Tip: Make a legend in the graph to list the contour traces. The trace names normally contain the contour levels. To do this, select the command window, type `Legend`, and press Return or Enter.

Similarly, if you don't want the triangulation or XY marker traces to be drawn, use the Modify Contour Appearance dialog to turn them off, rather than removing them with the Remove from Graph dialog.

Cursors on Contour Traces

You can attach cursors to a contour trace. Just like any other trace, the X and Y values are shown in the graph info panel (choose Show Info from the Graph menu). When the cursor is attached to a contour line the Z value (contour level) of the trace is also shown in the info panel.

There are several additional methods for displaying the Z value of a contour trace:

- The zcsr function returns the Z value of the trace the cursor is attached to. zcsr returns a NaN if the cursor is attached to a noncontour trace. You can use this in a procedure, or print the result on the command line using: `Print zcsr(A).`
- If you add an image behind the contour, you can use cursors to display the X, Y, and Z values at any point.
- The name of the trace the cursor is attached to shows up in the info panel, and the name usually contains the Z value.
- Contour labels show the Z value. Contour labels are tags that contain the \OZ escape code or TagVal(3). See **Contour Labels** on page II-377. You can drag these labels around by pressing Option (*Macintosh*) or Alt (*Windows*) while dragging a tag. See **Changing a Tag's Attachment Point** on page III-45.

Contour Trace Updates

Igor normally updates the contour traces whenever the contour data (matrix or XYZ triplets) changes or whenever the contour levels change. Because creating the contour traces can be a lengthy process, you can prevent these automatic updates through a pop-up menu item in the Modify Contour Appearance dialog. See **Update Contours Pop-Up Menu** on page II-369.

Preventing automatic updates can be useful when you are repeatedly editing the contour data in a table or from a procedure. Use the "once, now" pop-up item to manually update the traces.

Contour Trace Update Programming Note

Programmers should be aware of another update issue: contour traces are created in two steps. To understand this, look at this graph recreation macro that appends a contour to a graph and then defines the contour levels, styles and labels:

```
Window Graph1() : Graph
   PauseUpdate; Silent 1      // building window...
   Display /W=(11,42,484,303)
   AppendMatrixContour zw
   ModifyContour zw autoLevels={*,*,5}, moreLevels={0.5}
   ModifyContour zw rgbLines=(65535,0,26214)
   ModifyContour zw labels=0
   ModifyGraph lSize('zw=0')=3
   ModifyGraph lStyle('zw=0')=3
   ModifyGraph rgb('zw=0')=(0,0,65535)
   ModifyGraph mirror=2
EndMacro
```

First, the AppendMatrixContour operation runs and creates stub traces consisting of zero points. The ModifyContour and ModifyGraph operations that follow act on the stub traces. Finally, after all of the commands that define the graph have executed, Igor does an update of the entire graph, when the effect of the PauseUpdate operation expires when the graph recreation macro returns. This is the time when Igor does the actual contour computations which convert the stub traces into fully-formed contour traces.

This delayed computation prevents unnecessary computations from occurring when ModifyContour commands execute in a macro or function. The ModifyContour command often changes default contour level settings, rendering any preceding computations obsolete. For the same reason, the New Contour Plot