplification two lines above does not work with $x1 \neq x2$. This seems slightly unreasonable of *Mathematica,* but it may be because it needs the square root of $(x1 - x2)^2$. We get an answer only up to sign, but this doesn't matter—

```
bending[{{x1_, y1_}, r1_}, {{x2_, y2_}, r2_}] := Module[{v1, v2},
   v1 = {x1, y1} - intersectionPt[{{x1, y1}, r1}, {{x2, y2}, r2}];
   v2 =  {x2, y2} - intersectionPt[{{x1, y1}, r1}, {{x2, y2}, r2}];
   ArcTan @@ v1 - ArcTan @@ v2
   (*ArcTan can take two coordinates as argument in Mathematica,
    and then it gives a value in the quadrant of the corresponding vector*)
  ];
```

Applying the 1–parameter group, the circle moves according to the matrix derived from complex multiplication by $e^{2\pi t(i+c)}$. The radius changes according to the absolute value only. The bending is then an analytic function of *t*. We are interested only in the linear coefficient. Actually the bending is positive. But putting in absolute values would destroy the analyticity, so we don't force the sign to be positive.

```
Clear[c, k];
mat[t_, c_] = Exp[2 Pi * t * c] {
    {Cos[2 Pi * t], -Sin[2 Pi * t]},
    {Sin[2 Pi * t], Cos[2 Pi * t]}};
```

We express the bending as a function of *t* and then take a power series. The derivative is the coefficient of the term *t*. We use the function *LeafCount* to see how complicated the expressions get

```
ben = Series[bending[{centre[c, k], r[c, k]},
                {mat[t, c].centre[c, k], Exp[2 Pi * t * c] * r[c, k]}], {t, 0, 1}];
bendingDerivative[c_, k_] = SeriesCoefficient[ben, 1];
LeafCount[ben]
LeafCount[bendingDerivative[c, k]]
```

```
59534
```

```
59527
```

■ Computing the hyperbolic speed.

Now we want to calculate the hyperbolic speed at t=0 of the special point $\alpha \cap \gamma$ for which we have worked out coordinates. We work out the first two terms of the power series in *t* and then concentrate on the derivative as a function of *c* and *time[c]*

```
Clear[u, v, w, deriv, hypspeed];
Series[#, {t, 0, 1}] & /@
  Flatten[{mat[t, c].xycoords[c, k], Exp[2 Pi * t * c] * zcoord[c, k]}];
deriv[c_, k_] = SeriesCoefficient[#, 1] & /@ %;
hypspeed[c_, k_] = norm[ deriv[c, k] / zcoord[c, k]];
(*the hyperbolic length of the tangent vector to γ, when parametrized by t*)
```

■ Bending angle per unit time divided by hyperbolic speed.

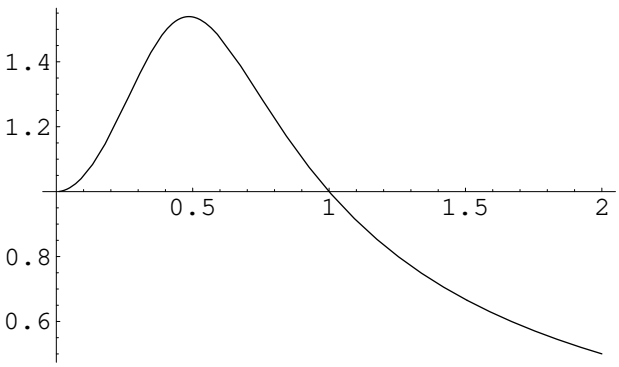
We take the bending derivative divided by the hyperbolic speed. Let's see how complicated the expression is

```
quot[c_, k_] = bendingDerivative[c, k] / hypspeed[c, k];
LeafCount[quot[c, k]]
```

```
59763
```

The expression for bending per unit length has nearly 60K terms.

```
Plot[quot[c, time[c]], {c, .01, 2}]
```




```
- Graphics -
```

Now let's scale to see how the maximum matches with $\pi/2$.
We sim zoom in on a certain part of the graph, so that we can read off the maximum more accurately. (This part commented out to reduce the time the program takes to run.)

```
(*
 Off[General::spell1];
Off[General::spell];
 xmin=.48558;
 xmax=.48622; ymin=.979894; ymax=.979898;
xvalue=.48589; yvalue=.9798961;
p=Plot[2*quot[c,time[c]]/Pi,{c,xmin,xmax},
   DisplayFunction→Identity];
ls=Graphics[
   {Line[{{xmin ,yvalue},{xmax,yvalue}}],Line[{{xvalue,ymin},{xvalue,ymax}}]}];
Show[{ls,p},{
   Axes→True,
   PlotRange→{{xmin ,xmax},{ymin,ymax}},
   AxesOrigin→{xmin ,ymin},
   DisplayFunction→$DisplayFunction
  }];
On[General::spell1];
 On[General::spell];
*)
```

We see that the maximum bending is .9798961× π / 2. This occurs with the logarithmic spiral having c=.48589. Let's redo this with an accurate computation. The last line shows that the first 66 digits of the answer are correct. In particular, the maximum long range bending per unit hyperbolic length three lines up has all its digits correct.

```
c = .48589`70; k = carefulTime[c]
quot[c, k] * 2 / Pi
Precision[%]
```

```
0.872723233891954755441910881601390781745436070547266021376481876043821 5

0.979896089578668830423463415210944582179714452493790679079331928151

66
```

■ The Riemann mapping and the extremal qc map

We will work out the Riemann mapping $f : \Omega \longrightarrow U^2$ with domain the complement of the logarithmic spiral and with values in the upper half plane . This turns out to be:

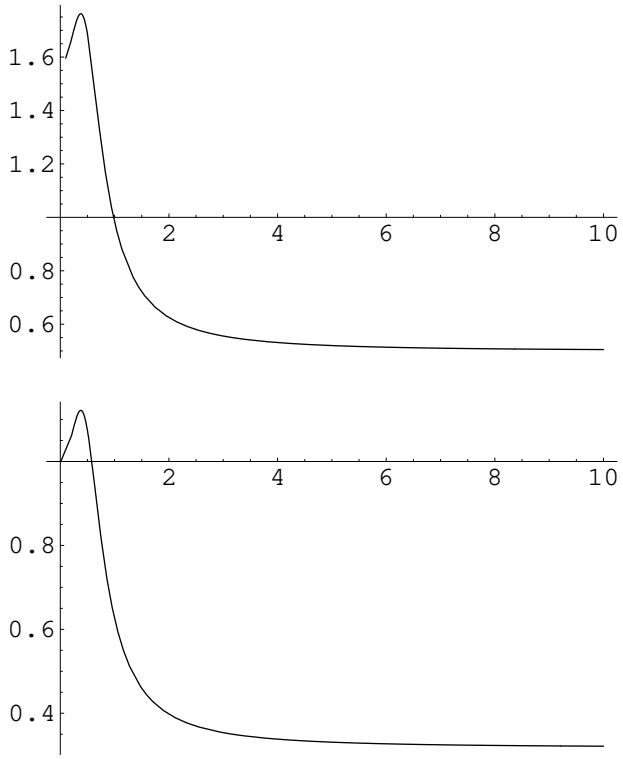
$$f(z) = \exp\left(-\left(\frac{c-i}{2c}\right)\log(z)\right)$$

$$f(z\exp(2\pi t(i+c))) = \exp\left(-\left(\frac{c-i}{2c}\right)(\log z + 2\pi t(i+c))\right)$$

$$= f(z) \cdot \exp\left(-\pi t\left(c+\frac{1}{c}\right)\right)$$

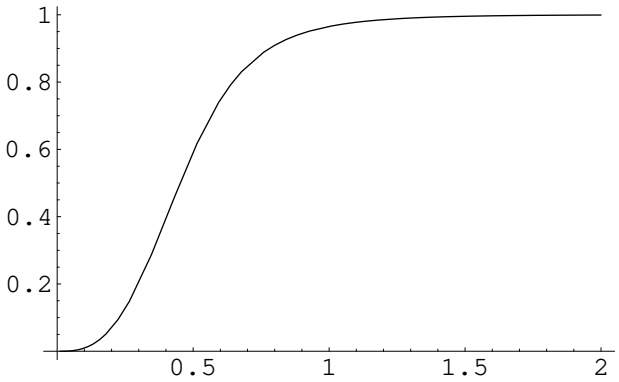
The derivative at $f(z) = i$, with $t = 0$, is $-i\pi\left(c+\frac{1}{c}\right)$. In hyperbolic coordinates, the speed of the curve is $\pi\left(c+\frac{1}{c}\right)$. Now we work out the speed in hyperbolic 3−space divided by the speed in the floor, with the hyperbolic metric. The first plot confirms that the ratio of speeds, as c tends to ∞, the ratio tends to 1/2. The second plot is rescaled vertically by $\pi / 2$. This confirms that, as c tends to 0, the ratio of speeds tends to $\pi / 2$, which is another confirmation that our computations are accurate. The results give nothing new.

```
Plot[(π (c + 1 / c)) / hypspeed[c, time[c]], {c, 0.1, 10}];
Plot[(2 (c + 1 / c)) / hypspeed[c, time[c]], {c, 0.01, 10}];
```



We need to work out the angle between the tangent vector to the bending line, pointing from 1 to *vec*[*c*,2*k*], and the tangent to the geodesic running along the top of the snail shell, pointing from −∞to ∞. First we give a general formula for the cosine of the angle in 3−space.

```
Clear[u, v, c, k, cosAng, cosAngAlphaInterGamma, cAAIG];
cosAng[u_, v_] := u.v / (norm[u] * norm[v]);
cosAngAlphaInterGamma[c_, k_] = cosAng[deriv[c, k], tangVectToBendingLine[c, k]];
cAAIG[c_, k_] := cosAngAlphaInterGamma[c, k];
Plot[cAAIG[c, time[c]], {c, .01, 2}];
```

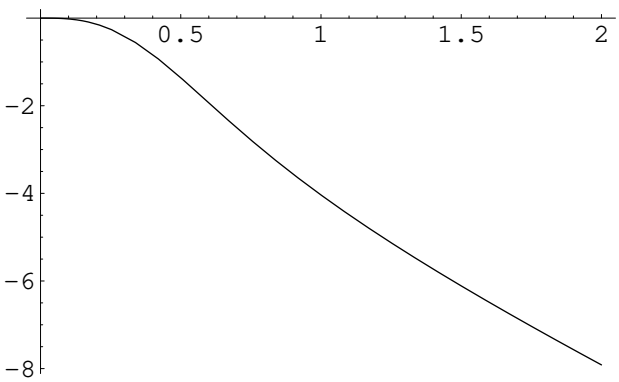


We can now transfer the picture to the upper half plane. Here γ is the positive y-axis.
The bending line is a semicircle with endpoints $1+\cos(\varphi)$ and $-1+\cos(\varphi)$,corresponding to *vec*[*c*,2*k*] and to 1 respectively. Here $\cos(\varphi)$ is given by the expression

```
cosAngAlphaInterGamma[c, time[c]]
```

We map the upper half plane by log to a horizontal strip of height π. The endpoints of the bending line are mapped to 0 and to $u=-\log\left(\frac{1+\cos(\varphi)}{1-\cos(\varphi)}\right)+\pi i$. The point on the upper edge is to the left of the point on the lower edge. We plot u against c. As expected, it gives 0 at c=0.

```
Clear[b, c, k, t, v, u];
u[c_, k_] = -Log[(1 + cAAIG[c, k]) / (1 - cAAIG[c, k])];
u[c_] := u[c, time[c]];
Plot[u[c], {c, .001, 2}]
```



- Graphics -

Now we need to do the same type of calculation for the spiral region. This time we apply the mapping

$$f(z) = -\left(\frac{c-i}{2c}\right)\log(z).$$

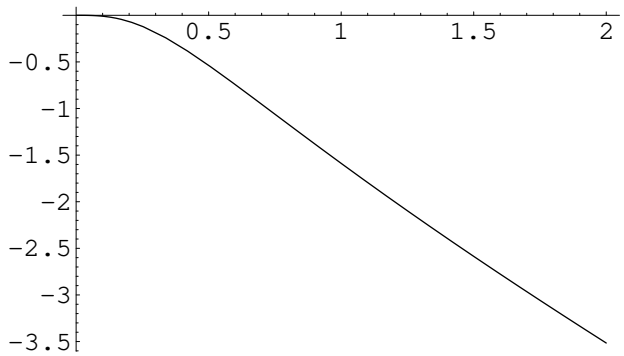
which maps the complement of the logarithmic spiral to the horizontal strip of height π.
Under this map, 1 goes to 0, and the point of the spiral at time t goes to

$$-\left(\frac{c-i}{2c}\right) 2\pi t\,(c+i) = -\pi t\left(c+\frac{1}{c}\right)$$

Because the segment from 1 to *vec*[*c*,2*k*] together with the spiral curve from *t*=0 to *t*=*k*=*time*(*c*) goes once around the origin, applying *f* to the vertical projection of the bending line into Ω gives a curve in the horizontal strip from 0 to $-\left(\frac{c-i}{2c}\right)(2\pi t\,(c+i)-2\pi i)$. The image of the bending line in the horizontal strip has one endpoint at 0 and the other endpoint at $v + i\pi$.

v is plotted below. As we expect, the value at *c*=0 is 0.

```
v[c_, t_] = ComplexExpand[Re[-((c - i)/(2 c)) (2 π t (c + i) - 2 π i)]];
v[c_] := v[c, time[c]];
Plot[v[c], {c, .001, 2}];
```



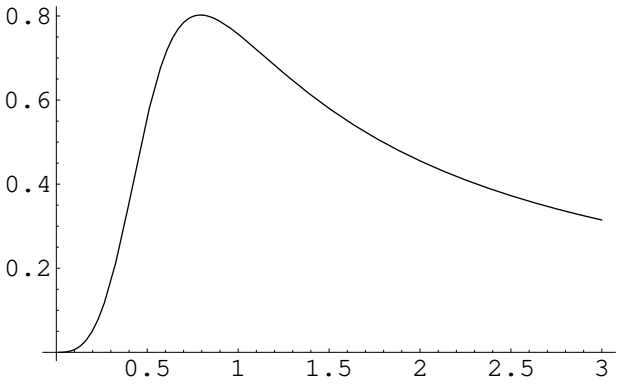
To sum all this up, we map the horizontal strip of height π to itself. 0 is sent to 0. We regard the dome as the domain and the floor Ω as the range. The identity map on the logarithmic spiral between the boundary of the convex hull boundary and the boundary of the region Ω translates to the restriction of an affine map from the boundary of the strip to the boundary of the strip. This is an inevitable consequence of the equivariance of the group action (which is a group of translations on the infinite strip). Now such boundary values correspond to a uniquely extremal affine map on the whole strip (Strebel) and so we can compute precisely the best possible qc map between convex hull boundary and region Ω. We work in the infinite horizontal strip of width π.

The horizontal stretch factor is then

```
Clear[a, c, k];
a[c_, k_] := (π (c + 1 / c)) / hypspeed[c, k];
a[c_] := a[c, time[c]];
```

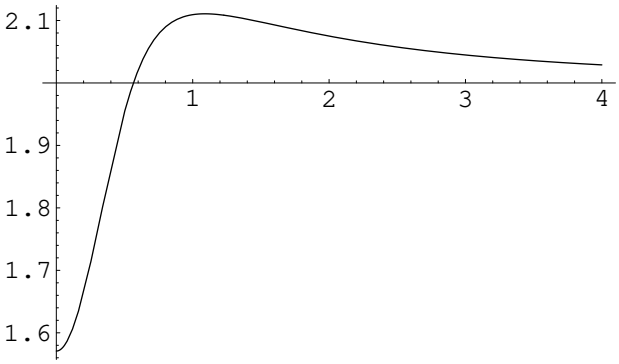
We have to find *b* so that *x*−>*ax*+*by*, *y*−>*y*sends 0 to 0 on the lower edge,
and {*u*,π} is sent to {*v*,π}

```
Clear[b, c, k, x, y];
b[c_, k_] := (-a[c, k] * u[c, k] + v[c, k]) / π;
b[c_] := b[c, time[c]];
Plot[b[c], {c, 0.01, 3}]
```



- Graphics -

```
Clear[compDil, dil, c, k];
compDil[c_, k_] := (a[c, k] - 1 + I * b[c, k]) / (a[c, k] + 1 - I * b[c, k]);
dil[c_, k_] := (1 + Abs[compDil[c, k]]) / (1 - Abs[compDil[c, k]]);
compDil[c_] := compDil[c, time[c]];
dil[c_] := dil[c, time[c]];
Plot[dil[c], {c, .0001, 4},
 PlotRange → All]
```



- Graphics -

The expression for the dilatation is less complicated than the expression for the bending. It has only 7745 terms. We will also do a careful computation of the dilatation at c=1. We see below that the expression 2.1 etc below for the dilatation is accurate to 67 digits. This shows that the Thurston K=2 conjecture is false.

```
LeafCount[dil[c, k]]
dil[1, carefulTime[1]]
Precision[%]
```

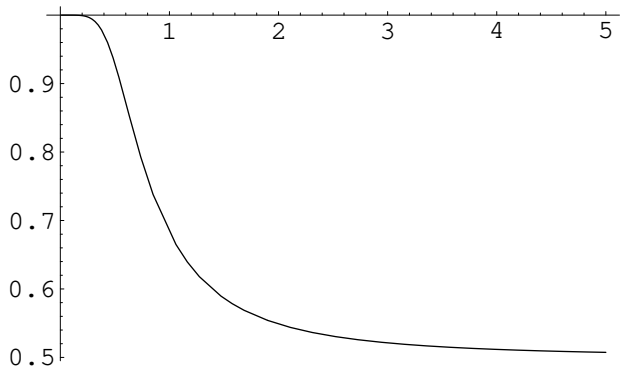
```
6161
```

```
2.109336190488874443225646947802086853981275070983385930376902875179 1
```

```
67
```

Next we compute Lipschitz constants associated with the uniquely extremal qc map. We find nothing remarkable.

```
Plot[(Sqrt[(1 + a[c])^2 + b[c]^2] - Sqrt[(1 - a[c])^2 + b[c]^2]) / 2,
 {c, 0.01, 5}, AxesOrigin → {0, 1}]
```




```
- Graphics -
```

```
? centreSemiCircle
```

```
Global`centreSemiCircle
```

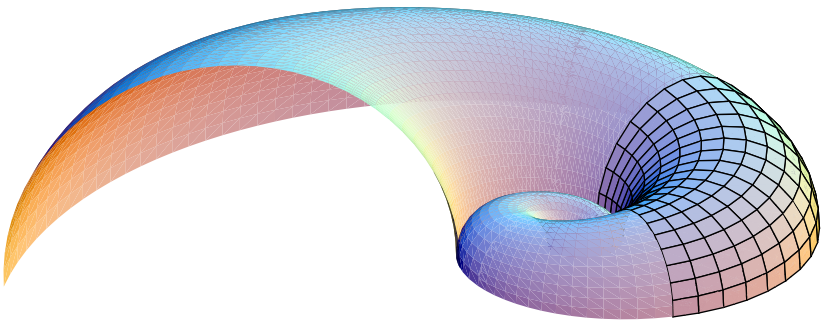
```
centreSemiCircle = {1/2 (1 + e^(2 c k π) Cos[2 k π]), 1/2 e^(2 c k π) Sin[2 k π], 0}
```

```
Remove[c, k, s, t, centreSemiCircle, horizVect, nHV];
centreSemiCircle[c_, k_, t_] = Flatten[{mat[t, c].({1, 0} + vec[c, 2 k]) / 2, 0}];
horizVect[c_, k_, t_] = Flatten[{mat[t, c].({1, 0} - vec[c, 2 k]) / 2, 0}];
nHV[c_, k_, t_] = norm[horizVect[c, k, t]];

c = .25; k = time[c]; startt = -1; endt = 1.4; deltat = .3;
myGraphicsList = {};
Off[ParametricPlot3D::ppcom];
Do[{p = ParametricPlot3D[centreSemiCircle[c, k, t] + Cos[s] * horizVect[c, k, t] +
      Sin[s] * {0, 0, nHV[c, k, t]}, {s, 0, π}, {t, tstep, tstep + deltat},
     {
      PlotPoints → Floor[Exp[tstep] * 10],
      PlotRange → All,
      DisplayFunction → Identity
     }],
   AppendTo[myGraphicsList, p]
  }, {tstep, startt, endt, deltat}];

q = ParametricPlot3D[centreSemiCircle[c, k, t] +
    1.01 * (Cos[s] * horizVect[c, k, t] + Sin[s] * {0, 0, nHV[c, k, t]}),
   {s, 0, π}, {t, 1, 1 + deltat},
   {
    PlotPoints → Floor[15],
    PlotRange → All,
    DisplayFunction → Identity
   }];
```

We show part of the snail shell. Some of the bending lines are drawn. We also draw part of the spiral curves. Each spiral curve is a line of constant distance from the main geodesic.

```
sh = Show[{myGraphicsList /. Graphics3D[x_, y_] → Graphics3D[{EdgeForm[], x}, y], q},
  {Axes → None,
   Boxed → False,
   PlotLabel -> "Snail shell for c=" <> ToString[c],
   ViewPoint -> {2.3, -2.4, 1.4},
   DisplayFunction → $DisplayFunction
  }]
```

Snail shell for c=0.25



■ Brennan's Conjecture

One can find the L^p class of the derivative of the Riemann mapping explicitly, using an infinite series to perform the computation. The complement of the spiral is cut into a countable number of pieces by the positive x–axis.We will not perform the computation explicitly, because it leads to nothing new.