A Description of *PlotCont*

C. Godsalve
F2 Swainstone Road
Reading
RG2 ODX
United Kingdom
Tel: (0118) 9864389

email:`seagods@hotmail.com`

Contents

1 The Problem

In the *Plot* routine, we had a function $F(x, y)$ defined at a set of nodes in the (x, y) plane. These points had a triangulation. The triangulation could have been defined on a regular grid, or it may have been a Delaunay triangulation on a set of arbitrary points.

We then mapped the triangulation in the two dimensional plane to a two dimensional surface embedded in a three dimensional space. That is we mapped $(x_i, y_i) \rightarrow (x_i, y_i, z_i)$ where $z_i = F(x_i, y_i)$. In *ContPlot*, we do not map the triangulation into a three dimensional space, but use contours instead.

Here, we give a brief description of how we do it. We have an array called *triangles* which is three times the length of the number of triangles. Then $*(triangles+0)$, $*(triangles+1)$, $*(triangles + 2)$ hold the nodes of the first triangle, and so on. (Rather odd I know, but it seemed like a good ideas at the time.)

For each triangle, we have the function values at the nodes, so for Contour levels are determined in the same way as the tickmarks on an axis. So, for a particular contour level, we can tell if it is cut if the function values are not all higher, or all lower, than the current contour. We can then work out the "cuts" on the edges via linear interpolation. We allow each triangle to be cut by more than one contour. The maximum allowed is set by *klinesmax* in the *PlotCont* header file. We look for cuts between edge 1 and edge 2, edge 1 and edge 3, and edge 2 and edge 3.

For each triangle, we loop over the contours. We work out the positions of cuts between node pairs. For instance, if there is a cut between nodes 1 and 2, this will have a cut at position $(xc12, yc12)$. Given all these, we first check an edge 1 to edge 2 line segment. If it is there, we store the cut in $contx1$, $contx2$ $conty1$, and $conty2$, and increment a kounter *kl*. These countour line segments are (effectively) a matrix where the rows are the triangle numbers and the columns the line segment number in that triangle. The number of line segments in each triangle is stored in a integer array *klines*.