

Dashed Lines

You can display traces in graphs, as well as drawn lines, rectangles, ovals, and polygons, using various line styles. This table, generated from the ColorsMarkersLinesPatterns.pxp example experiment, shows Igor's default line styles:

It is usually not necessary, but you can edit the built-in dashed line styles using the Dashed Lines dialog by choosing Misc→Dashed Lines.

Dashed line 0 (the solid line) cannot be edited. If you need to create a custom dashed line pattern, we recommend that you modify the high numbered dashed lines, leaving the low number ones in their default state. This ensures that the low numbered patterns will be the same for everyone.

You can also change dashed lines with the **SetDashPattern** operation.

Dashed lines are stored with the experiment, so each experiment can have different dashed lines. You can capture the current dashed lines as the preferred dashed lines for new experiments.

0	—————
1
2
3	-----
4
5	-----
6	-----
7	-----
8	-----
9	-----
10	-----
11	-----
12	—————
13	—————
14	—————
15	—————
16	—————
17	-----

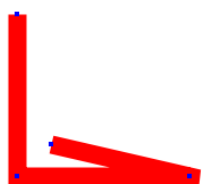
Line Join and End Cap Styles

When graph traces or drawn lines are very wide, you may want to control the appearance of the joins between line segments and the ends of line segments. The `ModifyGraph lineJoin` keyword provides control of line joins for graph traces in lines-between-points mode. The `SetDrawEnv lineJoin` keyword provides control for drawn lines.

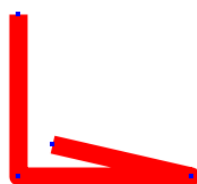
Line joins can have miter, round or bevel styles. For a miter join, you can specify the miter limit to control the length of miters on very acute angles. In these pictures, the blue dots show the position of the ends and join points of the lines:



`lineJoin={0, 10}`
miter, w/ miter limit=10



`lineJoin={0, sqrt(2)}`
miter, limit=sqrt(2)



`lineJoin={1, 0}`
round



`lineJoin={2, 0}`
bevel

Miter joins extend the outside edges of the line segments until they intersect. Acute angles may result in very long miters. Round joins draw a circular arc around the join point, and bevel joins truncate the intersection with a bevel at the intersection point.

You can set the miter limit to avoid very long miter extensions:

