

Usually source waves have the same number of points and X scaling as the destination wave. In some cases, it is useful to write a wave assignment statement where this is not true. This is discussed under **Mismatched Waves** on page II-83.

Indexing and Subranges

Igor provides two ways to refer to a specific point or range of points in a 1D wave: X value indexing and point number indexing. Consider the following examples.

```
wave0[54] = 92           // Sets wave0 at point 54 to 92
wave0(54) = 92           // Sets wave0 at X=54 to 92
wave0[1,10] = 92         // Sets wave0 from point 1 to point 10 to 92
wave0(1,10) = 92         // Sets wave0 from X=1 to X=10 to 92
```

Brackets tell Igor that you are indexing the wave using point numbers. The number or numbers inside the brackets are interpreted as point numbers of the indexed wave.

Parentheses tell Igor that you are indexing the wave using X values. The number or numbers inside the parentheses are interpreted as X values of the indexed wave. When you use an X value as an index, Igor first finds the point number corresponding to that X value based on the indexed wave's X scaling, and then uses that point number as the point index.

If the wave has point scaling then these two methods have identical effects. However, if you set the X scaling of the wave to other than point scaling then these commands behave differently. In both cases the range is inclusive.

You can specify not only a range but also a point number increment. For example:

```
wave0[0,98;2] = 1        // Sets even numbered points in wave0 to 1
wave0[1,99;2] = -1       // Sets odd numbered points in wave0 to -1
```

The number after the semicolon is the increment. Igor begins at the starting point number and goes up to and including the ending point number, skipping by the increment. At each resulting point number, it evaluates the right-hand side of the wave assignment statement and sets the destination point accordingly.

Increments can also be used when you specify a range in terms of X value but the increment is always in terms of point number. For example:

```
wave0(0,100;5) = PI      // Sets wave0 at specified X values to PI
```

Here Igor starts from the point number corresponding to $x = 0$ and goes up to and including the point number that corresponds to $x = 100$. The point number is incremented by 5 at each iteration.

You can take some shortcuts in specifying the range of a destination wave. The subrange start and end values can both be omitted. When the start is omitted, point number zero is used. When the end is omitted, the last point of the wave is used. You can also use a * character or INF to specify the last point. An omitted increment value defaults to a single point.

Here are some examples that illustrate these shortcuts:

```
wave0[ ,50] = 13         // Sets wave0 from point 0 to point 50
wave0[51,] = 27          // Sets wave0 from point 51 to last point
wave0[51,*] = 27         // Sets wave0 from point 51 to last point
wave0[51,INF] = 27       // Sets wave0 from point 51 to last point
wave0[ , ;2] = 18.7       // Sets every even point of wave0
wave0[1,*;2] = 100        // Sets every odd point of wave0
```

A subrange of a destination wave may consist of a single point or a range of points but a subrange of a source wave must consist of a single point. In other words the wave assignment statement:

```
wave1(4,5) = wave2(5,6)  // Illegal!
```