Waveform Arithmetic and Assignments

Waveform arithmetic is a very flexible and powerful part of Igor's analysis capability. You can write assignment statements that work on an entire wave or on a subset of a wave, much as you would write an assignment to a single variable in a standard programming language.

In a wave assignment statement, a wave appears on the left side and a mathematical expression appears on the right side. Here are some examples.

```
wave0 = sin(x)
wave0 = log(wave1/wave2)
wave0[0,99] = wave1[100 + p]
```

A wave on the left side is called the **destination wave**. A wave on the right side is called a **source wave**.

When Igor executes a wave assignment statement, it evaluates the expression on the right-hand side one time for each point in the destination wave. The result of each evaluation is stored in the corresponding point in the destination wave.

During execution, the symbol *p* has a value equal to the number of the point in the destination wave which is being set and the symbol *x* has a value equal to the X value at that point. The X value for a given point is determined by the number of the point and the X scaling for the wave. To see this, execute the following commands, one-at-a-time:

```
Make/N=5 wave0; SetScale/P x 0, .1, wave0; Edit wave0.xy
wave0 = p
wave0 = x
```

The first assignment statement sets the value of each point of wave0 to the point number. The second assignment statement sets the value of each point of wave0 to the X value for that point.

A source wave returns its data value at the point being evaluated. In the example

```
wave0 = log(wave1/wave2)
```

Igor evaluates the right-hand expression once for each point in wave0. During each evaluation of the expression, wave1 and wave2 return their data values at the point being evaluated. Consequently, these commands are equivalent:

```
wave0 = log(wave1/wave2)
```

```
wave0 = log(wave1[p]/wave2[p])
```

Understanding Wave Assignments

To understand wave assignments, it is critical to understand the meaning of p and x. To this end, it may be helpful to think of what happens inside Igor. A wave assignment statement causes a loop to run inside Igor.

p is the loop index of the internal loop and runs over the range of points being set in the destination wave. In the preceding example, p starts from 0 and is incremented each time through the internal loop, up to N-1, where N is the number of points in wave0. For each value of p, the right-hand expression is evaluated and the result is stored in the corresponding point of wave0.

When you see `wave1` or `wave1[p]` on the right-hand side of a waveform assignment statement, this returns the value of wave1 at point p, which is the point in the destination wave (wave0) that Igor is setting during the current iteration of the internal loop.

x is similar to p except that, instead of being the loop index of the internal Igor loop, it is calculated from the loop index using the X scaling of the destination wave:

```
x = x0 + p*dx
```