

## □ **Isosurface Dabbling**

Following:  
[http://maxima.sourceforge.net/docs/manual/maxima\\_18.html](http://maxima.sourceforge.net/docs/manual/maxima_18.html)  
<http://gkerns.people.ysu.edu/maxima/>

```
(%i1) load("vect");
      load("eigen");
(%o1) /Applications/Maxima.app/Contents/Resources/maxima/share/maxima/5.36.1/share/vector/vect.mac
(%o2) /Applications/Maxima.app/Contents/Resources/maxima/share/maxima/5.36.1/share/matrix/eigen.mac
```

## □ **1 Summary**

I wanted to find a distance bound for my beloved 2005 isosurface that I used in POV-Ray to make the corkscrew thingy. (Why: because I wanted to render it directly in a sphere tracer in GLSL, or in Appleseed.)

I attempted to follow the method described in Hart's Sphere Tracing paper, in which any implicit surface whose function is Lipschitz continuous can be turned to a distance bound function for that same surface. This proved very difficult since I am fairly certain that the function I created is *not* Lipschitz continuous. There is also a pile of other failed methods here after I dove down a rabbit-hole of vector calculus that I didn't really understand very well, and then attempted to use CGAL to convert this isosurface to a mesh, but it was creating meshes that were too full of facetization artifacts (especially along the 'edge' of the corkscrew), so I was trying to refine things. This is an unfinished mess, but the work is left here.

At some point I realized my function already *is* an SDF. See section 2.1. The formula works - I tried it in a sphere tracer in ShaderToy. See: <https://www.shadertoy.com/view/ttsXD4>

The sections after 2.1 try to match up the 2005 formula and hand-derived formula and prove that they are not just the same surface, but identical formulas - provided that parameters are properly matched up.

### □ **1.1 Questions**

- How do I have an SDF that is not Lipschitz continuous?  
 Is that a 'sufficient, but not necessary' condition to produce a distance estimate for a surface?

### □ **1.2 Probably just me whining**

- Can I do this in org-mode & babel? Jupyter? wxMaxima annoys me.  
 - Can I easily write this to let me work in x,y,z rather than subscripts of a vector argument?

## □ **2 Spiral Isosurface Thingy**

as first explored in my 2005 notes

```
(%i3) f3(p) := sqrt((p[2] * O - I * sin(p[1] * F + P))^2 + (p[3] * O - I * cos(p[1] * F + P))^2);
(%o3) f3(p) := sqrt((p2 O - I sin(p1 F + P))^2 + (p3 O - I cos(p1 F + P))^2)
```

### □ **2.1 SDFs derived by hand**

I started from:  
<http://www.iquilezles.org/www/articles/distfunctions/distfunctions.htm>  
 and looked at opTwist & sdCylinder, which I was able to combine to produce exactly the same surface.  
 The below is an exact SDF for cylinder along z (not a distance estimate).

```
--> cylSdf(p) := sqrt(p[1]^2 + p[2]^2) - R;
(%o4) cylSdf(p) := sqrt(p1^2 + p2^2) - R
```

and here is a domain transformation for a twist, starting with an XY rotation matrix I use F2 instead of F.

```

--> rot : ev(matrix([C,-S, 0],[S,C,0],[0,0,1]), C=cos(F2*a-P), S=sin(F2*a-P));
(%o48) 
$$\begin{bmatrix} \cos(F2 a - P) & -\sin(F2 a - P) & 0 \\ \sin(F2 a - P) & \cos(F2 a - P) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$


```

```

--> twist(p) := ev(rot, a=p[3]) . [p[1], p[2], p[3]];
(%o49) twist(p) := ev(rot, a=p[3]) . [p[1], p[2], p[3]]

```

```

--> twist([x,y,z]);
(%o50) 
$$\begin{bmatrix} x \cos(F2 z - P) - y \sin(F2 z - P) \\ x \sin(F2 z - P) + y \cos(F2 z - P) \\ z \end{bmatrix}$$


```

so stacking a few transformations should produce an identical surface...

```

--> f : cylSdf(twist([x,y,z]) - [x0,y0,0])[1];
(%o51) 
$$\sqrt{(-y \sin(F2 z - P) + x \cos(F2 z - P) - x0)^2 + (x \sin(F2 z - P) + y \cos(F2 z - P) - y0)^2} - R$$


```

```

--> trigsimp(f);
(%o52) 
$$\sqrt{(2 x0 y - 2 x x0) \sin(F2 z - P) + (-2 y y0 - 2 x x0) \cos(F2 z - P) + y0^2 + y^2 + x0^2 + x^2} - R$$


```

This, tested in ShaderToy, works perfectly.  
The constants may need some correction to match exactly what is used in my 2005 formula, but that should not be much of a problem.

## 2.2 Why are these different?

The below is my 2005 formula. Only modification is that I pass z,y,x instead of x,y,z.

```

--> g : ev(f3([z,y,x])-T);
(%o53) 
$$\sqrt{(O y - I \sin(F z + P))^2 + (O x - I \cos(F z + P))^2} - T$$


```

```

--> trigsimp(g);
(%o54) 
$$\sqrt{-2 I O y \sin(F z + P) - 2 I O x \cos(F z + P) + O^2 y^2 + O^2 x^2 + I^2} - T$$


```

Evaluated at 4 phases of its full period:

```

--> makelist(ev(((g = 0) + T)^2, [z=i*pi/(2*F)]), i, 0, 3);
(%o55) [(O y - I sin(P))^2 + (O x - I cos(P))^2 = T^2, (O y - I cos(P))^2 + (O x + I sin(P))^2 = T^2, (O y + I sin(P))^2 + (O x + I cos(P))^2 = T^2, (O y + I cos(P))^2 + (O x - I sin(P))^2 = T^2]

```

and the derived SDF:

```

--> makelist(ev(((f = 0) + R)^2, [z=i*pi/(2*F2)]), i, 0, 3);
(%o56) [(-y0 + cos(P) y - sin(P) x)^2 + (sin(P) y - x0 + cos(P) x)^2 = R^2, (-y0 + sin(P) y + cos(P) x)^2 + (-cos(P) y - x0 + sin(P) x)^2 = R^2, (-y0 - cos(P) y + sin(P) x)^2 + (-sin(P) y - x0 - cos(P) x)^2 = R^2, (-y0 - sin(P) y - cos(P) x)^2 + (cos(P) y - x0 - sin(P) x)^2 = R^2]

```

Sorry for atrocious variable names - I was 18. See the following replacements:

```

--> replace : [y0=0, x0=I/O, R=(T/O), F2=-F];
(%o57) [y0=0, x0=I/O, R=T/O, F2=-F]

```

...applied to the derived SDF:

```

--> trigsimp(ev(((f = 0) + R)^2, replace))*O^2;
(%o58) -2 I O y sin(F z + P) - 2 I O x cos(F z + P) + O^2 y^2 + O^2 x^2 + I^2 = T^2

```

✓ This is indeed identical to the 2005 one:

```

--> sdf : ((trigsimp(g)=0)+T)^2;
(%o59) -2 I O y sin(F z + P) - 2 I O x cos(F z + P) + O^2 y^2 + O^2 x^2 + I^2 = T^2

```

## □ 2.3 Analytical gradient

✓ (still needs normalized. ugly denominators can probably be ignored.)

```

--> ev(express(grad(lhs(sdf))), diff);
(%o60) [ 2 O^2 x - 2 I O cos(F z + P), 2 O^2 y - 2 I O sin(F z + P), 2 F I O x sin(F z + P) - 2 F I O y cos(F z + P) ]

--> gr : trigsimp(ev(express(grad(f)), diff));
(%o62) [ - (y0 sin(F2 z - P) + x0 cos(F2 z - P) - x) / (sqrt((2 x0 y - 2 x y0) sin(F2 z - P) + (-2 y y0 - 2 x x0) cos(F2 z - P) + y0^2 + y^2 + x0^2 + x^2) * (x0 sin(F2 z - P) - y0 cos(F2 z - P) + y) * sqrt((2 x0 y - 2 x y0) sin(F2 z - P) + (-2 y y0 - 2 x x0) cos(F2 z - P) + y0^2 + y^2 + x0^2 + x^2) * ((F2 y y0 + F2 x x0) sin(F2 z - P) + (F2 x0 y - F2 x y0) cos(F2 z - P)) / sqrt((2 x0 y - 2 x y0) sin(F2 z - P) + (-2 y y0 - 2 x x0) cos(F2 z - P) + y0^2 + y^2 + x0^2 + x^2) ],

--> optimize(gr);
(%o63) block ([ %1, %2, %3, %4 ], %1:F2 z - P, %2:cos(%1), %3:sin(%1), %4: 1 / (sqrt((2 x0 y - 2 x y0) %3 + (-2 y y0 - 2 x x0) %2 + y0^2 + y^2 + x0^2 + x^2) * ((F2 y y0 + F2 x x0) %3 + (F2 x0 y - F2 x y0) %2) ) , [ -(y0 %3 + x0 %2 - x) %4, (x0 %3 - y0 %2 + y) %4, %4 ]

```

## □ 3 I suck at vector calculus

✓ This was moved later on in the file on account of being basically useless.

The point of this work is mostly to verify that some scratch work I did with the CGAL library based around the code at [https://doc.cgal.org/latest/Surface\\_mesher/index.html#title3](https://doc.cgal.org/latest/Surface_mesher/index.html#title3) is valid.

The issue I was trying to tackle is that the meshes it generated were not quite optimal for me (perhaps due to me having different goals than a package for computational geometry for CAD). The positions of vertices in the mesh seemed to have a lot of random noise, even if the overall error was bounded and could be reduced arbitrarily by using more facets. I was looking at optimizing the mesh by doing per-vertex gradient descent to nudge each vertex closer to the surface itself, which the definition of an isosurface makes straightforward: for point  $F(x, y, z)$ , the closer  $F$  is to 0, the closer it is to the surface. Gradient descent of course requires the gradient. I was dealing with arbitrary chains of  $\mathbb{R}^3 \rightarrow \mathbb{R}^3$  functions being composed with an  $\mathbb{R}^3 \rightarrow \mathbb{R}$  function to produce the isosurface  $\mathbb{R}^3 \rightarrow \mathbb{R}$ , and didn't feel like working out derivatives by hand.

Chris Hodapp, 2018-07-27

✓ Note that the implicit surface itself is  $f_3(x, y, z) = T$  where  $T$  is some threshold. I use the squared version below because it doesn't actually change the isosurface's properties, but is easier to work with.

```

(%i4) surface : f3([x,y,z])^2 - T^2;
(%o4) -T^2 + (y O - I sin(P + x F))^2 + (z O - I cos(P + x F))^2

(%i5) f_x : diff(surface, x);
(%o5) 2 F I (z O - I cos(P + x F)) sin(P + x F) - 2 F I cos(P + x F) (y O - I sin(P + x F))

(%i6) f_y : diff(surface, y);
(%o6) 2 O (y O - I sin(P + x F))

(%i7) f_z : diff(surface, z);
(%o7) 2 O (z O - I cos(P + x F))

```

```
(%i8) f_yy : diff(f_y, y);
(%o8) 2 O^2
```

```
(%i9) f_zz : diff(f_z, z);
(%o9) 2 O^2
```

```
(%i10) f_yz : diff(f_y, z);
(%o10) 0
```

because I might forget:

```
(%i11) f_zy : f_yz;
(%o11) 0
```

note that K below is the 2D curvature within the slice.,  
it is not the 3D curvature of the surface!

```
--> K : (-f_y^2 * f_zz + 2*f_z*f_y*f_zy - f_z^2*f_yy) / ((f_z^2 + f_y^2)^(3/2));
(%o19) 
$$\frac{-8 O^4 (y O - I \sin(P + x F))^2 - 8 O^4 (z O - I \cos(P + x F))^2}{(4 O^2 (y O - I \sin(P + x F))^2 + 4 O^2 (z O - I \cos(P + x F))^2)^{3/2}}$$

```

```
--> K_y : diff(K, y);
(%o20) 
$$-\frac{16 O^5 (y O - I \sin(P + x F))}{(4 O^2 (y O - I \sin(P + x F))^2 + 4 O^2 (z O - I \cos(P + x F))^2)^{3/2}} - \frac{12 O^3 (y O - I \sin(P + x F)) (-8 O^4 (y O - I \sin(P + x F))^2 - 8 O^4 (z O - I \cos(P + x F))^2)}{(4 O^2 (y O - I \sin(P + x F))^2 + 4 O^2 (z O - I \cos(P + x F))^2)^{5/2}}$$

```

```
--> K_z : diff(K, z);
(%o21) 
$$-\frac{16 O^5 (z O - I \cos(P + x F))}{(4 O^2 (y O - I \sin(P + x F))^2 + 4 O^2 (z O - I \cos(P + x F))^2)^{3/2}} - \frac{12 O^3 (z O - I \cos(P + x F)) (-8 O^4 (y O - I \sin(P + x F))^2 - 8 O^4 (z O - I \cos(P + x F))^2)}{(4 O^2 (y O - I \sin(P + x F))^2 + 4 O^2 (z O - I \cos(P + x F))^2)^{5/2}}$$

```

we can try solving for singular points of the entire surface,  
but while we can find singular points on some surface, and  
can find points on the surface we want, we can't seem to find  
both:

```
--> solve([f_x = 0, f_y = 0, f_z = 0], [x,y,z]);
(%o22) [ [ x = %r1, y =  $\frac{I \sin(P + \%r1 F)}{O}$ , z =  $\frac{I \cos(P + \%r1 F)}{O}$  ] ]
```

```
--> solve([surface = 0], [x,y,z]);
(%o23) [ [ x = %r2, y = %r3, z =  $\frac{\sqrt{T^2 - I^2 \sin(P + \%r2 F)^2 + 2 \%r3 I O \sin(P + \%r2 F) - \%r3^2 O^2 + I \cos(P + \%r2 F)}}{O}$  ], [ x = %r4, y = %r5, z =  $-\frac{\sqrt{T^2 - I^2 \sin(P + \%r4 F)^2 + 2 \%r5 I O \sin(P + \%r4 F) - \%r5^2 O^2 - I \cos(P + \%r4 F)}}{O}$  ] ]
```

```
--> solve([f_x=0, f_y=0, f_z=0, surface=0], [x,y,z]);
(%o24) [ ]
```

no singular points? may need lagrange multipliers after all...

guess I should start with the simpler 2D case.  
that is: finding extrema of K under the constraint that surface=0.

```

--> extrema : solve([surface = 0, K_y = -L * f_y, K_z = -L * f_z], [L, y, z]);
(%o25) [ [ L = - sqrt( O^4 ( 4 ( sqrt(T^2 - I^2 sin(P + x F)^2 + 2 %r6 I O sin(P + x F) - %r6^2 O^2 - I cos(P + x F))^2 / O^2 + 4 %r6^2 ) + 8 I O^2 cos(P + x F) ( sqrt(T^2 - I^2 sin(P + x F)^2 + 2 %r6 I O sin(P + x F) - %r6^2 O^2 - I cos(P + x F)) + 4 I^2 O^2 sin(P + x F)^2 - 8 %r6 I O^3 sin(P + x F) + 4 I^2 O^2 cos(P + x F)^2 ) / ( T^2 ( O^2 ( 4 ( sqrt(T^2 - I^2 sin(P + x F)^2 + 2 %r6 I O sin(P + x F) - %r6^2 O^2 - I cos(P + x F))^2 / O^2 + 4 %r6^2 ) + 8 I cos(P + x F) ( sqrt(T^2 - I^2 sin(P + x F)^2 + 2 %r6 I O sin(P + x F) - %r6^2 O^2 - I cos(P + x F)) + 4 I^2 sin(P + x F)^2 - 8 %r6 I O sin(P + x F) + 4 I^2 cos(P + x F)^2 ) ) , y = %r6 , z = - sqrt(T^2 - I^2 sin(P + x F)^2 + 2 %r6 I O sin(P + x F) - %r6^2 O^2 - I cos(P + x F)) / O ] , [ L = - sqrt( O^4 ( 4 ( sqrt(T^2 - I^2 sin(P + x F)^2 + 2 %r7 I O sin(P + x F) - %r7^2 O^2 + I cos(P + x F))^2 / O^2 + 4 %r7^2 ) - 8 I O^2 cos(P + x F) ( sqrt(T^2 - I^2 sin(P + x F)^2 + 2 %r7 I O sin(P + x F) - %r7^2 O^2 + I cos(P + x F)) + 4 I^2 O^2 sin(P + x F)^2 - 8 %r7 I O^3 sin(P + x F) + 4 I^2 O^2 cos(P + x F)^2 ) / ( T^2 ( O^2 ( 4 ( sqrt(T^2 - I^2 sin(P + x F)^2 + 2 %r7 I O sin(P + x F) - %r7^2 O^2 + I cos(P + x F))^2 / O^2 + 4 %r7^2 ) - 8 I cos(P + x F) ( sqrt(T^2 - I^2 sin(P + x F)^2 + 2 %r7 I O sin(P + x F) - %r7^2 O^2 + I cos(P + x F)) + 4 I^2 sin(P + x F)^2 - 8 %r7 I O sin(P + x F) + 4 I^2 cos(P + x F)^2 ) ) , y = %r7 , z = sqrt(T^2 - I^2 sin(P + x F)^2 + 2 %r7 I O sin(P + x F) - %r7^2 O^2 + I cos(P + x F)) / O ] , [ L = - 2 |O| ( O^2 ( 4 ( (T + I sin(P + x F))^2 / O^2 + 4 I^2 cos(P + x F)^2 / O^2 ) - 8 I sin(P + x F) (T + I sin(P + x F)) + 4 I^2 sin(P + x F)^2 - 4 I^2 cos(P + x F)^2 ) ) |T| = (T + I sin(P + x F)) / O , z = I cos(P + x F) / O ] , [ L = - 2 |O| ( O^2 ( 4 ( (T - I sin(P + x F))^2 / O^2 + 4 I^2 cos(P + x F)^2 / O^2 ) + 8 I sin(P + x F) (T - I sin(P + x F)) + 4 I^2 sin(P + x F)^2 - 4 I^2 cos(P + x F)^2 ) ) |T| = - (T - I sin(P + x F)) / O , z = I cos(P + x F) / O ] ]

--> length(extrema);
(%o26) 4

since we don't need the value of L:

--> [extrema[1][2], extrema[1][3]];
(%o27) [ y = %r6 , z = - sqrt(T^2 - I^2 sin(P + x F)^2 + 2 %r6 I O sin(P + x F) - %r6^2 O^2 - I cos(P + x F)) / O ]

--> [extrema[2][2], extrema[2][3]];
(%o28) [ y = %r7 , z = sqrt(T^2 - I^2 sin(P + x F)^2 + 2 %r7 I O sin(P + x F) - %r7^2 O^2 + I cos(P + x F)) / O ]

--> [extrema[3][2], extrema[3][3]];
(%o29) [ y = (T + I sin(P + x F)) / O , z = I cos(P + x F) / O ]

--> [extrema[4][2], extrema[4][3]];
(%o30) [ y = - (T - I sin(P + x F)) / O , z = I cos(P + x F) / O ]

those don't look bad.

(%i12) params1 : [O=2.0, I=0.4, F=20, P=%pi, T=0.3];
(%o12) [ O=2.0 , I=0.4 , F=20 , P=pi , T=0.3 ]

```

```

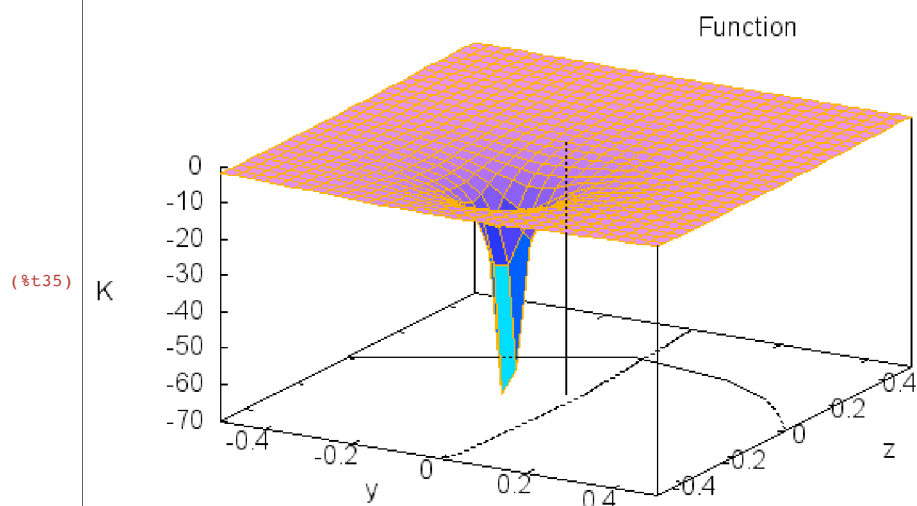
--> surf1 : ev(surface, params1);
(%o6)  $(2.0 z + 0.4 \cos(20 x))^2 + (2.0 y + 0.4 \sin(20 x))^2 - 0.09$ 

--> grad1 : ev([f_x, f_y, f_z], params1);
(%o11)  $[16.0 \cos(20 x) (2.0 y + 0.4 \sin(20 x)) - 16.0 \sin(20 x) (2.0 z + 0.4 \cos(20 x)), 4.0 (2.0 y + 0.4 \sin(20 x)), 4.0 (2.0 z + 0.4 \cos(20 x))]$ 

--> K1 : ev(K, params1);
(%o34) 
$$\frac{-128.0 (2.0 z + 0.4 \cos(20 x))^2 - 128.0 (2.0 y + 0.4 \sin(20 x))^2}{(16.0 (2.0 z + 0.4 \cos(20 x))^2 + 16.0 (2.0 y + 0.4 \sin(20 x))^2)^{3/2}}$$


--> wxplot3d(ev(K1, x=11), [y, -0.5, 0.5], [z, -0.5, 0.5],
[zlabel,"K"]);

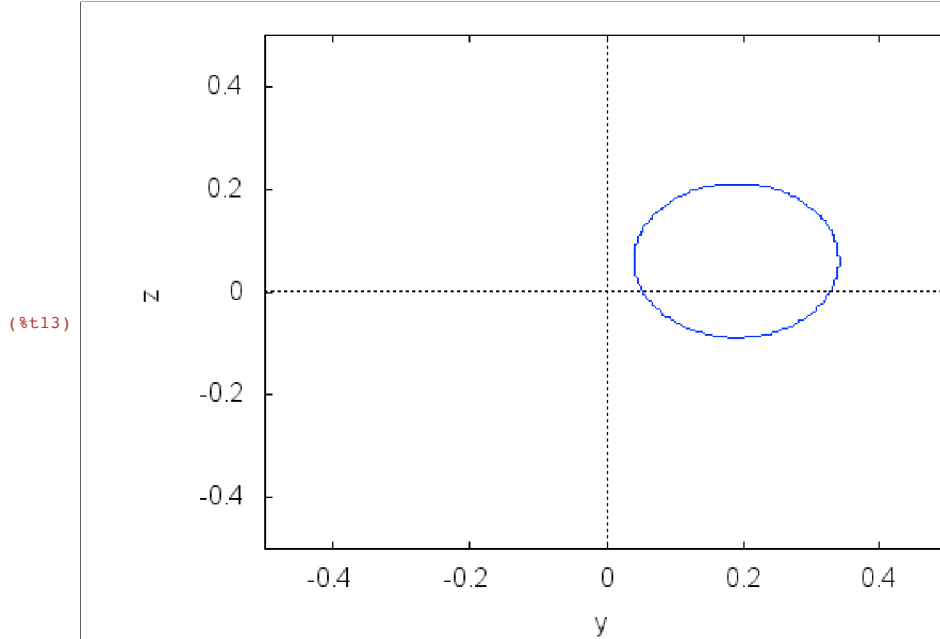
```



```

--> load(implicit_plot)$
wximplicit_plot(ev(surf1, x=0.22), [y, -0.5, 0.5], [z, -0.5, 0.5])$
rat: replaced 0.09 by 9/100 = 0.09 rat: replaced -0.3806408295558065 by -30389751/79838390 = -0.3806408295558064
rat: replaced 2.0 by 2/1 = 2.0 rat: replaced -0.1229331479913677 by -9425438/76671249 = -0.1229331479913677
rat: replaced 2.0 by 2/1 = 2.0

```



```

--> pt : [x=-1.805000, y=-0.049920, z=0.005662];
(%o38) [ x = -1.805 , y = -0.04992 , z = 0.005662 ]

```

```

--> ev(surf1, pt);
(%o39) -2.057867292955157 10^-7

```

so, looks very close to surface

```

--> ev(grad1, pt);
(%o40) [-0.135885076781867 , 1.199998628084485 , -2.772298097038217 10^-6 ]

```

so, nowhere near a singular point.

```

--> ev(K1, pt);
(%o41) -6.666674288410452

```

```

--> test1 : ev([extrema[3][2], extrema[3][3]], params1, pt[1]);
(%o42) [ y = -0.04991982851056059 , z = 0.00566234653726213 ]

```

```

--> test2 : ev([extrema[4][2], extrema[4][3]], params1, pt[1]);
(%o43) [ y = -0.3499198285105606 , z = 0.00566234653726213 ]

```

### 3.1 wtf seriously

so, take a look again at surf1...

```

--> surf1 = 0;
(%o44) (2.0 z + 0.4 cos(20 x))^2 + (2.0 y + 0.4 sin(20 x))^2 - 0.09 = 0

```

IT'S A FREAKING CIRCLE

```
--> circle_params : [y2 = y - I*sin(P+x*F)/ O, z2 = z - I*cos(P+x*F) / O, R=T/O];
(%o45) [ y2=y - \frac{I \sin(P+x F)}{O}, z2=z - \frac{I \cos(P+x F)}{O}, R=\frac{T}{O} ]
```

```
--> ev(y2^2 + z2^2 - R^2 = 0, circle_params, params1, eval);
(%o46) (z+0.2 cos(20 x))^2+(y+0.2 sin(20 x))^2-0.0225=0
```

and note that curvature is just 1/R:

```
--> ev(1/R, circle_params, params1, eval);
(%o47) 6.666666666666667
```

so the cross-section (sliced through x) is always a circle.  
the 'sharpness' seen here has to be from something else, then.

what about if I use the full 3D curvature, simply using x as the tangent vector?

### 3.2 3D curvature

```
--> H_F : hessian(surface, [x, y, z]);
(%o53)
\begin{array}{rrrr}
2 F^2 I^2 \sin(P+x F)^2+2 F^2 I \sin(P+x F)(y O-I \sin(P+x F))+2 F^2 I^2 \cos(P+x F)^2+2 F^2 I \cos(P+x F)(z O-I \cos(P+x F)) & -2 F I O \cos(P+x F) & 2 F I O \sin(P+x F) & 0 \\
-2 F I O \cos(P+x F) & 2 O^2 & 0 & 0 \\
2 F I O \sin(P+x F) & 0 & 2 O^2 & 0
\end{array}
```

```
--> gradF : ev(express(grad(surface)), diff);
(%o59) [ 2 F I (z O-I cos(P+x F)) sin(P+x F)-2 F I cos(P+x F)(y O-I sin(P+x F)), 2 O (y O-I sin(P+x F)), 2 O (z O-I cos(P+x F)) ]
```

```
--> v: [vx, vy, vz];
(%o65) [ vx, vy, vz ]
```

Finally, normal curvature in 3D (for unit tangent vector v):

```
--> Kn : transpose(v) . H_F . v / sqrt(gradF . gradF);
(%o98) ( vx ( vx
( 2 F^2 I^2 sin(P+x F)^2+2 F^2 I sin(P+x F)(y O-I sin(P+x F))+2 F^2 I^2 cos(P+x F)^2+2 F^2 I cos(P+x F)(z O-I cos(P+x F))
+2 vz F I O sin(P+x F)-2 vy F I O cos(P+x F))+vz ( 2 vx F I O sin(P+x F)+2 vz O^2)+vy ( 2 vy O^2-2 vx F I O cos(P+x F)) ) /
sqrt(( 2 F I (z O-I cos(P+x F)) sin(P+x F)-2 F I cos(P+x F)(y O-I sin(P+x F)))^2+4 O^2 (y O-I sin(P+x F))^2+4 O^2 (z O-I cos(P+x F))^2 )
```

Note that this is *not* the same as the curvature of any cross-section.  
It is the curvature of: the curve projected onto the plane containing  
the curve's tangent, and the surface normal.  
(You don't choose the normal!)

Somewhat arbitrarily, I pick as my tangent vector: the cross product  
of the surface normal (which is just grad F), and the direction the  
cross-section moves (which is itself a tangent vector).

but how do I handle the v=[vx,vy,vz] here?  
I need v declared as a vector but still something  
I can evaluate like below:

```
--> ev(Kn, vx=v_xc[1], vy=v_xc[2], vz=v_xc[3], params1, eval);
(%o111) ( \frac{128.0 (2.0 z+0.4 cos(20 x))^2}{(8.0 z+1.6 cos(20 x))^2+(-8.0 y-1.6 sin(20 x))^2} + \frac{128.0 (2.0 y+0.4 sin(20 x))^2}{(8.0 z+1.6 cos(20 x))^2+(-8.0 y-1.6 sin(20 x))^2} ) / sqrt(
(16.0 cos(20 x)(2.0 y+0.4 sin(20 x))-16.0 sin(20 x)(2.0 z+0.4 cos(20 x)))^2+16.0 (2.0 z+0.4 cos(20 x))^2+16.0 (2.0 y+0.4 sin(20 x))^2 )
```



--- ignore the below, it's earlier scratch work ---

is this the tangent I want? I'm pretty sure it's not since, you know,  
IT'S NOT AN ACTUAL TANGENT OF THE SURFACE.  
needs to be perpendicular to the normal, doofus.

```
--> Kx : f_xx / (f_x^2 + f_y^2 + f_z^2)^0.5;
(%o49) (2 F^2 I^2 sin(P+x F)^2 + 2 F^2 I sin(P+x F) (y O - I sin(P+x F)) + 2 F^2 I^2 cos(P+x F)^2 + 2 F^2 I cos(P+x F)
(z O - I cos(P+x F))) / ((2 F I (z O - I cos(P+x F)) sin(P+x F) - 2 F I cos(P+x F) (y O - I sin(P+x F)))^2 + 4 O^2
(y O - I sin(P+x F))^2 + 4 O^2 (z O - I cos(P+x F))^2) ^ 0.5
```

```
--> Kx_x : diff(Kx, x)$
      Kx_y : diff(Kx, y)$
      Kx_z : diff(Kx, z)$
```

the below hangs for some reason (possibly because my tangent vector is completely wrong):

```
extrema2 : solve([surface = 0, Kx_y = -L * f_y, Kx_z = -L * f_z], [L, y, z]);
```

### 3.3 Lipschitz constant or something

So, the gradient is unbounded in each component.  
Is this the same as the total derivative being unbounded?  
If that's the case, then the function isn't Lipschitz continuous.  
But... various other functions express the same surface. Like this:

```
--> surf2 : f3([x,y,z]) - T;
(%o5) sqrt((y O - I sin(P+x F))^2 + (z O - I cos(P+x F))^2) - T
```

```
--> ev(express(grad(surf2)), diff);
(%o9) [ ( 2 F I (z O - I cos(P+x F)) sin(P+x F) - 2 F I cos(P+x F) (y O - I sin(P+x F)) ) /
( 2 sqrt((y O - I sin(P+x F))^2 + (z O - I cos(P+x F))^2) ) ,
( y O - I sin(P+x F) ) /
( sqrt((y O - I sin(P+x F))^2 + (z O - I cos(P+x F))^2) ) ,
( z O - I cos(P+x F) ) /
( sqrt((y O - I sin(P+x F))^2 + (z O - I cos(P+x F))^2) ) ]
```

This has order 1 on top, order 1 on bottom, so this should balance out.  
If I understand right what Wikipedia is saying with,  
"Moreover, if K is the best Lipschitz constant of f, then |Df(x)| ≤ K  
whenever the total derivative Df exists."  
then...

```
--> grad2 : ev(express(grad(surf2)), diff)$
```

```
--> Df : trigsimp(sqrt(grad2 . grad2));
(%o22) sqrt(-( (z^2 - y^2) F^2 I^2 O^2 sin(P+x F)^2 + (-2 y z F^2 I^2 O^2 cos(P+x F) - 2 y I O^3) sin(P+x F) - 2 z I O^3 cos(P+x F) +
(z^2 + y^2) O^4 + (y^2 F^2 + 1) I^2 O^2 ) / (2 y I O sin(P+x F) + 2 z I O cos(P+x F) + (-z^2 - y^2) O^2 - I^2 ) )
```

```
--> ev(Df, x=-P/F);
(%o32) sqrt(-((z^2 + y^2) O^4 - 2 z I O^3 + (y^2 F^2 + 1) I^2 O^2) / (-z^2 - y^2) O^2 + 2 z I O - I^2)
```

```
--> trigsimp(ev(Df, x=-P/F + %pi / F / 2));
(%o33) sqrt((z^2 + y^2) O^4 - 2 y I O^3 + (z^2 F^2 + 1) I^2 O^2) / ((z^2 + y^2) O^2 - 2 y I O + I^2)
```

```
--> trigsimp(ev(Df, x=-P/F + %pi / F));
(%o35) sqrt((z^2 + y^2) O^4 + 2 z I O^3 + (y^2 F^2 + 1) I^2 O^2) / ((z^2 + y^2) O^2 + 2 z I O + I^2)
```

```
--> trigsimp(ev(Df, x=-P/F + 3 * %pi / F / 2));
```

$$(\%036) \sqrt{\frac{(z^2+y^2)O^4+2yIO^3+(z^2F^2+1)I^2O^2}{(z^2+y^2)O^2+2yIO+I^2}}$$

this is guesswork under the assumption that max/min occur where sin/cos hit their max/min

```
--> Df2 : subst([sin(P+x*F)=S, cos(P+x*F)=C], Df);
```

$$(\%058) \sqrt{\frac{(z^2-y^2)F^2I^2O^2S^2+(-2yIO^3-2yzCF^2I^2O^2)S+(z^2+y^2)O^4-2zCIO^3+(y^2F^2+1)I^2O^2}{2yIOS+(-z^2-y^2)O^2+2zCIO-I^2}}$$

So I \*think\* that regardless the additive terms in the numerator -  $(z^2-y^2)*F^2*I^2*O^2$ ,  $2*y*z*F^2*I^2*O^2$ ,  $(z^2+y^2)*O^4$ ,  $(y^2*F^2+1)*I^2*O^2$ , and - will dominate due to the powers on y and z.

Denominator is dominated by  $(-z^2-y^2)*O^2$ .

$(z^2-y^2)*F^2*I^2*O^2$  becomes (for large z or y)  $I^2 * F^2$ .

$2*y*z*F^2*I^2*O^2$  vanishes? I think?

$(z^2+y^2)*O^4$  becomes (for large z or y)  $O^2$ .

$(y^2*F^2+1)*I^2*O^2$  becomes (for large y)  $I^2 * F^2$ .

Then, if I'm right, the overall bound is something like:  $|O|+|I*F|$

This seems sensible, I think.

Intuitively: higher (spatial) frequencies correspond to steeper change.

```
--> ev(Df2, [z=0, y=0]);
```

$$(\%060) |O|$$

```
--> ev(Df2, [z=1, y=0]);
```

$$(\%061) \sqrt{-\frac{F^2I^2O^2S^2+O^4-2CIO^3+I^2O^2}{-O^2+2CIO-I^2}}$$

```
--> ev(Df2, [z=0, y=1]);
```

$$(\%062) \sqrt{-\frac{-F^2I^2O^2S^2-2IO^3S+O^4+(F^2+1)I^2O^2}{2IOS-O^2-I^2}}$$

### □ 3.4 More Lipschitz derivation?

□ The below is a lower-bound on K:

```
--> K : ((f3([x1,y1,z1]) - T) - (f3([x2,y2,z2]) - T)) / sqrt([x1,y1,z1] . [x2,y2,z2]);
```

$$(\%012) \frac{\sqrt{(y1O-I\sin(P+x1F))^2+(z1O-I\cos(P+x1F))^2}-\sqrt{(y2O-I\sin(P+x2F))^2+(z2O-I\cos(P+x2F))^2}}{\sqrt{z1z2+y1y2+x1x2}}$$

```
--> trigsimp(K);
```

$$(\%013) -(\sqrt{-2y2IO\sin(P+x2F)-2z2IO\cos(P+x2F)+(z2^2+y2^2)O^2+I^2}-\sqrt{-2y1IO\sin(P+x1F)-2z1IO\cos(P+x1F)+(z1^2+y1^2)O^2+I^2})/\sqrt{z1z2+y1y2+x1x2}$$

□ So is this (it's just squaring numerator & denominator) but I feel like I just made it worse:

```
--> K2 : ((f3([x1,y1,z1]) - T) - (f3([x2,y2,z2]) - T))^2 / ([x1,y1,z1] . [x2,y2,z2]);
```

$$(\%014) \frac{\left(\sqrt{(y1O-I\sin(P+x1F))^2+(z1O-I\cos(P+x1F))^2}-\sqrt{(y2O-I\sin(P+x2F))^2+(z2O-I\cos(P+x2F))^2}\right)^2}{z1z2+y1y2+x1x2}$$

```
--> trigsimp(K2);
```

$$(\%015) -\left(2\sqrt{-2y1IO\sin(P+x1F)-2z1IO\cos(P+x1F)+(z1^2+y1^2)O^2+I^2}-\sqrt{-2y2IO\sin(P+x2F)-2z2IO\cos(P+x2F)+(z2^2+y2^2)O^2+I^2}+2y2IO\sin(P+x2F)+2z2IO\cos(P+x2F)+2y1IO\sin(P+x1F)+2z1IO\cos(P+x1F)+(-z2^2-z1^2-y2^2-y1^2)O^2-2I^2\right)/(z1z2+y1y2+x1x2)$$

⌈ If one considers that cos/sin are both bounded to [-1,1]...

⌈ --> K\_simp : subst([sin(P+x1\*F)=S1, cos(P+x1\*F)=C1, sin(P+x2\*F)=S2, cos(P+x2\*F)=C2], trigsimp(K));  
 (%o22) 
$$-\frac{\sqrt{-2 y^2 I O S^2 + (z^2 + y^2)^2 O^2 - 2 z^2 C^2 I O + I^2} - \sqrt{-2 y^1 I O S^1 + (z^1 + y^1)^2 O^2 - 2 z^1 C^1 I O + I^2}}{\sqrt{z^1 z^2 + y^1 y^2 + x^1 x^2}}$$

⌈ Also, the trig terms are always paired with a lower-order y or z (they aren't squared, but are under a radical) than those of the denominator.

⌈ --> trigsimp(ev(K, [y1=0, y2=0, z1=0, z2=0]));  
 (%o29) 0

⌈ so, this is at a minimum for those...  
 but suppose we let (for instance) z1 or z2 grow without bound.

⌈ --> trigsimp(ev(K, [y1=0, y2=0, z1=0, z2=inf]));  
 (%o34) 
$$-\frac{\sqrt{-2 \infty I O \cos(P+x^2 F) + \infty^2 O^2 + I^2} - |I|}{\sqrt{x^1 x^2}}$$

⌈ --> trigsimp(ev(K, [y1=0, y2=0, z1=inf, z2=0]));  
 (%o35) 
$$\frac{\sqrt{-2 \infty I O \cos(P+x^1 F) + \infty^2 O^2 + I^2} - |I|}{\sqrt{x^1 x^2}}$$

⌈ then K grows without bound (nothing in the numerator counteracts it, even if the other z is nonzero).

⌈ --> trigsimp(ev(K, [y1=0, y2=0, z1=1, z2=inf]));  
 (%o36) 
$$-\frac{\sqrt{-2 \infty I O \cos(P+x^2 F) + \infty^2 O^2 + I^2} - \sqrt{-2 I O \cos(P+x^1 F) + O^2 + I^2}}{\sqrt{x^1 x^2 + \infty}}$$

⌈ but... as long as f<sup>-1</sup>(x) remains the same, this is the same surface.  
 I can just take it to a lower power.

⌈ --> f4(p) := ((p[2] \* O - I \* sin(p[1] \* F + P))^2 + (p[3] \* O - I \* cos(p[1] \* F + P))^2)^(1/4);  
 (%o38) f4(p) := ((p2 O - I sin(p1 F + P))^2 + (p3 O - I cos(p1 F + P))^2)^(1/4)

⌈ --> K4 : ((f4([x1,y1,z1]) - T) - (f4([x2,y2,z2]) - T)) / sqrt([x1,y1,z1] . [x2,y2,z2]);  
 (%o40) 
$$\frac{\left((y^1 O - I \sin(P+x^1 F))^2 + (z^1 O - I \cos(P+x^1 F))^2\right)^{1/4} - \left((y^2 O - I \sin(P+x^2 F))^2 + (z^2 O - I \cos(P+x^2 F))^2\right)^{1/4}}{\sqrt{z^1 z^2 + y^1 y^2 + x^1 x^2}}$$

⌈ --> trigsimp(ev(K4, [y1=0, y2=0, z1=inf, z2=0]));  
 (%o42) 
$$\frac{(-2 \infty I O \cos(P+x^1 F) + \infty^2 O^2 + I^2)^{1/4} - \sqrt{|I|}}{\sqrt{x^1 x^2}}$$

⌈ oh, I guess that doesn't really do much since the infinity is still there in the numerator.....

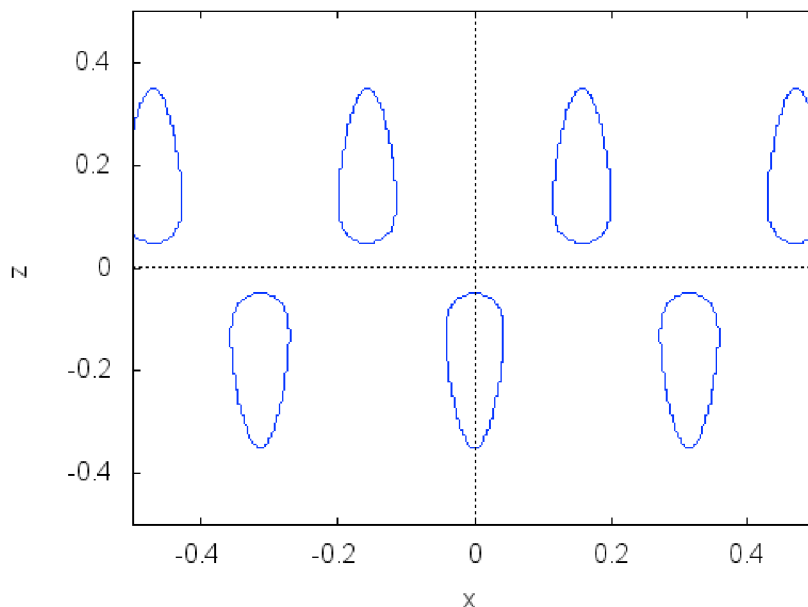
so it is \*not\* Lipschitz continuous, even if its isosurface is continuous. It is probably locally Lipschitz continuous but I don't know that that gets me much further than simply ray-marching.

### □ 3.5 Other cross sections

⌈ This is for y = 0.  
 Imagine spinning the entire graph around the x axis, and as you do, moving it to the right (positive x) so that a half-turn makes each 'bottom' lobe line up with the next 'top' lobe to the right, and a full turn makes the 'bottom' lobe line up with the next bottom one (and the top one with the next top one).

--> wximplicit\_plot(ev(surf1, y=0), [x, -0.5, 0.5], [z, -0.5, 0.5])\$  
 rat: replaced 0.09 by 9/100 = 0.09 rat: replaced -0.16 by -4/25 = -0.16 rat: replaced 0.4 by 2/5 = 0.4  
 rat: replaced 2.0 by 2/1 = 2.0

(%t46)



I could probably have inverted this in Maxima, but instead did it by hand:

(%i13) x\_inv : acos((T^2-I^2-(O\*z)^2)/(2\*O\*z\*I))/F - P/F;

(%o13) 
$$\frac{\arccos\left(\frac{T^2 - z^2 O^2 - I^2}{2 z I O}\right)}{F} - \frac{P}{F}$$

arccos is periodic with integer multiples of 2\*pi, so I added that below:

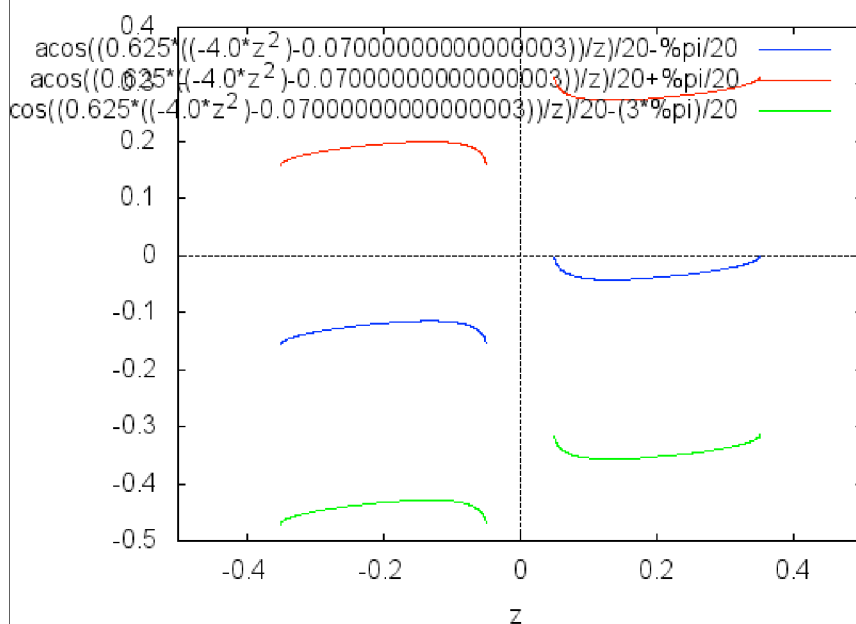
--> wxplot2d([ev(x\_inv, params1), ev(x\_inv + 2\*pi/F, params1), ev(x\_inv - 2\*pi/F, params1)], [z, -0.5, 0.5])\$

plot2d: expression evaluates to non-numeric value somewhere in plotting range.

plot2d: expression evaluates to non-numeric value somewhere in plotting range.

plot2d: expression evaluates to non-numeric value somewhere in plotting range.

(%t43)

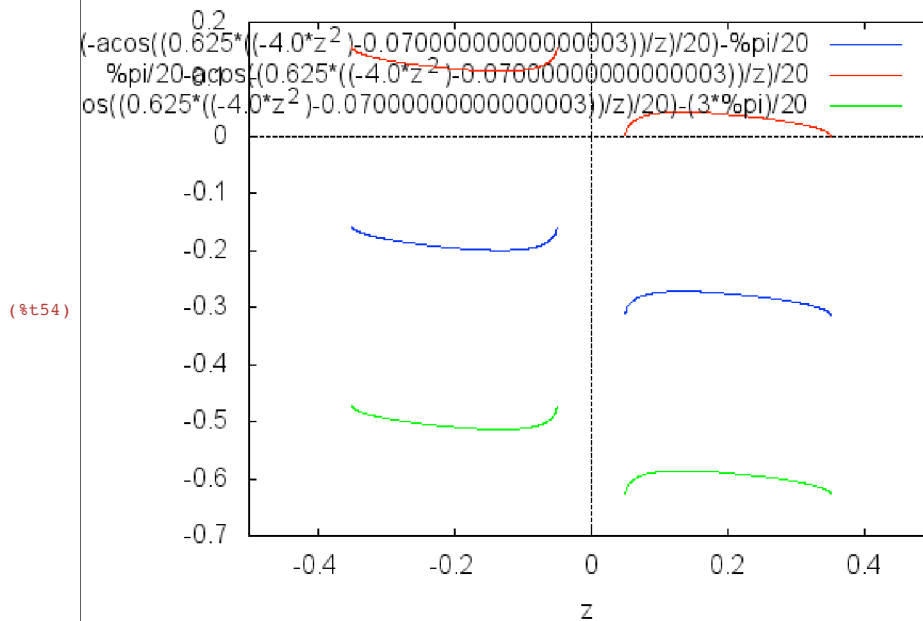


This looks right except that I'm losing an entire half of each lobe and I can't reason through why.  
 If  $\cos(x)=y$ , then  $x=\arccos(y)+2\pi n$ .  
 But: if  $\cos(x)=y$  then  $\cos(-x)=y$ , so  $y=-\arccos(x)+2\pi n$ .  
 So, below I have negated F and P (which is equivalent to negating  $(F*x+P)$ , the argument to  $\arccos$ :

```
--> x_inv2 : ev(x_inv, F=-F, P=-P);
```

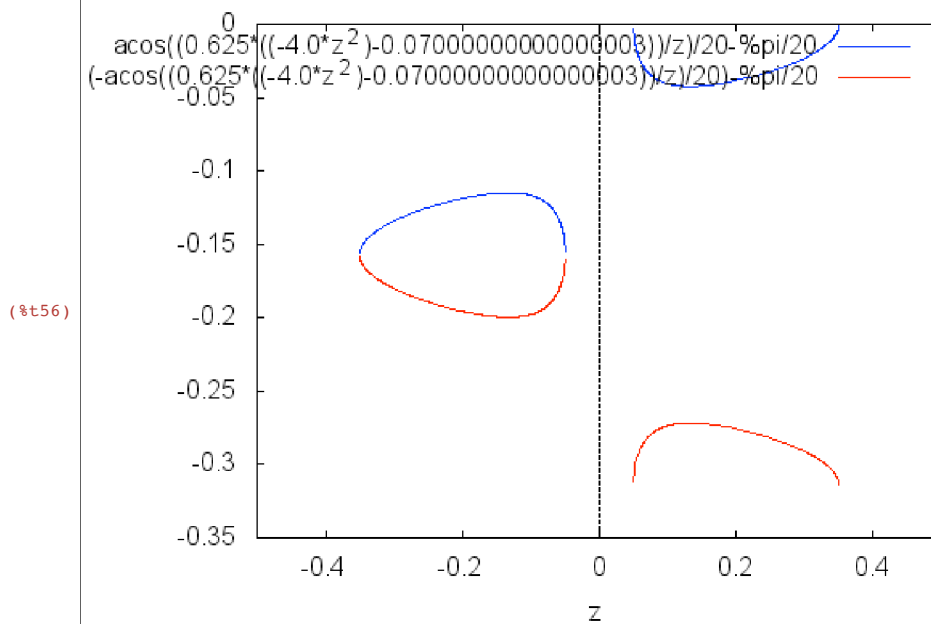
$$(\%o53) \quad -\frac{\arccos\left(\frac{T^2 - z^2 O^2 - I^2}{2 z I O}\right)}{F} - \frac{P}{F}$$

```
--> wxplot2d([ev(x_inv2, params1), ev(x_inv2 + 2*pi/F, params1), ev(x_inv2 - 2*pi/F, params1)], [z, -0.5, 0.5])$
plot2d: expression evaluates to non-numeric value somewhere in plotting range.
plot2d: expression evaluates to non-numeric value somewhere in plotting range.
plot2d: expression evaluates to non-numeric value somewhere in plotting range.
```



so there we go:  
 (how can I make wxplot2d get rid of the annoying labels?)

```
--> wxplot2d([ev(x_inv, params1), ev(x_inv2, params1)], [z, -0.5, 0.5])$
plot2d: expression evaluates to non-numeric value somewhere in plotting range.
plot2d: expression evaluates to non-numeric value somewhere in plotting range.
```



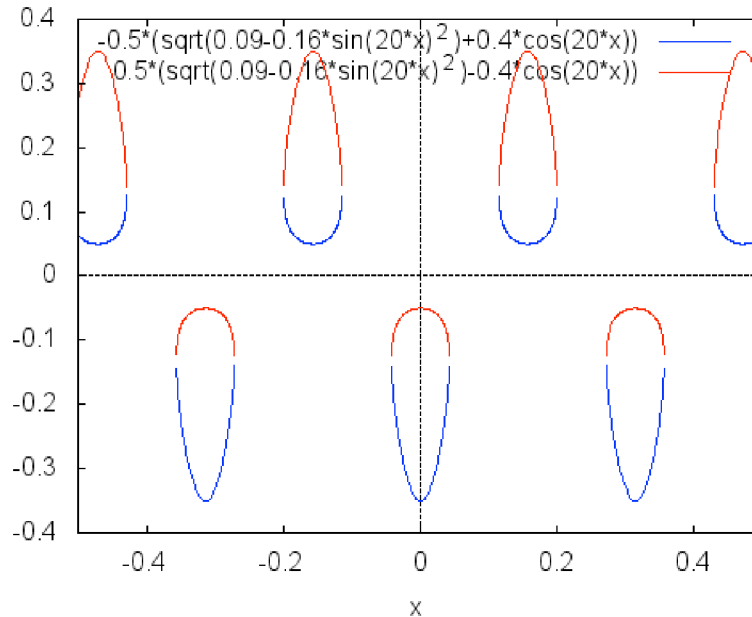
```
(%i15) z_inv : solve(ev(surface, y=0), z);
(%o15) [ z = - \sqrt{\frac{T^2 - I^2 \sin(P + x F)^2 - I \cos(P + x F)}{0}}, z = \sqrt{\frac{T^2 - I^2 \sin(P + x F)^2 + I \cos(P + x F)}{0}} ]
```

```
oh. duh - it was a quadratic, and I ignored it the first time because
I saw that it wouldn't factor easily.
well, regardless, it has to be piecewise even for just one period worth,
like the other.
```

```
--> z_inv_ev : [rhs(ev(z_inv, params1)[1]), rhs(ev(z_inv, params1)[2])];
(%o14) [ -0.5 (\sqrt{0.09 - 0.16 \sin(20 x)^2 + 0.4 \cos(20 x)}), 0.5 (\sqrt{0.09 - 0.16 \sin(20 x)^2 - 0.4 \cos(20 x)}) ]
```

```
--> wxplot2d(z_inv_ev, [x, -0.5, 0.5])$
plot2d: expression evaluates to non-numeric value somewhere in plotting range.
plot2d: expression evaluates to non-numeric value somewhere in plotting range.
```

(%t63)



For the sake of my explicit representation to generate a mesh, I need to guarantee that a vertex is directly at the valley at the bottom, and somewhat also at the peak on top.

```
--> solve(rhs(diff(z_inv[1], x)) = 0, x);
solve: using arc-trig functions to get a solution. Some solutions will be lost.
(%o69) [ x = -P/F, cos(P+x F) = sqrt(T^2 - I^2 sin(P+x F)^2) / I ]
```

```
--> solve(rhs(diff(z_inv[2], x)) = 0, x);
solve: using arc-trig functions to get a solution. Some solutions will be lost.
(%o85) [ x = -P/F, cos(P+x F) = -sqrt(T^2 - I^2 sin(P+x F)^2) / I ]
```

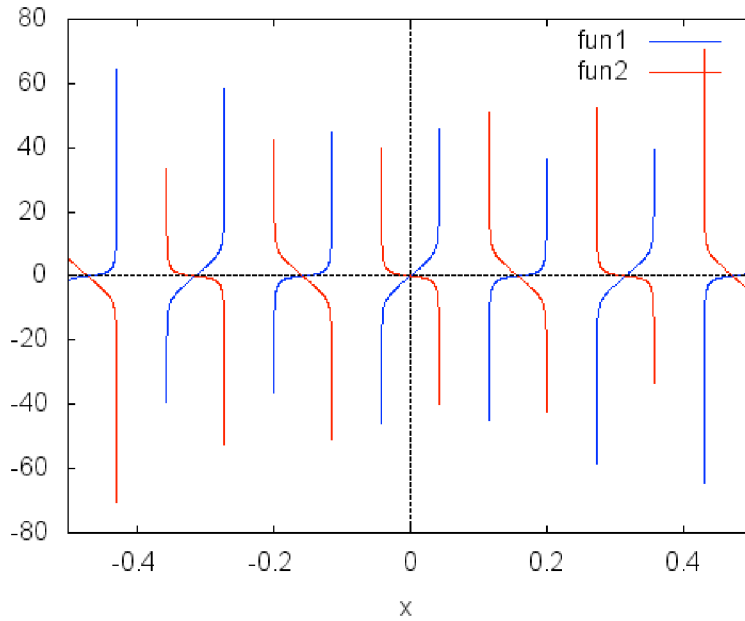
this first solution checks out: if phase  $P=0$ ,  $x=0$  is a peak of both.

I may have one additional issue: the vertex sampling needs to be sensible, as if I should be incrementing  $x$  \*inversely\* proportionally with the derivative of  $f(x)$ . At first glance, my vertices should be placed at (approximately) equal arc lengths along the function itself, but this also isn't fully right, because I also want portions with higher curvature to have more vertices devoted to them.

Derivative:

```
--> wxplot2d(diff(z_inv_ev, x), [x, -0.5, 0.5])$
plot2d: expression evaluates to non-numeric value somewhere in plotting range.
plot2d: expression evaluates to non-numeric value somewhere in plotting range.
```

(%t86)



Curvature:

```
(%i16) Kz : diff(z_inv, x, 2) / (1 + diff(z_inv, x)^2)^(3/2);
```

$$\begin{aligned}
 (%o16) \quad & 0 = - \frac{\frac{F^2 I^2 \sin(P+x F)^2}{\sqrt{T^2 - I^2 \sin(P+x F)^2}} - \frac{F^2 I^2 \cos(P+x F)^2}{\sqrt{T^2 - I^2 \sin(P+x F)^2}} - \frac{F^2 I^4 \cos(P+x F)^2 \sin(P+x F)^2}{(T^2 - I^2 \sin(P+x F)^2)^{3/2}} + F^2 I \cos(P+x F)}{\left( \frac{\left( F I \sin(P+x F) - \frac{F I^2 \cos(P+x F) \sin(P+x F)}{\sqrt{T^2 - I^2 \sin(P+x F)^2}} \right)^2}{O^2} + 1 \right)^{3/2}}, \quad 0 = \\
 & \frac{\frac{F^2 I^2 \sin(P+x F)^2}{\sqrt{T^2 - I^2 \sin(P+x F)^2}} - \frac{F^2 I^2 \cos(P+x F)^2}{\sqrt{T^2 - I^2 \sin(P+x F)^2}} - \frac{F^2 I^4 \cos(P+x F)^2 \sin(P+x F)^2}{(T^2 - I^2 \sin(P+x F)^2)^{3/2}} - F^2 I \cos(P+x F)}{\left( \frac{\left( -\frac{F I^2 \cos(P+x F) \sin(P+x F)}{\sqrt{T^2 - I^2 \sin(P+x F)^2}} - F I \sin(P+x F) \right)^2}{O^2} + 1 \right)^{3/2}} ]
 \end{aligned}$$

```
(%i17) trigsimp(rhs(Kz[1]));
```

$$\begin{aligned}
 (%o17) \quad & - \left( \sqrt{T^2 - I^2 \sin(P+x F)^2} \left( F^2 I \cos(P+x F) T^2 - F^2 I^3 \cos(P+x F) \sin(P+x F)^2 \right) + \left( 2 F^2 I^2 \sin(P+x F)^2 - F^2 I^2 \right) \frac{T^2 - F^2 I^4 \sin(P+x F)^4}{\left( \sqrt{T^2 - I^2 \sin(P+x F)^2} \left( O T^2 - I^2 O \sin(P+x F)^2 \right) \right)} \left( - \left( 2 F^2 I^3 \cos(P+x F) \sin(P+x F)^2 \sqrt{T^2 - I^2 \sin(P+x F)^2} \right. \right. \right. \\
 & \left. \left. + \left( -F^2 I^2 \sin(P+x F)^2 - O^2 \right) T^2 + 2 F^2 I^4 \sin(P+x F)^4 + \left( I^2 O^2 - F^2 I^4 \right) \sin(P+x F)^2 \right) / \left( O^2 T^2 - I^2 O^2 \sin(P+x F)^2 \right) \right)^{3/2} )
 \end{aligned}$$

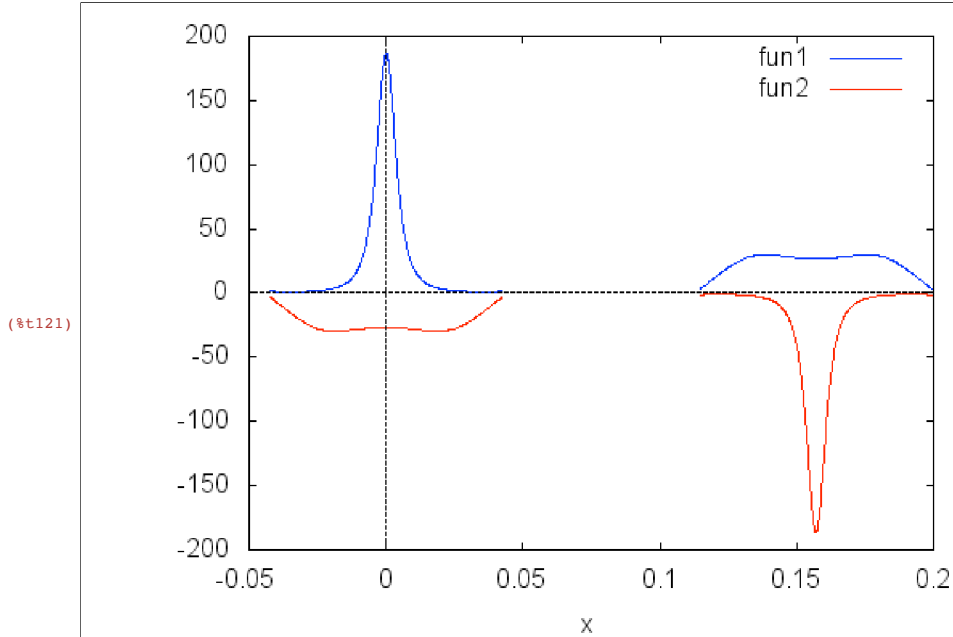
```
(%i18) trigsimp(rhs(Kz[2]));
```

$$\begin{aligned}
 (%o18) \quad & - \left( \sqrt{T^2 - I^2 \sin(P+x F)^2} \left( F^2 I \cos(P+x F) T^2 - F^2 I^3 \cos(P+x F) \sin(P+x F)^2 \right) + \left( F^2 I^2 - 2 F^2 I^2 \sin(P+x F)^2 \right) \frac{T^2 + F^2 I^4 \sin(P+x F)^4}{\left( \sqrt{T^2 - I^2 \sin(P+x F)^2} \left( O T^2 - I^2 O \sin(P+x F)^2 \right) \right)} \left( \left( 2 F^2 I^3 \cos(P+x F) \sin(P+x F)^2 \sqrt{T^2 - I^2 \sin(P+x F)^2} \right. \right. \right. \\
 & \left. \left. + \left( F^2 I^2 \sin(P+x F)^2 + O^2 \right) T^2 - 2 F^2 I^4 \sin(P+x F)^4 + \left( F^2 I^4 - I^2 O^2 \right) \sin(P+x F)^2 \right) / \left( O^2 T^2 - I^2 O^2 \sin(P+x F)^2 \right) \right)^{3/2} )
 \end{aligned}$$



```
(%i23) optimize(Kz);
(%o23) block([%1,%2,%3,%4,%5,%6,%7,%8,%9,%10,%11,%12,%13,%14,%15],%1:= $\frac{1}{0}$ ,%2:= $F^2$ ,%3:= $P+x F$ ,%4:= $\cos(\%3)$ ,
%5:= $\%4^2$ ,%6:= $\sin(\%3)$ ,%7:= $\%6^2$ ,%8:= $I^2$ ,%9:= $\sqrt{T^2-\%8 \%7}$ ,%10:= $-\frac{\%2 I^4 \%5 \%7}{\%9^3}$ ,%11:= $\frac{1}{\%9}$ ,%12:= $-\%2 \%8 \%5 \%11$ ,%13:= $\%2 \%8 \%7 \%11$ ,
%14:= $\frac{1}{0^2}$ ,%15:= $-F \%8 \%4 \%6 \%11$ ,[ $0=-\frac{\%1(\%13+\%12+\%10+\%2 I \%4)}{\sqrt{\%14(\%15+F I \%6)^2+1}^3}$ , $0=\frac{\%1(\%13+\%12+\%10-\%2 I \%4)}{\sqrt{\%14(\%15-F I \%6)^2+1}^3}$ ])
```

```
--> Kz : diff(z_inv_ev, x, 2) / (1 + diff(z_inv_ev, x)^2)^(3/2)$
wxplot2d(Kz, [x, -0.05, 0.2])$
plot2d: expression evaluates to non-numeric value somewhere in plotting range.
plot2d: expression evaluates to non-numeric value somewhere in plotting range.
```

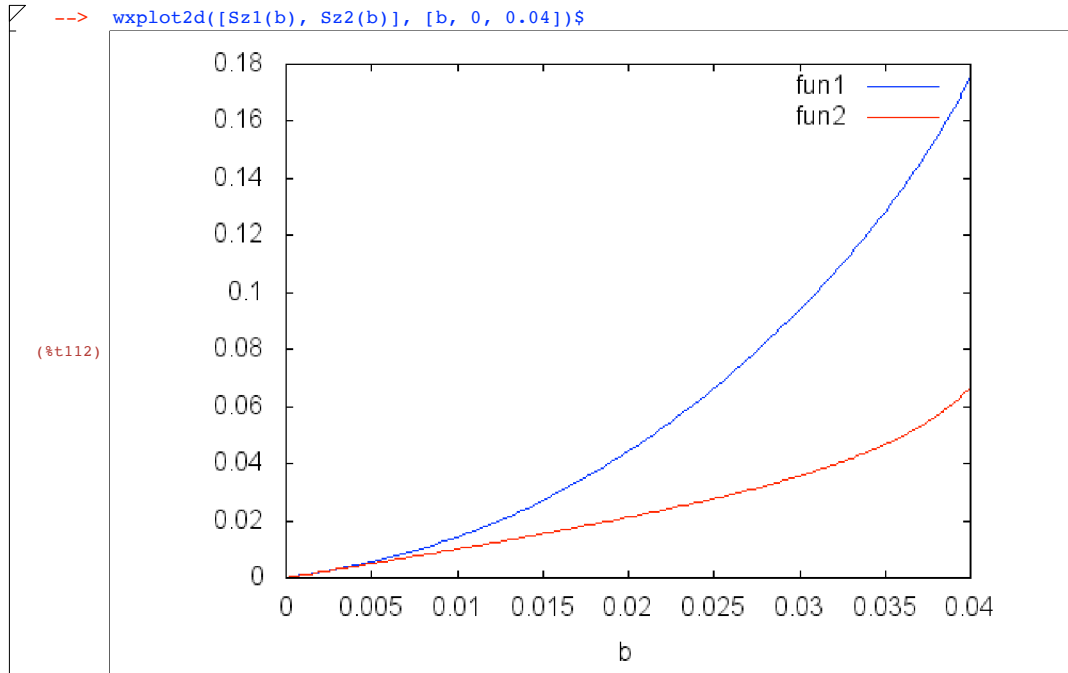


```
--> ev(Kz, x=0);
(%o113) [ 186.6666666666667, -26.6666666666667 ]
```

Arc length:

```
--> Sz1(b) := quad_qag(sqrt(1+diff(z_inv_ev[1],x)^2), x, 0, b, 3)[1];
(%o110) Sz1(b):=(quad_qag( $\sqrt{1+\text{diff}(z\_inv\_ev_1, x)^2}$ ), x, 0, b, 3))1
```

```
--> Sz2(b) := quad_qag(sqrt(1+diff(z_inv_ev[2],x)^2), x, 0, b, 3)[1];
(%o111) Sz2(b):=(quad_qag( $\sqrt{1+\text{diff}(z\_inv\_ev_2, x)^2}$ ), x, 0, b, 3))1
```



☑ I guess that's nothing particularly interesting.

#### ☐ **4 where I'm stuck and/or what I learned**

- ☑ - Cross section curvature was completely the wrong way to go here
  - 3D curvature has its own complexities
  - I can't see how to solve for that d vector (direction of the cross section) without finding  $dy/dx$  and  $dz/dx$  of  $F$ , which is tricky when it's an implicit function, though this may still be possible
  - I already know that while the edges are sharp, they are neither 1D features nor singular points/curves and it may not even have the desired effect to make CGAL preserve them
  - I am pretty sure I can find an analytical answer without requiring curvature
- 
- ☑ - Can't I find a way to make CGAL subdivide more in areas of higher curvature, or something like that? Could I subdivide it myself and then do gradient-descent?