```
   Duplicate/O/C wv,dupWv      // dupWv is complex
   dupWv[0]=cmplx(5.0,1.0)     // no error, because dupWv known complex
   . . .
End
```

If Duplicate did not have the /C flag, you would get a "function not available for this number type" message when compiling the assignment of `dupWv` to the result of the `cmplx` function.

Problems with Automatic Creation of WAVE References

Operations that change a wave's type or which can create output waves of more than one type, such as **FFT**, **IFFT** and **WignerTransform** present special issues.

In some cases, the wave reference automatically created by an operation might be of the wrong type. For example, the FFT operation automatically creates a complex wave reference for the destination wave, so if you write:

```
FFT/DEST=destWave srcWave
```

it is as if you wrote:

```
FFT/DEST=destWave srcWave
WAVE/C destWave
```

However, if a real wave reference for the same name already exists, FFT/DEST does not create a new wave reference. For example:

```
Wave destWave                 // Real wave reference
. . .
FFT /DEST=destWave srcWave    // FFT does not create wave reference
                              // because it already exists.
```

In this case, you would need to create a complex wave reference using a different name, like this:

```
Wave/C cDestWave = destWave
```

The output of the FFT operation can sometimes be real, not complex. For example:

```
FFT/DEST=destWave/OUT=2 srcWave
```

The /OUT=2 flag creates a real destination wave. Thus the complex wave reference automatically created by the FFT operation is wrong and can not be used to subsequently access the destination wave. In this case, you must explicitly create a real wave reference, like this:

```
FFT/DEST=destWave/OUT=2 srcWave
WAVE realDestWave = destWave
```

Note that you can not write:

```
FFT/DEST=destWave/OUT=2 srcWave
WAVE destWave
```

because the FFT operation has already created a complex wave reference named `destWave`, so the compiler will generate an error. You must use a different name for the real wave reference.

The **IFFT** has a similar problem but in reverse. IFFT automatically creates a real wave reference for the destination wave. In some cases, the actual destination wave will be complex and you will need to create an explicit wave reference in order to access it.

WAVE Reference Is Needed to Pass a Wave By Name

As of Igor Pro 6.3, new procedure windows are created with this pragma statement:

```
#pragma rtGlobals=3 // Use modern global access method and strict wave access.
```

The strict wave access mode causes a compile error if you pass a literal wave name as a parameter to an operation or function. Instead you must pass a wave reference.