$$\gamma = \frac{\sum_{i=0}^{M} F(s_{Ai})[F(s_{Bi})]^*}{\sqrt{\sum_{i=0}^{M} F(s_{Ai})[F(s_{Ai})]^* \sum_{i=0}^{M} F(s_{Bi})[F(s_{Bi})]^*}}.$$

The bias in the degree of coherence is calculated using the approximation

$$B = \frac{1}{M}\left[1 - |\gamma|^2\right]^2.$$

The bias is stored in the wave W_Bias.

If you use the /SEGN flag the actual number of segments is reported in the variable V_numSegments.

Note that DSPPeriodogram does not test the dimensionality of the wave; it treats the wave as 1D. When you compute the cross-spectral density or the degree of coherence the number-type, dimensionality and the scaling of the two waves must agree.

### Normalization Satisfying Parseval's Theorem

After executing DSPPeriodogram with the /PARS flag, you can check that the normalization satisfies Parseval's theorem using this function:

```
Function CheckNormalization(srcWave, periodogramWave)
   Wave srcWave                         // A real valued time series
   Wave periodogramWave                 // e.g., W_Periodogram

   Duplicate/FREE periodogramWave,wp    // Preserve original
   wp[0]/=2                             // Correct the 0 bin
   wp[numpnts(wp)-1] /=2                // Correct the Nyquist bin
   MatrixOP/FREE w2=magsqr(srcWave)/numPoints(srcWave)
   Print sum(wp), sum(w2)               // Parseval: These should be equal
End
```

### See Also

The **ImageWindow** operation for 2D windowing applications. **FFT** for window equations and details.

The **Hanning**, **LombPeriodogram** and **MatrixOp** operations.

### References

For more information about the use of window functions see:

Harris, F.J., On the use of windows for harmonic analysis with the discrete Fourier Transform, *Proc, IEEE*, *66*, 51-83, 1978.

G.C. Carter, C.H. Knapp and A.H. Nuttall, The Estimation of the Magnitude-squared Coherence Function Via Overlapped Fast Fourier Transform Processing, *IEEE Trans. Audio and Electroacoustics*, V. AU-21, (4) 1973.

## Duplicate

**`Duplicate`** `[`***`flags`***`][`***`type flags`***`]` ***`srcWaveName`***, ***`destWaveName`*** `[,` ***`destWaveName`***`]…`

The Duplicate operation creates new waves, the names of which are specified by *destWaveName*s and the contents, data type and scaling of which are identical to *srcWaveName*.

### Parameters

*srcWaveName* must be the name of an existing wave.

The *destWaveName*s should be wave names not currently in use unless the /O flag is used to overwrite existing waves.

### Flags

/FREE[=*nm*] Creates a free wave. Allowed only in functions and only if a simple name or wave reference structure field is specified.

See **Free Waves** on page IV-91 for further discussion.

If *nm* is present and non-zero, then waveName is used as the name for the free wave, overriding the default name '_free_'. The ability to specify the name of a free wave was added in Igor Pro 9.00 as a debugging aid - see **Free Wave Names** on page IV-95 and **Wave Tracking** on page IV-207 for details.

/O Overwrites existing waves with the same name as *destWaveName*.

/R=(*startX*,*endX*) Specifies an X range in the source wave from which the destination wave is created.

See Details for further discussion of /R.

/R=(*startX*,*endX*)(*startY*,*endY*)

Specifies both X and Y range. Further dimensions are constructed analogously.

See Details for further discussion of /R.

/R=[*startP*,*endP*] Specifies a row range in the source wave from which the destination wave is created. Further dimensions are constructed just like the scaled dimension ranges.

See Details for further discussion of /R.

/RMD=[*firstRow*,*lastRow*][*firstColumn*,*lastColumn*][*firstLayer*,*lastlayer*][*firstChunk*,*lastChunk*]

Designates a contiguous range of data in the source wave to which the operation is to be applied. This flag was added in Igor Pro 7.00.

You can include all higher dimensions by leaving off the corresponding brackets. For example:

```
/RMD=[firstRow,lastRow]
```

includes all available columns, layers and chunks.

You can use empty brackets to include all of a given dimension. For example:

```
/RMD=[][firstColumn,lastColumn]
```

means "all rows from column A to column B".

You can use a * to specify the end of any dimension. For example:

```
/RMD=[firstRow,*]
```

means "from firstRow through the last row".

### Type Flags *(used only in functions)*

When used in user-defined functions, Duplicate can also take the /B, /C, /D, /I, /S, /U, /W, /T, /DF and /WAVE flags. This does not affect the result of the Duplicate operation - these flags are used only to identify what kind of wave is expected at runtime.

This information is used if, later in the function, you create a wave assignment statement using a duplicated wave as the destination:

```
Function DupIt(wv)
   Wave/C wv                  //complex wave

   Duplicate/O/C wv,dupWv     //tell Igor that dupWv is complex
   dupWv[0]=cmplx(5.0,1.0)    //no error, because dupWv known complex
   …
```

If Duplicate did not have the /C flag, Igor would complain with a “function not available for this number type” message when it tried to compile the assignment of dupWv to the result of the cmplx function.

These type flags do not need to be used except when it needed to match another wave reference variable of the same name or to identify what kind of expression to compile for a wave assignment. See **WAVE Reference Types** on page IV-73 and **WAVE Reference Type Flags** on page IV-74 for a complete list of type flags and further details.