**Adding FFT and IFFT operators to gmtmath and grdmath**

**Paul Wessel, Aug 18, 2014**

**1. Overview**

There are lots of times we would like to do mathematical operations in the frequency domain, then convert back to space or time and continue with other operations, or the other way around, or do calculations in both domains. Both **gmtmath** and **grdmath** currently operate only in one domain, but with some changes could become much more powerful modules. Tools like **grdfft** allows only for a tiny selection of preset operators and are mostly useful for filtering, etc. **spectrum1d** allows for no calculations on the spectrum.

**2. Common issues for both tools:**

**a)** We need to add two operators: **FFT** (forwards FFT) and **IFFT** (inverse FFT). The modules will know their state so FFT cannot be called after another FFT, etc. However, there are times when we wish to set things up directly in the frequency domain, do some math, then call IFFT and save the result. We need a way to notify the two modules which domain we start in. I propose we add a new option to both modules:

**–D***mode* with *mode* among **s**|**t**|**f**|**w**

Here, **s** and **t** stand for space or time domain; the only difference between them is one of labeling (e.g., for notifications) and both imply we are in the time domain [Default]. The **f** and **w** stand for frequency or wavenumber and means we are starting in the frequency domain (again, their difference is just for labeling).

**b)** While T (in gmtmath) and X,Y (grdfft) have been used for spatial coordinates, these will return frequencies or wavenumbers in that mode. We could instead insist that users use **F** or **K** (frequency or wavenumber – same thing), and **KX**, **KY, KR** (wavenumbers) instead as these names are still available.

**3. Specific issues for gmtmath**

GMT data tables are normally real-valued only, with one user-selected column designated to hold time (or distance). When a GMT data table is read by **gmtmath** and **FFT** is called, we will double the number of columns and use the new columns to store the imaginary components, and let the duplicate time-column hold the frequency vector. When **IFFT** is called the output is placed in the real columns and the no-longer-used imaginary columns are removed. There are some special cases: (a) If **–Df** mode is in effect (or we have called **FFT** with no matching **IFFT** yet) we assume any input data tables are already complex with all extra columns present. (b) If we call **=** (the output operator) while still in frequency mode then we output all the real and imaginary columns as they are. Because the 1-D FFT function expects a single multiplexed array we may need to duplicate the real and imaginary data columns into a temp vector for the purpose of doing the FFT, then copy the result back into the data table. Some of this may be hidden in the calling sequences.

**4. Specific issues for grdmath**

GMT grids are real-valued only; there is no complex GMT grid, and most of the grid formats could not handle such data (netCDF is one exception, of course). However, when reading programmatically we can tell GMT that a grid is either the real or imaginary part of a complex grid and store both in a single enlarged GMT\_GRID structure. Thus, complex grids in GMT are handled externally using two separate grids but internally (in **grdmath** and others dealing with FFTs) as a single enhanced grid. This has numerous advantages such as any GMT tool can also operate or make plots of the imaginary part, and **grdmath** can convert the complex grid to polar form, etc. Internally, the format feeds naturally into and out of the FFT machinery (we have code in place for the mux and demux). If **–Df** mode is in effect (or we have called **FFT** with no matching **IFFT** yet) we assume any input grids are given in pairs, separated by a comma, i.e., Re-grid.nc,Im-grid.nc. Any of the two names in the pair could be a hyphen (**–**), meaning the component is zero. (b) If we call **=** (the output operator) while still in frequency mode then we output the real and imaginary component grids as they are, separately.

**5. Examples**

a) Read in a complex grid via its two component grids, then take inverse transform and write the result

grdfft –Df real.nc,imag.nc IFFT = model.nc

b) Read in a data table with time and three observable series, remove the means, take the FFT, multiply all by the frequency, take inverse transform and save.

gmtmath mydata.txt DUP MEAN SUB FFT F MUL IFFT = newstuff.txt

**6. Notes:**

1. Programs like Matlab have an **FFTSHIFT** operator as well (reshuffle array or grid layout to place zero frequency in the middle; it is a toggle operator). However, in GMT we will ensure the zero frequency is always in the middle so that frequencies have a natural range from negative Nyquist to +positive Nyquist. This eliminates the need for an explicit **FFTSHIFT**.
2. The common **–N** option for FFT specifications allows for the saving of a raw spectrum as real and imaginary grids. This is a bit different from what happens when using **=** but related and the code will take advantage of that.
3. I decided against adding a **SPECTRUM** operator to mimic what **spectrum1d** and **grdfft –E** does it would (a) kind of eliminate **spectrum1d** from the mix and also be messy since **spectrum1d** has many options about windowing, output values, etc.). Likewise, **grdfft** can do cross-spectra so I think we leave these capabilities where they currently are.