Bowlplot PhL 2.6.11

Early efforts at plotting the bowl potential. The final plots below are I think the selection of plots that is shown in bowl doc. Probably this is where I learned how to do these plots for the first time.

Overview: I was having the usual Maple problems as reviewed here, but eventually got the plots I was after. Gave up on implicit plot, the regular 3D plot is much better. See bowlplot.mws for code.

Now that I have a "theoretical" evaluated form for the potential, namely

$$V(\xi,u)= (V_0/\pi)\sqrt{ch\xi - cosu}\ * \\ \{ (1/\sqrt{ch\xi - cosu})\ \cot^{-1}[(-1)^{\eta 1}|\sqrt{1+cosu}|\ /\sqrt{ch\xi - cosu}] \\ +\ (1/\sqrt{ch\xi - cos(2u_0-u)})\cot^{-1}[(-1)^{\eta 2}|\sqrt{1+cos(2u_0-u)}|/\sqrt{ch\xi - cos(2u_0-u)}]\ \} \qquad (2.1)$$

where $\eta_1 = floor[(3\pi-u)/2\pi]$ u_0 in $(0,\pi)$

$\eta_2 = floor[(2u_0-u+\pi)/2\pi]$ u in $(u_0, u_0+2\pi)$

I thought it would be nice to see the potential contours relative to the bowl. I thought this would be easy to do, but it has turned out to have various problems, hence we now start this new doc to track these problems (aka "issues").

1. Finding ξ. In the implicitplot, we are going to scan ρ and z, and compute u and ξ, and thus compute V, and check to see it V = 0.95 or whatever to get a contour plot. It is easy to find ξ

$$\xi = \tanh^{-1}[2a\rho/(a^2+\rho^2+z^2)]$$

This is never singular, and never ambiguous. We only use $\rho \geq 0$.

2. Finding u. Finding u is much harder! We start with these equations

$$\rho = ash\xi/(ch\xi - cosu)$$
$$z = asinu/(ch\xi - cosu)$$

We can solve the first one for cosu as follows

$$(ch\xi - cosu)\rho = a\,sh\xi \quad \Rightarrow \quad ch\xi - cosu = (a/\rho)\,sh\xi \quad \Rightarrow$$

$$cosu = ch\xi - (a/\rho)sh\xi$$

However, when ρ = 0, this does not work so we have to take a limit. From above we get

$$\xi \approx \tanh^{-1}\ [2a\rho/(a^2+z^2)]\ \approx\ 2a\rho/(a^2+z^2)\ \approx sh\xi \qquad ch\xi\ \approx 1 \text{ through first order}$$

Then we have

$$\text{cosu} = \text{ch}\xi - (a/\rho)\text{sh}\xi = 1 - (a/\rho)[\, 2a\rho/(a^2+z^2)\,] = 1 - a[\, 2a/(a^2+z^2)\,]$$

$$= 1 - 2a^2/(a^2+z^2) = (a^2+ z^2 - 2a^2)/(a^2+z^2) = (z^2- a^2)/(z^2+a^2)$$

So our algorithm to compute cosu is this:

if $\rho=0$ then $\text{cosu} = (z^2- a^2)/(z^2+a^2)$ else $\text{cosu} = \text{ch}\xi - (a/\rho)\text{sh}\xi$;

Now what about sinu ? We know that

$$\rho/z = \text{sh}\xi/\text{sinu} \Rightarrow \qquad \text{sinu} = (z/\rho)\,\text{sh}\xi$$

We have the same problem here if $\rho = 0$ and again we use $2a\rho/(a^2+z^2) \approx \text{sh}\xi$ so that

$$\text{sinu} = (z/\rho)\,\text{sh}\xi \approx z\, 2a/(a^2+z^2) = 2az/(a^2+z^2)$$

So our algorithm to compute sinu is this

if $\rho=0$ then $\text{sinu} = 2az/(a^2+z^2)$ else $\text{sinu} = (z/\rho)\,\text{sh}\xi$;

So we can summarize these two results

if $\rho=0$ then $\text{cosu} = (z^2- a^2)/(z^2+a^2)$ else $\text{cosu} = \text{ch}\xi - (a/\rho)\text{sh}\xi$;
if $\rho=0$ then $\text{sinu} = 2az/(a^2+z^2)$ else $\text{sinu} = (z/\rho)\,\text{sh}\xi$;

Once we have cosu and sinu, we can compute u in the range $(0,2\pi)$ in this way

u = arctan2Pi(x,y) = arctan2Pi(cosu,sinu)

This function arctan2Pi is explained in doc trigonometry/arc tangent.doc which I wrote yesterday.

We then want to adjust the range to get u in u_0, $u_0+2\pi$ so we would then add

of $u < u_0$ then $u = u+2\pi$;

Note this is much different from saying $u = u+u_0$.

3. Maple routine to compute u

```
> getu := proc(rho,z)
    global a,u0;
    local xi,cosu,sinu,t1;
    if type(rho,numeric) and type(z,numeric) then
      xi := arctanh(2*a*rho/(a^2+rho^2+z^2));
      if rho = 0 then
        cosu := (z^2- a^2)/ (z^2+a^2);
```

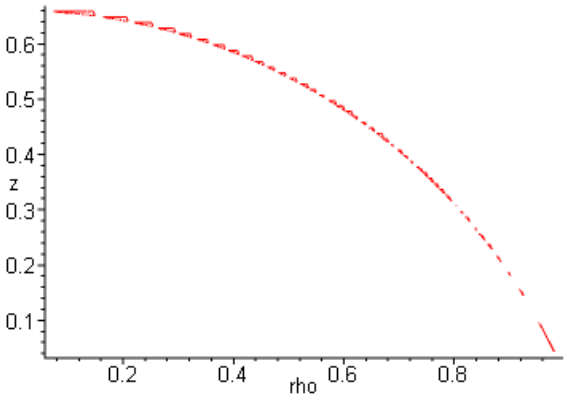
```
        sinu :=  2*a*z/(z^2+a^2);
      else
        cosu := cosh(xi)-(a/rho)*sinh(xi);
        sinu := (z/rho)*sinh(xi);
      fi;
      t1 := arctan2Pi(cosu,sinu);
      if t1 < u0 then t1 := t1 + 2*Pi fi;    # get into proper range
      RETURN(t1);
    else
      'getu(rho,z)';
    fi;
end:
```

4. Technical problem: If we try to plot u = u_0, the plot disappears.

We have this Maple code active

```
> a := 1;
> u0 := 5*Pi/8;
> xi := arctanh(2*a*abs(rho)/(a^2+rho^2+z^2));
> u := getu(rho,z);
with(plots):implicitplot(u=u0+.01, rho = 0..1, z = -1..1, grid = [100,100],
scaling = CONSTRAINED);
```

```
with(plots):implicitplot(u=u0+.01, rho = 0..1, z = -1..1, grid =
[200,200], scaling = CONSTRAINED);
```



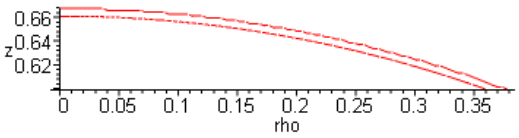
You can see that the curve is ragged even with fairly high resolution on the scan. When we remove the .01 adder, we get no curve at all. That is our problem for this section. I think implicitplot sees groups of points as "little islands" and we are getting a plot of this archipelago instead of a curve.

This makes you wonder how this program works. It probably accumulates just a set of 2D points, then with some ε it connects them with lines as best it can.

Increasing Digits to 11,12,13 makes no visible difference.

If you zoom in on a region, you find there are really two curves, as I expect , for example

```
with(plots):implicitplot(u=u0+.01, rho = 0..1, z = 0.6..1,grid =
[250,250], scaling = CONSTRAINED);
```



so that is part of the raggedness problem I think. But even zoomed in, if we get rid of the .01, we get no plot at all. This is the question of the moment: why is that?

Here is an idea. Look at the getu routine above. Suppose it is just the nature of the calculation that when we compute u, we get $u = u_0 - .000000001$. Since we have $u<u_0$, this gets kicked up by 2π and that is why we get no plot. But I put ± .0001 and neither fixed the problem.

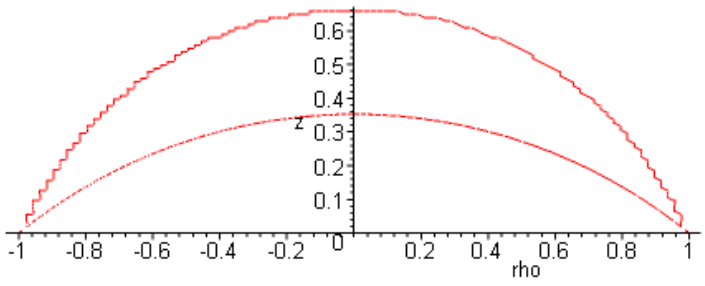
The next idea is that during the scan, we just skip over (ρ,z) pairs which are on the curve. The subject here would be to look at $\partial_\rho u$ and $\partial_z u$ and see if these are very large for some reason. I poked around on this and nothing violent seems to happen. Denominators do not go small for example.

I could define a tiny window and "track" the computation during the scan within that window, just to learn what is happening. Logic analyzer.

Well, this led to the realization that Maple cannot do Booleans involving constants like Pi unless they are evaluated.

Now here is another plot attempt, plotting not V but just u:

```
with(plots):implicitplot(u=u0+.5, rho = -2..2, z = -1..1,grid = [200,200],
scaling = CONSTRAINED);
```

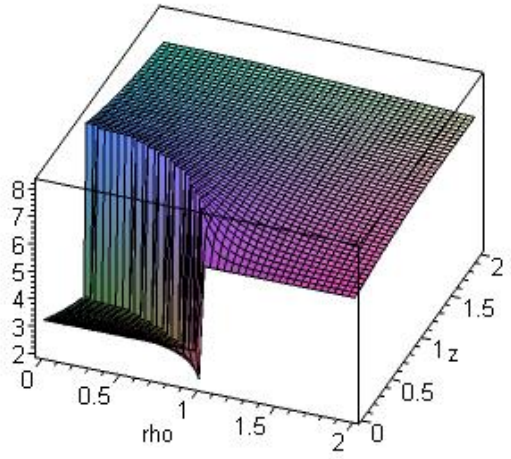


Here I am trying to "light up" the condition $u = u_0 + .5$ which is = 2.46 which lies between u_0 and π. I think this is probably the lower curve above, but I am getting a "ghost" upper curve. So that is my next mystery: what is causing the ghost upper curve? So I am not even ready to look at the potential because I can't even get "u" to work! I have been working on this all morning, perhaps 4 hours with no real progress. Par for the course.

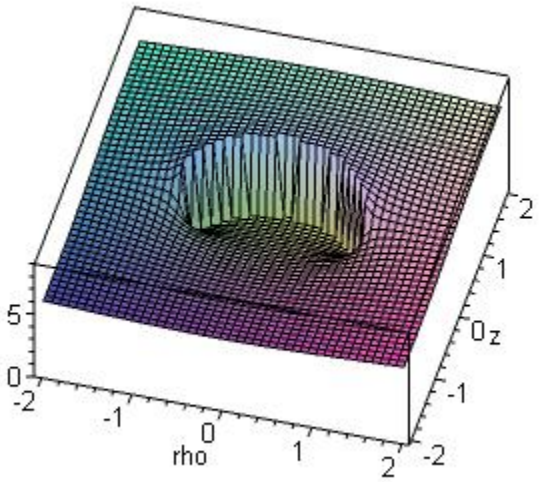
As I increase the adder, $u = u_0 +$ adder, the lower curve moves down as it should, while the upper curve stays right where it is! It is in a fixed position all the time!

OK, rather than pick out specific values like u = u_0+ 0.5, let's just plot the entire first quadrant of u

```
plot3d(getu(rho,z), rho = 0..2, z = 0..2,numpoints=2000,axes=BOXED);
```



and below is a fuller shot. It seems exactly right! The curved low line is u = u_0 and as we move away from it u builds up, gets to π on the line between the foci, then builds to 2π+u_0 = 9.2 at the top of the waterfall.



OK, I think I see what is happening. If we try to plot u_0 + .5, say, we first get the desired curve. Then the vertical edge of the waterfall is somehow "rough" and we hit this value at random places right on that near vertical surface. Consider then a (ρ,z) very close to (or at) a point on the bottom of the waterfall. Our algorithm computes a number that can be u_0+ε or it can be u_0-ε+2π. You cannot be on the vertical surface, you are either up or you are down. How could "noise" (computational error) cause us to be halfway up the wall, say?

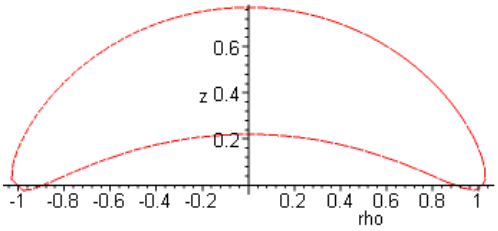
Back to the logic analyzer. I need to capture a point on the ghost curve!

I now think this is a bug inside implicitplot. If I just plot the surface u, it looks exactly right. I have no way to "trap" points that "make the cut" inside implicitplot and are put into the red points bin.

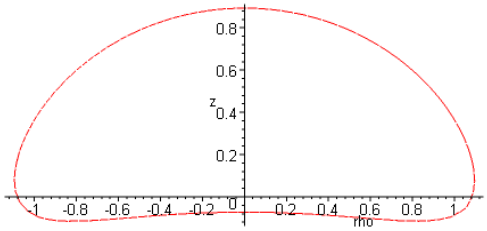
So perhaps it has even worst trouble when I try to plot V.

Well, when I plot V, implicitplot seems to work better.

V = 0.9



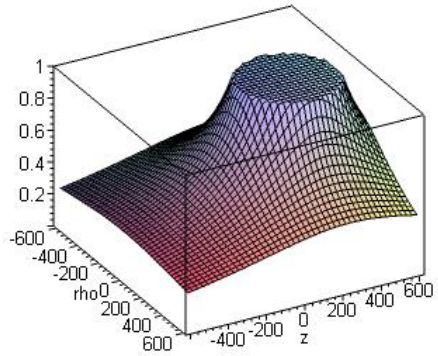
V = 0.8



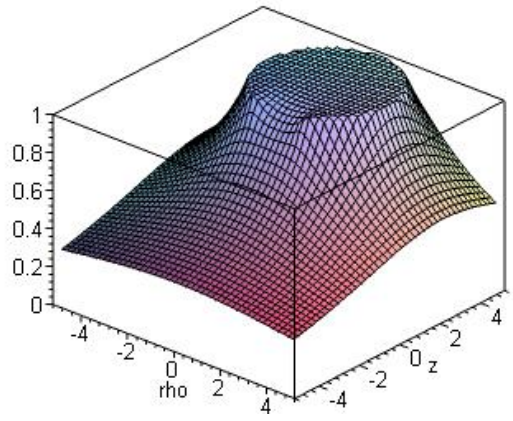
I now realize that it is better to just do a 3D plot of the potential and forget about implicitplot.

So here now is a gallery of nice 3D plots showing the potential of the charged bowl V as the height, where the base coordinates are ρ and z

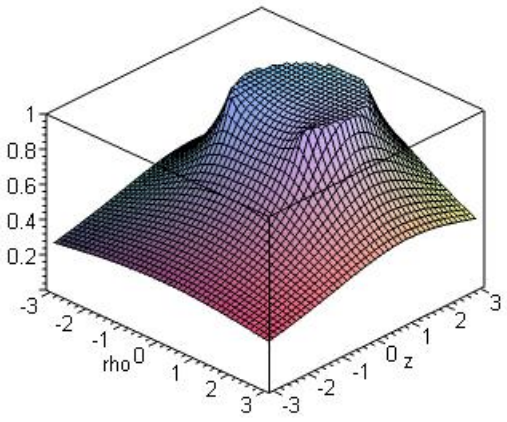
$u_0 = .01\pi/8$:



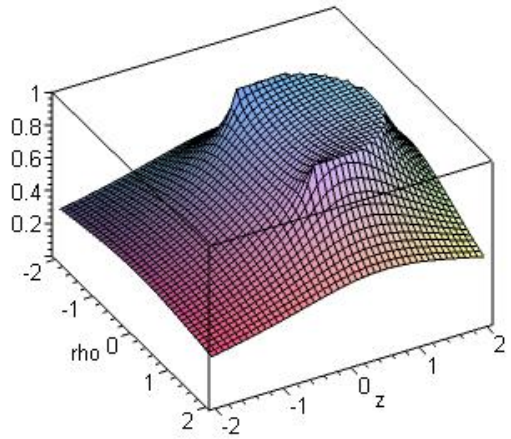
$u_0 = \pi/8$: a bowl that is almost a complete sphere



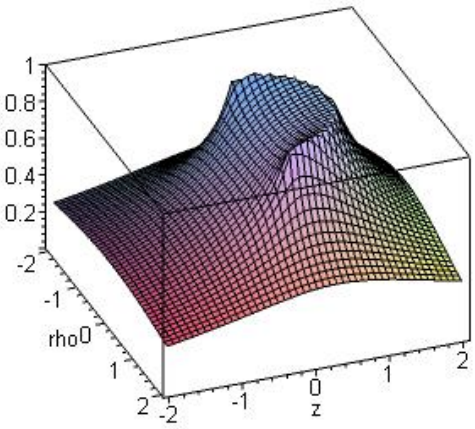
$u_0 = 2\pi/8$:



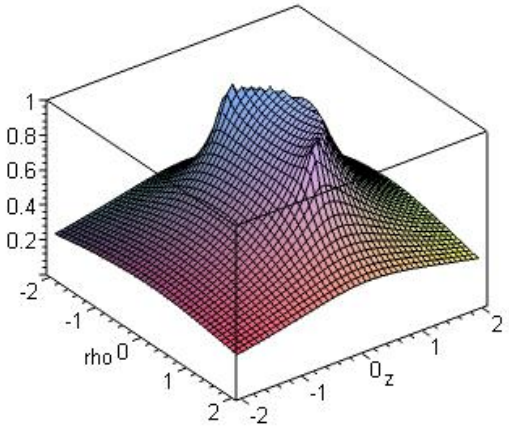
$u_0 = 3\pi/8$:



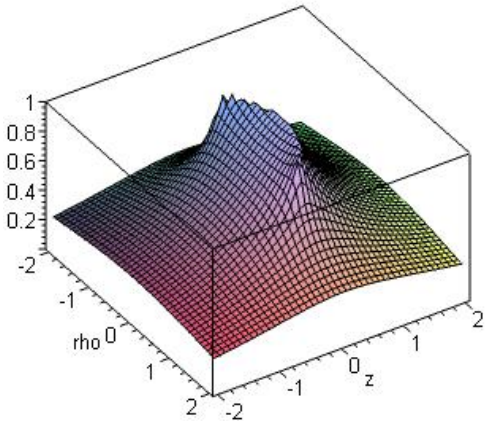
$u_0 = 4\pi/8$, the hemispherical bowl



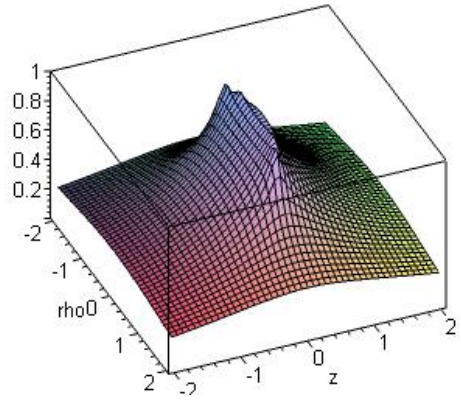
$u_0 = 5\pi/8$



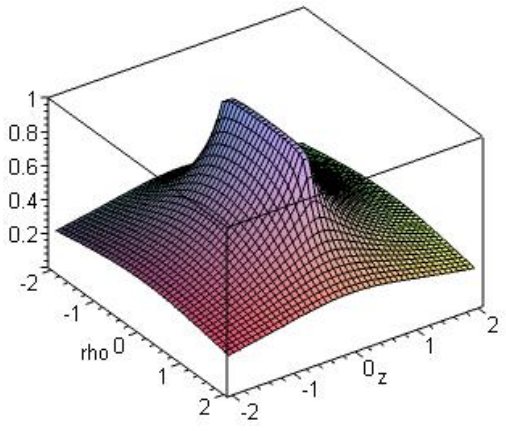
$u_0 = 6\pi/8$



$u_0 = 7\pi/8$



$u_0 = \pi$, potential of a disk!!



This is all very excellent. In particular, when $u_0 = \pi$, the bowl slice becomes the line between the two foci, and the bowl becomes a perfect disk of radius a. So built into this problem solution is an exact plot

of the disk potential. You can see, for example, that if you move 1" off the edge of the disk, V drops faster than if you move 1" out from the disk center.

Side note: the fact that these plots all look good is vindication of my post-evaluation formula for the toroidal bowl potential. It would be impossible to plot the pre-evaluation integrals!