Using Operations in Dependency Formulas

The dependency formula must be a single expression — and you can not use operations, such as FFT's, or other command type operations. However, you can invoke user-defined functions which invoke operations. For example:

```
Function MakeDependencyUsingOperation()
   Make/O/N=128 data = p         // A ramp from 0 to 127
   Variable/G power

   SetFormula power, "RMS(data)" // Dependency on function and wave
   Print power

   data = p * 2                  // Changes something power depends
   DoUpdate                      // Make Igor recalc formulae
   Print power
EndMacro

Function RMS(w)
   Wave w

   WaveStats/Q w           // An operation! One output is V_rms
   return V_rms
End
```

When `MakeDependencyUsingOperation` is executed, it prints the following in the history area:

```
•MakeDependencyUsingOperation()
   73.4677
   146.935
```

Dependency Caveats

The extensive use of dependencies can create a confusing tangle that can be difficult to manage. Although you can use the Object Status dialog to explore the dependency hierarchy, you can still become very confused very quickly, especially when the dependencies are highly cascaded. You should use dependencies only where they are needed. Use conventional assignments for the majority of your calculations.

Dependency formulas are generally not recalculated when a user-defined function is running unless you explicitly call the DoUpdate operation. However, they can run at other unpredictable times so you should not make any assumptions as to the timing or the current data folder when they run.

The text of the dependency formula that is saved for a dependent object is the original literal text. The dependency formula needs to be recompiled from time to time, for example when procedures are compiled. Therefore, any objects used in the formula must persist until the formula is deleted.

We recommend that you never use $ expressions in a dependency formula.