Details

If /R is omitted, the entire wave is duplicated.

In the range specifications used with /R, a `*` for the end means duplicate to the end. You can also simply leave out the end specification. To include all of a given dimension, use `/R=[]`. If you leave off higher dimensions, all those dimensions are duplicated. That is, `/R=[1,5]` for a 2D wave is equivalent to `/R=[1,5][]`.

The destination wave will always be unlocked, even if the source wave was locked.

Warning:

Under some circumstances, such as in loops in user-defined functions, Duplicate may exhibit undesired behavior. When you use

```
Duplicate/O srcWave, DestWaveName
```

in a user-defined function, it creates a local WAVE variable named *DestWaveName* at compile time. At runtime, if the WAVE variable is NULL, it creates a wave of the same name in the current data folder. If, however, the WAVE variable is not NULL, as it would be in a loop, then the referenced wave will be overwritten no matter where it is located. If the desired behavior is to create (or overwrite) a wave in the current data folder, you should use one of the following two methods:

```
Duplicate/O srcWave, $"DestWaveName"
WAVE DestWaveName   // only if you need to reference dest wave
```

or

```
Duplicate/O srcWave, DestWaveName
// then after you are finished using DestWaveName…
WAVE DestWaveName=$""
```

See Also

Rename, **Concatenate**, **SplitWave**

DuplicateDataFolder

`DuplicateDataFolder [ /O=`*`options`*` /Z ] `*`sourceDataFolderSpec`*`, `*`destDataFolderSpec`*

The DuplicateDataFolder operation makes a copy of the source data folder and everything in it and places the copy at the specified location with the specified name.

Parameters

sourceDataFolderSpec can be just the name of a child data folder in the current data folder, a partial path (relative to the current data folder) and name or an absolute path (starting from root) and name, or a data folder reference (DFREF) in a user-defined function.

destDataFolderSpec can be just a data folder name, a partial path (relative to the current data folder) and name or an absolute path (starting from root) and name. If just a data folder name is used then the new data folder is created in the current data folder, or a data folder reference (DFREF) in a user-defined function.

Depending on the syntax you use, DuplicateDataFolder may overwrite the data folder specified by destDataFolderSpec or it may copy the source data folder into the data folder specified by destDataFolderSpec. See **DuplicateDataFolder Destination** on page V-188 below for details.

Flags

/O=*options*	Overwrites the destination data folder if it already exists. The /O flag was added in Igor Pro 8.00. options=0: Attempting to overwrite a data folder generates an error, as if you omitted /O. options=1: Completely overwrites the destination data folder. If the destination data folder exists, DuplicateDataFolder first deletes it if possible. If it can not be deleted, for example because it contains a wave that is in use, DuplicateDataFolder generates an error. If the deletion succeeds, DuplicateDataFolder then copies the source to the destination.

options=2: Merges the source data folder into the destination data folder. If an item in the source data folder exists in the destination data folder, DuplicateDataFolder overwrites it. Otherwise it copies it. Items in the destination data folder that do not exist in the source data folder remain in the destination data folder.

options=3: Merges the source data folder into the destination data folder and then deletes items that are not in the source data folder if possible. If an item in the source data folder does not exist in the destination data folder, DuplicateDataFolder attempts to delete it from the destination data folder. If it can not be deleted because it is in use, no error is generated.

/Z	Errors are not fatal - if an error occurs, procedure execution continues. You can check the V_flag output variable to see if an error occurred.

Details

An error is issued if the destination is contained within the source data folder.

A common use for a data folder reference variable is to create a copy of a data folder inside a free data folder:

```
DFREF dest = NewFreeDataFolder()
DuplicateDataFolder source, dest     // Copy of source in a free data folder
```

DuplicateDataFolder Destination

Depending on the syntax you use, DuplicateDataFolder may overwrite the data folder specified by destDataFolderSpec or it may copy the source data folder into the data folder specified by destDataFolderSpec. To explain this, we will assume that you have executed these commands:

```
KillDataFolder/Z root:
NewDataFolder/O root:DataFolderA
NewDataFolder/O root:DataFolderA:SubDataFolderA
NewDataFolder/O root:DataFolderB
```

This gives you a data hierarchy like this:

```
root
    DataFolderA
        SubDataFolderA
    DataFolderB
```

The following commands illustrate what DuplicateDataFolder does for a given syntax starting with that data hierarchy.

```
// 1. Literal dest without trailing colon:
// Overwrites DataFolderB with copy of DataFolderA named DataFolderB
DuplicateDataFolder/O=1 root:DataFolderA, root:DataFolderB

// 2. Literal dest with trailing colon:
// Copies DataFolderA into DataFolderB
DuplicateDataFolder/O=1 root:DataFolderA, root:DataFolderB:

// 3. Literal dest with explicit dest child name:
// Copies DataFolderA into DataFolderB with name Child
DuplicateDataFolder/O=1 root:DataFolderA, root:DataFolderB:Child

// 4. DFREF dest without trailing colon:
// Copies DataFolderA into DataFolderB
DFREF dest = root:DataFolderB
DuplicateDataFolder/O=1 root:DataFolderA, dest

// 5. DFREF dest with trailing colon:
// Generates error
DFREF dest = root:DataFolderB
DuplicateDataFolder/O=1 root:DataFolderA, dest:// Error - trailing colon not allowed

// 6. DFREF dest with explicit dest child name:
// Copies DataFolderA into DataFolderB with name Child
DFREF dest = root:DataFolderB
DuplicateDataFolder/O=1 root:DataFolderA, dest:Child
```