

Built-In Routines

Each of Igor's built-in routines is categorized as a function or as an operation.

A built-in function is an Igor routine, such as sin, exp or ln, that directly returns a result. A built-in operation is a routine, such as Display, FFT or Integrate, that acts on an object and may create new objects but does not directly return a result.

A good way to get a sense of the scope of Igor's built-in routines is to scan the sections **Built-In Operations by Category** on page V-1 and **Built-In Functions by Category** on page V-7 in the reference volume of this manual.

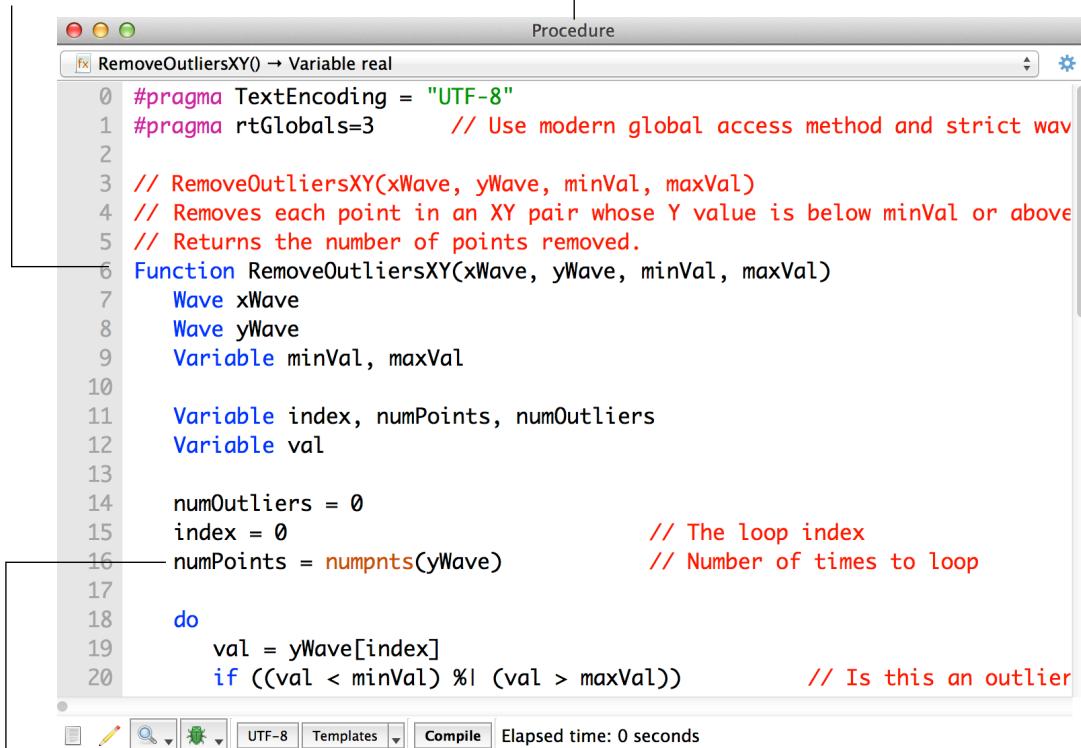
For getting reference information on a particular routine it is usually most convenient to choose Help→Command Help and use the Igor Help Browser.

User-Defined Procedures

A user-defined procedure is a routine written in Igor's built-in programming language by entering text in a procedure window. It can call upon built-in or external functions and operations as well as other user-defined procedures to manipulate Igor objects. Sets of procedures are stored in procedure files.

You can create Igor procedures by entering text in a procedure window.

Each procedure has a name which you use to invoke it.



```

Procedure
RemoveOutliersXY() → Variable real
0 #pragma TextEncoding = "UTF-8"
1 #pragma rtGlobals=3      // Use modern global access method and strict wav
2
3 // RemoveOutliersXY(xWave, yWave, minValue, maxValue)
4 // Removes each point in an XY pair whose Y value is below minValue or above
5 // Returns the number of points removed.
6 Function RemoveOutliersXY(xWave, yWave, minValue, maxValue)
7   Wave xWave
8   Wave yWave
9   Variable minValue, maxValue
10
11   Variable index, numPoints, numOutliers
12   Variable val
13
14   numOutliers = 0
15   index = 0                      // The loop index
16   numPoints = numpnts(yWave)       // Number of times to loop
17
18   do
19     val = yWave[index]
20     if ((val < minValue) || (val > maxValue))           // Is this an outlier

```

The screenshot shows the Igor Procedure window with the title bar 'Procedure'. The main area contains a code editor with the following text:

```

Procedure
RemoveOutliersXY() → Variable real
0 #pragma TextEncoding = "UTF-8"
1 #pragma rtGlobals=3      // Use modern global access method and strict wav
2
3 // RemoveOutliersXY(xWave, yWave, minValue, maxValue)
4 // Removes each point in an XY pair whose Y value is below minValue or above
5 // Returns the number of points removed.
6 Function RemoveOutliersXY(xWave, yWave, minValue, maxValue)
7   Wave xWave
8   Wave yWave
9   Variable minValue, maxValue
10
11   Variable index, numPoints, numOutliers
12   Variable val
13
14   numOutliers = 0
15   index = 0                      // The loop index
16   numPoints = numpnts(yWave)       // Number of times to loop
17
18   do
19     val = yWave[index]
20     if ((val < minValue) || (val > maxValue))           // Is this an outlier

```

The code uses color coding for keywords (e.g., 'Function', 'Variable', 'do', 'if') and comments (e.g., //). Line numbers are on the left. A vertical line from the text 'Each procedure has a name which you use to invoke it.' points to the word 'Procedure' in the title bar. Another vertical line from the text 'Procedures can call operations, functions or other procedures. They can also perform waveform arithmetic.' points to the 'yWave' variable in line 16.

Procedures can call operations, functions or other procedures. They can also perform waveform arithmetic.

Igor Extensions

An extension is a “plug-in” - a piece of external C or C++ code that adds functionality to Igor. For historical reasons, we use the term “XOP” to refer to an Igor extension. “XOP” is a contraction of “external operation”. The terms “XOP” and “Igor extension” are synonymous.