HMI Global Helioseismology Pipeline

Modules:

v2helio – remaps dopplergrams to heliographic coordinates. input is dopplergrams of any resolution, output is maps equally spaced in longitude and sin latitude.

helio2mlat – input from v2helio, performs fourier transform in longitude and transposes data.

qdotprod – input is a series of “images” from helio2mlat, performs projection onto the spherical harmonics. this modules takes us from images to timeseries. output is timeseries, chunked in l.

retile – does what it says. input is timeseries, output is timeseries. changes tiling in time and chunking in l. restricted to one output at a time.

ts_fiddle(_rml) – input is timeseries, assumed to contain a single l. detrends and fills gaps. output is usually timeseries for us, but can also be fourier transforms and power spectra. rml is the initials of a certain tall dane who made improvements.

pkbgn24 – Jesper’s peakbagging module. input is timeseries. module performs fourier transform, writes to disk, passes filename to fortran subroutine. we usually run 10 interations. output is an ascii file containing mode parameters for each l and n it was able to fit.

inv2d.x – not actually a module, just an executable that performs a 2d inversion for internal rotation. input is mode parameters from ascii files and output is internal rotation profile as ascii files. code is in fortran.

we also have a module in place for correcting dopplergrams for distortion and interpolating them to HMI resolution for testing.

Progress

All modules above have been ported to DRMS.

All have been compiled along with their accompanying libraries.

All have been minimally tested: they execute, and produce output in (mostly) good agreement with the corresponding DSDS modules.

JSOC dataseries has been designed for the output of v2helio. could use similar for helio2mlat, but we may simply merge this module with qdotprod. a dataseries has also been designed to hold a timeseries in each record, primekeys are t_start, lmin, max. this handles output of qdotprod; input and output of retile and ts_fiddle; input to pkbgn24. same structure will work for fourier transforms and power spectra.

To Do Right Away

only v2helio has been (somewhat) thoroughly tested and modified to use HMI keywords. remaining modules need more testing. conversion to HMI keywords will be trivial. ts_fiddle_rml has a known discrepancy between 32 and 64 bit compilations (in point of fact we were never able to run this module on lws or lws2).

clean up libraries, make all modules use the same library for functions they have in common. convert use of linpack to lapack. possibly use of blas to cblas? after this step we can put everything into cvs.

write c shell scripts to handle submission of each module to the JSOC cluster. these also go into cvs. run modules in the cluster to measure load and run time for actual processing.

once we measure some times (for reading/writing to disk, compressing/uncompressing) we can decide how we want to store the data. in DSDS everything is mostly floats, except for fd_V which is shorts. how do we want to store data in DRMS?

To Do After That (in no particular order)

at this point modules have more or less the exact functinality they did in DSDS. several of them can be easily generalized and/or improved in efficiency.

convert scripts to perl where appropriate. this will eliminate some of our need for IDL. the rest of that need could probably be removed by using python.

we need an automated method for identifying gaps. a timeseries of a status word giving the various type of gaps can be put in the same type of dataseries as other timeseries. we hope to derive this information from the housekeeping data.

write a script to tie all modules together.

add more people's code to the pipeline. here are some candidates, in decreasing order of feasibility (according to my guess):

- ridge fitting - Cristina
- Johann Reiter's code
- David Salabert's collapsogram code
- Sylvain Korzennik's long timeseries code
- GONG PEAKFIND
- Stuart Jeffries' metrology algorithm

we also need to get sound speed inversions working, as well as finalize how we want to do the rotation inversions.

decide on a standard format for storing results in DRMS.

write some more modules:
- simulate vw_V from HMI dopplergrams
- futz around with time dilation in timeseries

Processing Plan Ideas

run the first part of the pipeline once a day, producing 1 day timeseries with l=0 to l=lmax.

don't keep any intermediate products unless someone specfically asks for them. they can always be remade at any time on demand.

every 36 or 72 days, retile the above timeseries into longer timeseries containing 1 l each. detrend and gapfill these timeseries. save these and discard the raw timeseries? should either of these be archived?

we could have power spectra and/or fourier transforms as standard products, but probably we will make them from the timeseries as people want them. it depends on what code we have in the pipeline. certainly they will not be archived.

every 36 or 72 days, run peakbagging and whatever other algorithms are in the pipeline to produce mode parameters and run inversions on them.

l=0-1500 -> 1,127,251 m's
1day = 1728 records at 50s cadence, 1800 at 48s, 1920 at 45s
storing floats at 48s means 16,232,414,400 bytes per day, say 16GB
36 days -> 576 GB.