Other Uses of Execute

Execute can also accept as an argument a string variable containing commands that are algorithmically constructed. Here is a simple example:

```
Function Fit(w, fitType)
   WAVE w               // Source wave
   String fitType       // One of the built-in Igor fit types

   String name = NameOfWave(w)
   name = PossiblyQuoteName(name)   // Liberal names need quotes

   String cmd
   sprintf cmd, "CurveFit %s %s", fitType, name
   Execute cmd
End
```

Use this function to do any of the built-in fits on the specified wave. Without using the Execute operation, we would have to write it as follows:

```
Function Fit(w, fitType)
   WAVE w                  // Source wave
   String fitType          // One of the built-in Igor fit types

   strswitch(fitType)
      case "line":
         CurveFit line w
         break

      case "exp":
         CurveFit exp w
         break

      <and so on for each fit type>
   endswitch
End
```

Note the use of sprintf to prepare the string containing the command to be executed. The following would not work because when Execute runs, local variables and parameters are not accessible:

```
Execute "CurveFit fitType name"
```

Deferred Execution Using the Operation Queue

It is sometimes necessary to execute a command after the current function has finished. This is done using the Execute/P operation. For details, see **Operation Queue** on page IV-278.

Procedures and Preferences

As explained under **Preferences** on page III-515, Igor allows you to set many preferences. Most of the preferences control the style of new objects, including new graphs, traces and axes added to graphs, new tables, columns added to tables, and so on.

Preferences are usually used only for manual "point-and-click" operation. We usually don't want preferences to affect the behavior of procedures. The reason for this is that we want a procedure to do the same thing no matter who runs it. Also, we want it to do the same thing tomorrow as it does today. If we allowed preferences to take effect during procedure execution, a change in preferences could change the effect of a procedure, making it unpredictable.

By default, preferences do not take effect during procedure execution. If you want to override the default behavior, you can use the Preferences operation. From within a procedure, you can execute `Preferences 1` to turn preferences on. This affects the procedure and any subroutines that it calls. It stays in effect until you execute `Preferences 0`.