Zlab: Explore the complex plane in Matlab

Robert L. Parker

A series of *matlab* scripts allows the user to generate informative views of an analytic function of a complex variable in the complex plane. The user provides the matlab formula for the function, which can be displayed in various ways: magnitude-phase contour diagram, with singularities and branch cuts indicated; perspective views of the function magnitude, or its real and imaginary parts. Help scripts are provided enabling the user to move around the plane, or to zoom in on interesting features, and to rotate any of the perspective plots. In addition, a program is provided for performing a contour integral numerically on a path specified by mouse over the plane.

Zlab comprises a series of scripts and runs essentially as command-line language, with occasionally mouse input. The master routine is **zlab** which generates the function values and updates the contour map. All the other routines work on the values that **zmap** creates; some invoke **zmap** to move the center of the graph window or investigate the *z* plane on a new scale.

Here is an example of a run made with *zlab*. If you type in these instructions using the top function in the script *fun.m* provided various features of the complex plane will be illuminated. The top function is a hankel function.

```
zmap          %  Plots a magnitude-phase contour plot of the Hankel function
              %  The log branch cut and a simple zero appear
solid         %  A perspective fish-net plot is drawn of the same area
turn          %  The perspective figure is rotated
zooom         %  Move in, getting a closer view of the branch point
zooom         %  Move in further
pan           %  Using the mouse, chose another center for the plot, say
              %  the saddle point at 0.0575 - 0.0838i
pullback      % Reverse the last zooom
pullback
pullback      % We can see a larger area, including more zeros.
clear         %  Restore all default settings
```

Prolog

At first blush it may seem trivial to get *matlab* to contour up analytic function of a complex variable; why would anyone need a whole collection of special scripts? To do the job right, however, requires a great deal of attention to detail. For example, simply contouring the phase of *f(z)* in a naive fashion leads to lots of jagged lines. Similarly, because most interesting functions have singularities, or exponential growth, run-away maximum values makes elementary contouring impractical: the peaks have to be suppressed to make sure vital detail isn't lost. And then there is *matlab's* baroque profusion of commands to get the axes labeled and the figure proportions right, and so on. Once sorted out, who wants to remember them all? Better to have them safely tucked away in *zlab*..

These scripts operate on a set of variables that are all out in the common workspace. The defaults are invoked unless the user explicitly sets them. For example, **zmap** will choose the origin *z*=0 as the center of the main plot, and draw inside a square window that is 6.3 units high and wide. When **pan** or **zooom** are used, these parameters are reset. The user can start the plot on a different center simply by setting the 2-vector *center* to the real and imaginary parts of the new origin. The viewing window size is also set by a 2-vector, *side* which gives the x and y lengths of the viewing box.

Command Catalog

zmap: Draw a contour map centered at the origin in a square region 6.3 units high and wide of the analytic function of *z* defined in the function script *fun.m*. The map is drawn in figure 1. It is an magnitude map, indicated by color and black level lines; superimposed are white contours of phase, spaced by 30 degrees. Indicated in yellow and red are estimates of the singularities and the branch cuts as positioned by *matlab* defaults. If the user wishes to displace the cuts, he or she may be able to trick the *matlab* algorithm. For example, to get the square-root cut to run up the imaginary axis write:

```
y = (1 - i)*sqrt(i*z/2);
```

in *fun.m*. This may not be possible with more complicated functions like the Hankel function.

The command **zmap** must be the first one issued before any of the others in the catalog can be used. If *fun.m* needs to be modified, a separate window should be opened, and the script edited there, without having to stop *matlab*.

solid: In figure 2, draw a fishnet altitude plot of the function magnitude $|f(z)|$ in the same region as the **zmap** graph. This command draws on arrays computed by **zmap**. It can be viewed from a variety of angles with the command **turn**.

zview: In figure 3 draw two altitude plots, one for the real part the other the imaginary part of the function *f*.

turn: Rotate in 30 degree increments the most recently drawn altitude plot.

zooom: Recompute and redraw the **zmap** magnitude-phase plot on a larger scale, with center unaltered. The size of the viewing box is reduced by a factor of two-thirds by this command The native *matlab* command *zoom* still works, but it does not recompute the function, and so no more detail will be added as the viewer closes in on a feature of interest.

pullback: Reverse the effect of **zooom**.

pan : Recompute and redraw the **zmap** plot centered on a new point that is within the current viewing window. The new center is specified by double-clicking on the desired point. If the pointer is left in the viewing window, another center can be chosen without re-issuing the **pan** command, so that the user can conveniently focus the center of interest again. This command modifies the 2-vector *center*.

integrate: Perform a complex contour integration numerically over a path drawn on the **zmap** graph with the mouse. The path may be closed, if the first and last points are close together, or open if the program decides they are far apart. The instructions for mouse input are printed to the terminal and will not be repeated here.

zlab: A do-nothing script containg a comments that can be accessed by the matlab help system, as in

```
help zlab
```

More detailed information on each command can also be found by entering the command name in the help call; for exmaple
help integrate
The documentation that you are now reading is the most extensive however.

fun : This is not a *zlab* command, but a required function which **zmap** refers to in order to get the complex function under investigation. An example of such a function might be

```
function y = fun(z)
%  Integrand of a simple Fourier transform
y = exp(i*z) ./ (1 + z.^2); return
```

The calculations must be written to accept a complex matrix for the variable *z* and operations must not be those of linear algebra, but element-by-element. Thus, as shown in this example, *z* squared must be written z .^ 2 not as z^2 which is matrix multiply in *matlab*. The "return" instruction here allows the user to maintain a list of alternative functions below these lines in the script, any one of which can be promoted to the top to be executed when needed. A small library of interesting functions is provided in the *zlab* distribution.