There is only one restriction on the dimensions of a wave: when performing a forward FFT on real data, the number of rows must be even. Note, however, that if a given dimension size is a prime number or contains a large prime in its factorization, the speed will be reduced to that of a normal Discrete Fourier Transform (i.e., the number of operations will be on the order of N^2 rather than N*log(N)).

For more information about the FFT, see **Fourier Transforms** on page III-270 and the **FFT** operation on page V-222.

Treating Multidimensional Waves as 1D

Sometimes it is useful to treat a multidimensional wave as if it were 1D. For example, if you want to know the number of NaNs in a 2D wave, you can pass the wave to **WaveStats**, even though WaveStats treats its input as 1D.

In other cases, you need to understand the layout of data in memory in order to treat a multidimensional wave as 1D.

A 2D wave consists of some number of columns. In memory, the data is laid out column-by-column. This is called "column-major order". In column-major order, consecutive elements of a given column are contiguous in memory.

For example, execute:

```
Make/N=(2,2) mat = p + 2*q
Edit mat
```

The wave mat consists of two columns. The first column contains the values 0 and 1. The second column contains the values 2 and 3.

You can pass this wave to an operation or function that is not multidimensional-aware and it will treat the wave as if it were one column containing 0, 1, 2, 3. For an example:

```
Print WaveMax(mat)       // Prints 3
```

Here is an example of using the knowledge of how a multidimensional wave is laid out in memory:

```
Function DemoMDAs1D()
   // Make a 2D wave
   Make/O/N=(5,3) mat = p + 10*q
   Variable numRows = DimSize(mat,0)

   // Find the sum of each column
   Variable numColumns = DimSize(mat,1)
   Make/O/N=(numColumns) Sums
   Sums = sum(mat, p*numRows, (p+1)*numRows-1)
   Edit mat, Sums
End
```

The statement

```
Sums = sum(mat, p* numRows, (p+1)* numRows-1)
```

passes the 2D wave mat to the **sum** function which is not multidimensional-aware. Because sum is not multidimensional-aware, it requires that we formulate the startX and endX parameters treating mat as if it were a 1D wave with the data arranged in column-major order.

You can also treat 3D and 4D waves as 1D. In a 3D wave, the data for layer n+1 follows the data for layer n. In a 4D wave, the data for chunk n+1 follows the data for chunk n.