```
   // Create a wave to hold data folder references returned by Worker.
   // /WAVE specifies the data type of the wave as "wave reference".
   Make/WAVE/N=(numPlanes) ww

   Variable timerRefNum = StartMSTimer

   MultiThread ww= Worker(srcData,p)

   Variable elapsedTime = StopMSTimer(timerRefNum) / 1E6

   Printf "Statement took %g seconds for %d planes\r", elapsedTime, numPlanes

   // At this point, ww holds wave references to 50 M_ImagePlane free waves
   // created by Worker. Each M_ImagePlane holds the extracted and filtered
   // data for one plane of the source 3D wave.

   // Create an output wave named out3D by cloning the first filtered plane
   WAVE w= ww[0]
   Duplicate/O w, out3D

   // Concatenate the remaining filtered planes onto out3D
   Variable i
   for(i=1;i<numPlanes;i+=1)
      WAVE w= ww[i]
      Concatenate {w}, out3D
   endfor

   // Create a 3D output wave by concatenating the filtered planes
   Concatenate/O {ww}, out3D

   // ww holds references to the free waves. By killing ww, we kill
   // the last reference to the free waves which causes them to be
   // automatically deleted.
   KillWaves ww
End
```

To run the demo, execute:

```
Demo(50)
```

Structure Array MultiThread Example

In a preceding example, free data folders were used to hold data processed by threads. A somewhat simpler approach is to use one or more structures to pass input data and to receive output data. The following example uses a single structure for both input and output. An array of these structures stored in a wave ensures that each thread works on its own data. After the calculation, the results are extracted. The net result for this simple example is nothing more than: `dataOutput = sin(p)`.

```
Structure ThreadIOData
   // Input to thread
   double x

   // Output from thread
   double out
EndStructure

Function Demo()
   if (IgorVersion() < 6.36)
      // This example crashes in Igor Pro 6.35 or before
      // because of a bug in StructGet/StructPut
      Abort "Function requires Igor Pro 6.36 or later."
```