With rtGlobals=3, this function has errors on both lines:

```
Function Test()
   Display jack                          // Error: Expected wave reference
   Variable tmp = mean(jack,0,100)       // Error: Expected wave reference
End
```

The proper way to do this is to create a wave reference, like this:

```
Function Test()
   WAVE jack
   Display jack                          // OK
   Variable tmp = mean(jack,0,100)       // OK
End
```

The purpose of the strict wave access mode is to detect inadvertent name mistakes. This applies to simple names only, not to full or partial paths. Even with rtGlobals=3, it is OK to use a full or partial path where a wave reference is expected:

```
Function Test()
   Display :jack                         // OK
   Variable tmp = mean(root:jack,0,100)  // OK
End
```

If you have old code that is impractical to fix, you can revert to using rtGlobals=1 or rtGlobals=2.

Wave Reference Function Results

Advanced programmers can create functions that return wave references using Function/WAVE:

```
Function/WAVE Test(wIn) // /WAVE flag says function returns wave reference
   Wave wIn    // Reference to the input wave received as parameter

   String newName = NameOfWave(wIn) + "_out" // Compute output wave name

   Duplicate/O wIn, $newName                 // Create output wave

   Wave wOut = $newName       // Create wave reference for output wave
   wOut += 1                  // Use wave reference in assignment statement

   return wOut                // Return wave reference
End
```

This function might be called from another function like this:

```
Make/O/N=5 wave0 = p
Wave wOut = Test(wave0)
Display wave0, wOut
```

This technique is useful when a subroutine creates a free wave for temporary use:

```
Function Subroutine()
   Make/FREE tempWave = <expression>
   return tempWave
End

Function Routine()
   Wave tempWave = Subroutine()
   <Use tempWave>
End
```

When Routine returns, tempWave is automatically killed because all references to it have gone out of scope.