T. How would you PLOT the charge density on a charged metal ellipsoid using Maple?

Here is our expression obtained above for the surface charge σ on our charged metal ellipsoid,

$$4\pi\sigma = E^{\perp} = -dV/dn = -dV(\xi_1)/d\xi_1\ (d\xi_1/dn) = -[\ dV(\xi_1)/d\xi_1]\ (1/h_{\xi 1})$$

$$-[\ dV(\xi_1)/d\xi_1]\quad \sqrt{(\xi_1^2 - a^2)\,(\xi_1^2 - b^2)}\ \ /\ \sqrt{(\xi_1^2 - \xi_2^2)\,(\xi_1^2 - \xi_3^2)}$$

so all the interest is in the last factor, since $\xi_1 = c$, a constant for our ellipsoid.

The plan is to make the surface brightness be proportional to σ (but later I try hue as well). To that end, we set f equal to the ratio of radicals shown above, which is normalized to unity at $(\xi_2, \xi_3) = (a,b)$. This point is the tip of the longest end of the ellipsoid, so we expect that it will be the brightest point. (I will insert the full *text* code below).

```
f := (sqrt(c^2 - a^2)*sqrt(c^2 - b^2))/(sqrt(c^2 - xi2^2)*sqrt(c^2 - xi3^2));
```

$$f := \frac{\sqrt{c^2 - a^2}\,\sqrt{c^2 - b^2}}{\sqrt{c^2 - \xi 2^2}\,\sqrt{c^2 - \xi 3^2}}$$

Next, we enter our "x,y,z equations" to map from ellipsoidal coordinate ξ space to Cartesian space

```
x := sqrt((c^2-a^2)*(xi2^2-a^2)*(xi3^2-a^2)/(a^2*(a^2-b^2)));
```

$$x := \sqrt{\frac{(c^2 - a^2)\,(\xi 2^2 - a^2)\,(\xi 3^2 - a^2)}{a^2\,(a^2 - b^2)}}$$

```
y := sqrt((c^2-b^2)*(xi2^2-b^2)*(xi3^2-b^2)/(b^2*(b^2-a^2)));
```

$$y := \sqrt{\frac{(c^2 - b^2)\,(\xi 2^2 - b^2)\,(\xi 3^2 - b^2)}{b^2\,(-a^2 + b^2)}}$$

```
z := c*xi2*xi3/(a*b);
```

$$z := \frac{c\,\xi 2\,\xi 3}{a\,b}$$

which can be compared with our earlier MF equations where we have set $\xi_1 = c$. (Notice also $h_{\xi 1}$)

$$x = \sqrt{\frac{(\xi_1^2 - a^2)(\xi_2^2 - a^2)(\xi_3^2 - a^2)}{a^2(a^2 - b^2)}};\quad y = \sqrt{\frac{(\xi_1^2 - b^2)(\xi_2^2 - b^2)(\xi_3^2 - b^2)}{b^2(b^2 - a^2)}};$$

$$z = \frac{\xi_1\xi_2\xi_3}{ab};\quad h_1 = \sqrt{\frac{(\xi_1^2 - \xi_2^2)(\xi_1^2 - \xi_3^2)}{(\xi_1^2 - a^2)(\xi_1^2 - b^2)}};\quad h_2 = \sqrt{\frac{(\xi_2^2 - \xi_1^2)(\xi_2^2 - \xi_3^2)}{(\xi_2^2 - a^2)(\xi_2^2 - b^2)}};$$

Next, we want to have the user enter the three semi-major axes to define an ellipsoid that way, and from these we compute the two long axis focal distances a and b,

```
A := 4;      // x, shortest
```

$$A := 4$$

```
B := 6;      //y
```

$$B := 6$$

```
C := 10;      //z, longest
```

$$C := 10$$

```
c := C;
```

$$c := 10$$

```
a := evalf(sqrt(C^2-A^2));
```

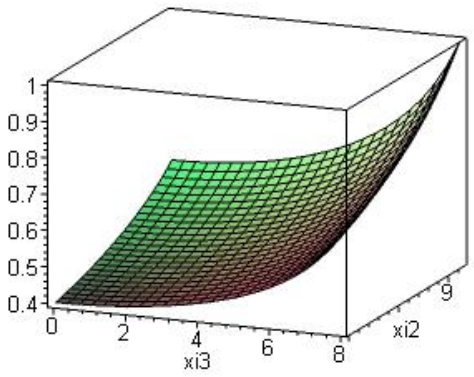
$$a := 9.165151390$$

```
b := evalf(sqrt(C^2-B^2));
```

$$b := 8.$$

Next, for warmup we just plot the surface charge density versus ξ_2 and ξ_3 . Notice that the two variables (ξ_2 , ξ_3) = (xi2,xi3) are scanned over their legal ranges which are $0 \le \xi_3 \le b$ and $b \le \xi_2 \le a$,

```
plot3d(f, xi3=0..b,xi2=b..a, axes=boxed);
```

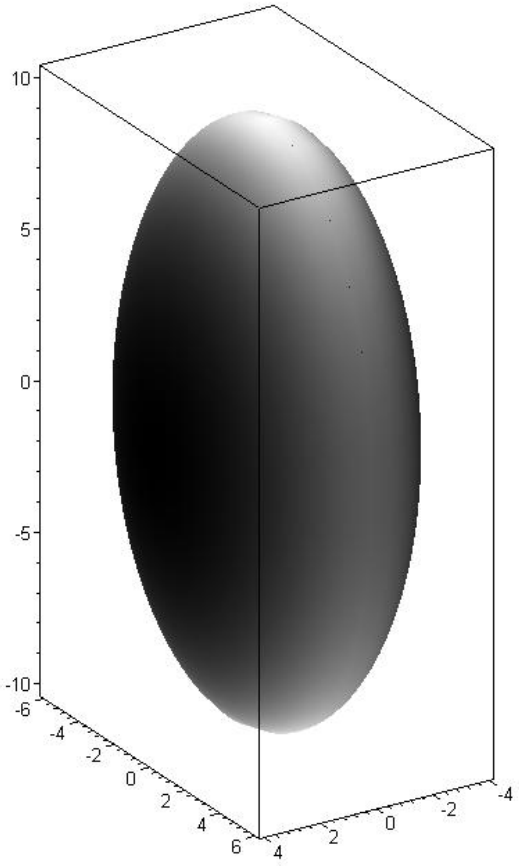


This shows that our scaled σ ranges from about 0.4 up to 1.0 in this example. The upper corner is at the largest values of a and b, which as noted correspond to the longest top of the ellipsoid which we expect to have the most charge.

Next, we create the desired "football plot". The code is this:

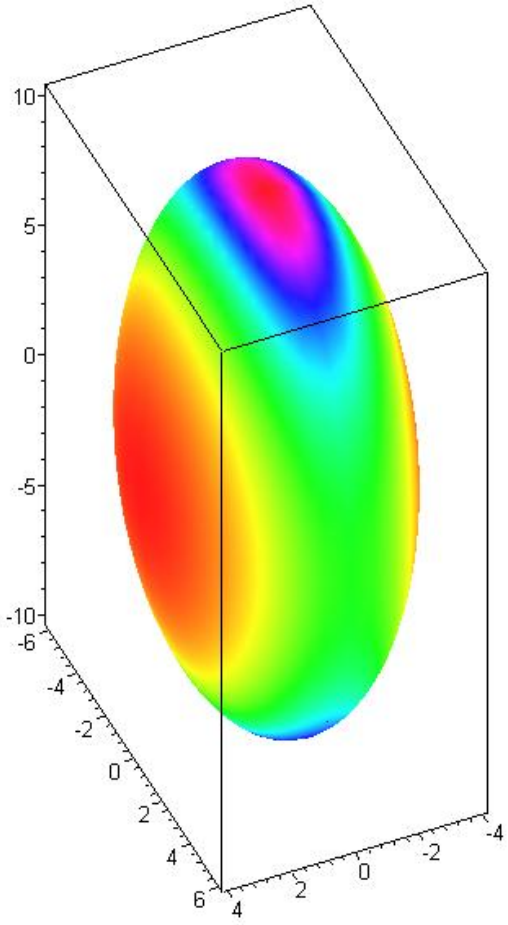
```
with(plots):
octants := {[x,y,z],[-x,y,z],[x,-y,z],[x,y,-z],[-x,-y,z],[-x,y,-z],[x,-y,-z],[-x,-y,-z]}:
plot3d(octants, xi3=0..b,xi2=b..a, axes=boxed, scaling = CONSTRAINED, grid=[30,30], color =
[f,f,f], style=PATCHNOGRID);
```

In order to see all 8 octants of our ellipsoid, we create a set {} "octants" containing the obvious 8 points. Then the plot3d command "scans" the space (ξ_2,ξ_3) = (xi2,xi3) just as in the previous plot. At each scanning point, it computes the 8 mirror points and puts them in a data list along with a grayscale color set by our color function f (which is just σ, the surface charge), represented by RGB = [f,f,f]. You can at least see that the charge is most on the pointiest ends and least on the most oblate ends.

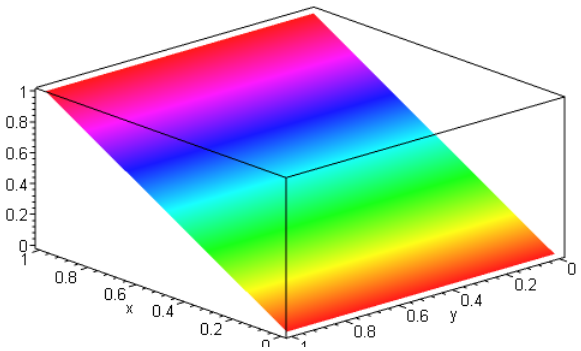


Unfortunately, Maple *scales* the RGB "color" so it fills the entire range from 0 to 1, whereas we would like to see the color being roughly 0.4 to 1 as shown by the previous graph. I cannot find a way around this problem. (I tried lots of tricks, you would have to directly edit the data structure).

The hue method is a little more exciting. Instead of having it span the luminance spectrum as I did above, we can let it span the hue spectrum by saying "color=f". Here is the plot along with a calibration hue spectrum



plot3d(x,x=0..1,y=0..1, color = x, style=PATCHNOGRID);



Now the pointy ends of our ellipsoid are red as expected, the most oblate ends are orange, and the intermediate ends are green. In some sense, this is a very good way to plot σ, as long as we understand that the full hue spectrum is only mapping σ in the range (0.4, 1.0). Hue is very sensitive to the eye.

Text code:

```
> restart:
> f := (sqrt(c^2 - a^2)*sqrt(c^2 - b^2))/(sqrt(c^2 - xi2^2)*sqrt(c^2 - xi3^2)):
> x := sqrt((c^2-a^2)*(xi2^2-a^2)*(xi3^2-a^2)/(a^2*(a^2-b^2))):
> y := sqrt((c^2-b^2)*(xi2^2-b^2)*(xi3^2-b^2)/(b^2*(b^2-a^2))):
> z := c*xi2*xi3/(a*b):
> A := 4:   // x, shortest
> B := 6:   //y
> C := 10:   //z, longest
> c := C:
> a := evalf(sqrt(C^2-A^2)):
> b := evalf(sqrt(C^2-B^2)):
> plot3d(f, xi3=0..b,xi2=b..a, axes=boxed):
> with(plots):
> octants := {[x,y,z],[-x,y,z],[x,-y,z],[x,y,-z],[-x,-y,z],[-x,y,-z],[x,-y,-z],[-x,-y,-z]}:
> plot3d(octants, xi3=0..b,xi2=b..a, axes=boxed, scaling = CONSTRAINED, grid=[30,30], color= f , style=PATCHNOGRID);
> restart;
> plot3d(x,x=0..1,y=0..1, color = x, style=PATCHNOGRID);
```