Jet Propulsion Laboratory
California Institute of Technology

GIPSY/OASIS (GIPSY) Overview and Under the Hood

Shailen Desai, Da Kuang, and Willy Bertiger

Near Earth Tracking Systems and Applications Groups
Jet Propulsion Laboratory, California Institute of Technology

March 3, 2014

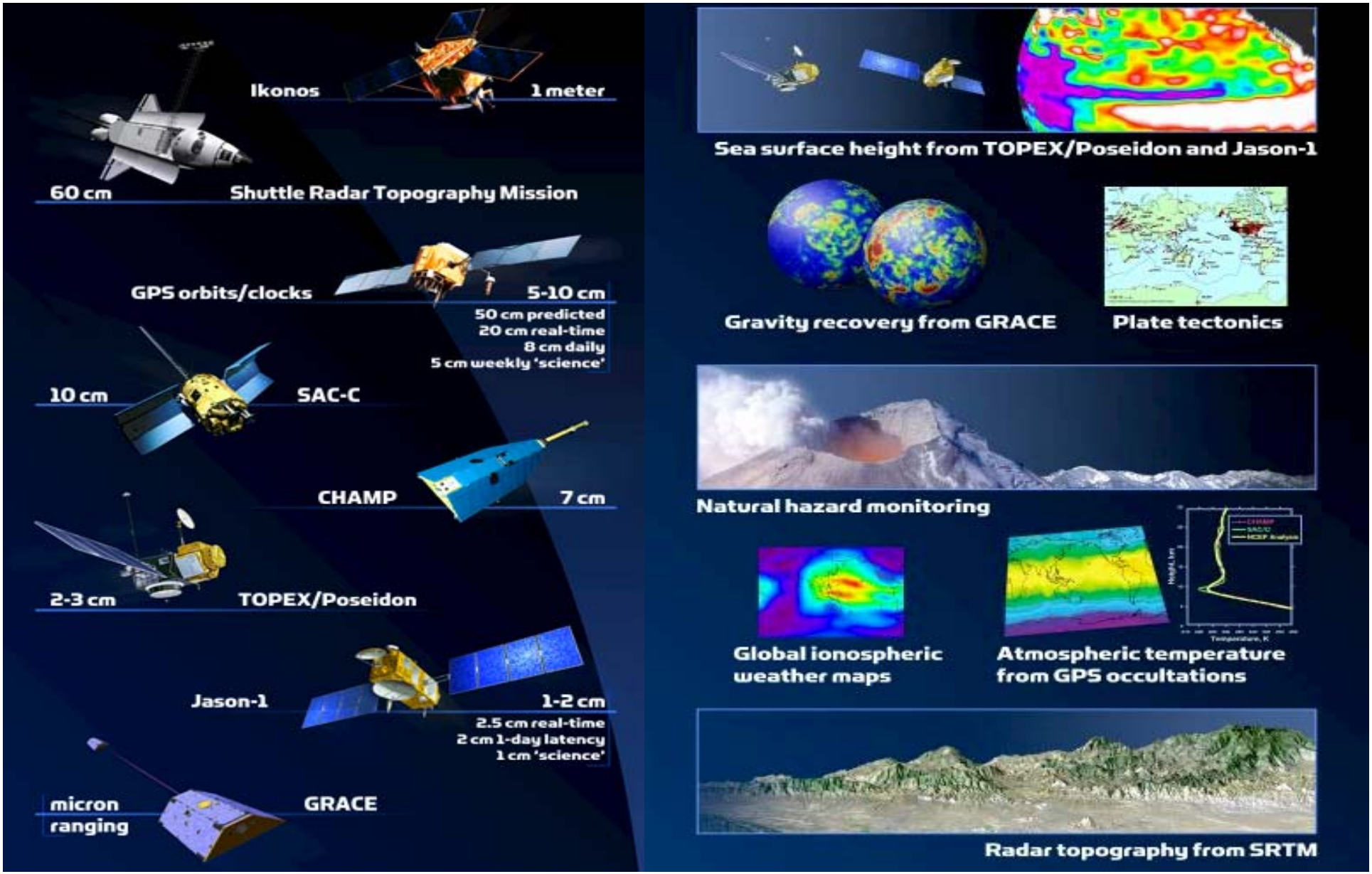
Jet Propulsion Laboratory
California Institute of Technology

History

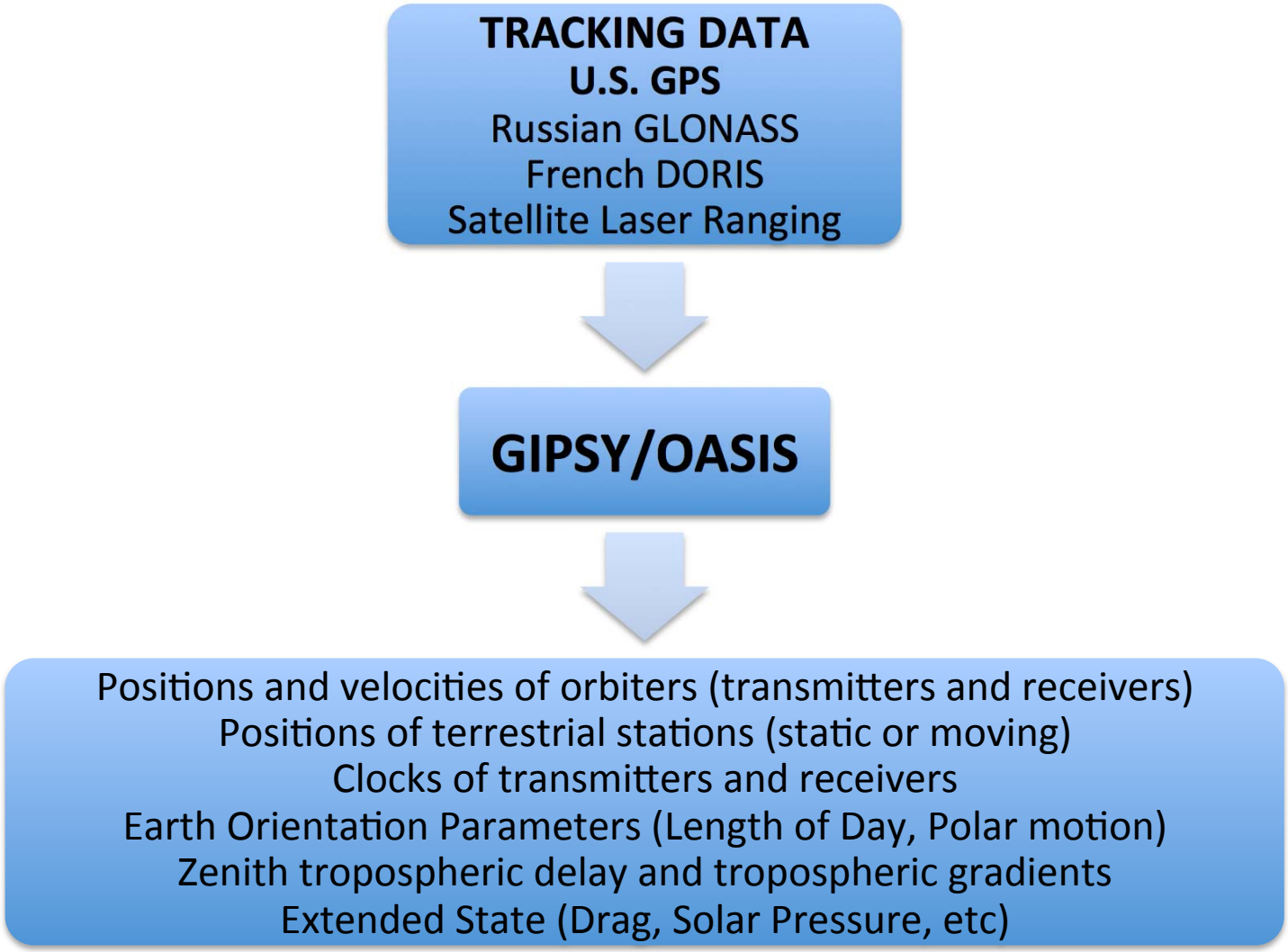
- GIPSY/OASIS:
  - **G**NSS-**I**nferred **P**ositioning **SY**stem and **O**rbit **A**nalysis **SI**mulation **S**oftware
  - Developed by the Jet Propulsion Laboratory (JPL) starting mid-1980's.
    - Builds upon more than 25 years of JPL experience with GPS data analysis.
  - Initially aimed towards supporting GPS-based precise orbit determination of the Topex/Poseidon satellite altimeter mission.
  - Evolved from VLBI software package (MODEST).
  - Hundreds of research and educational users in more than 20 countries.
- JPL
  - FFRDC – Federally Funded Research and Development Center
  - NASA laboratory managed by California Institute of Technology

Orbit Determination in Support of Science and Terrestrial GNSS Applications

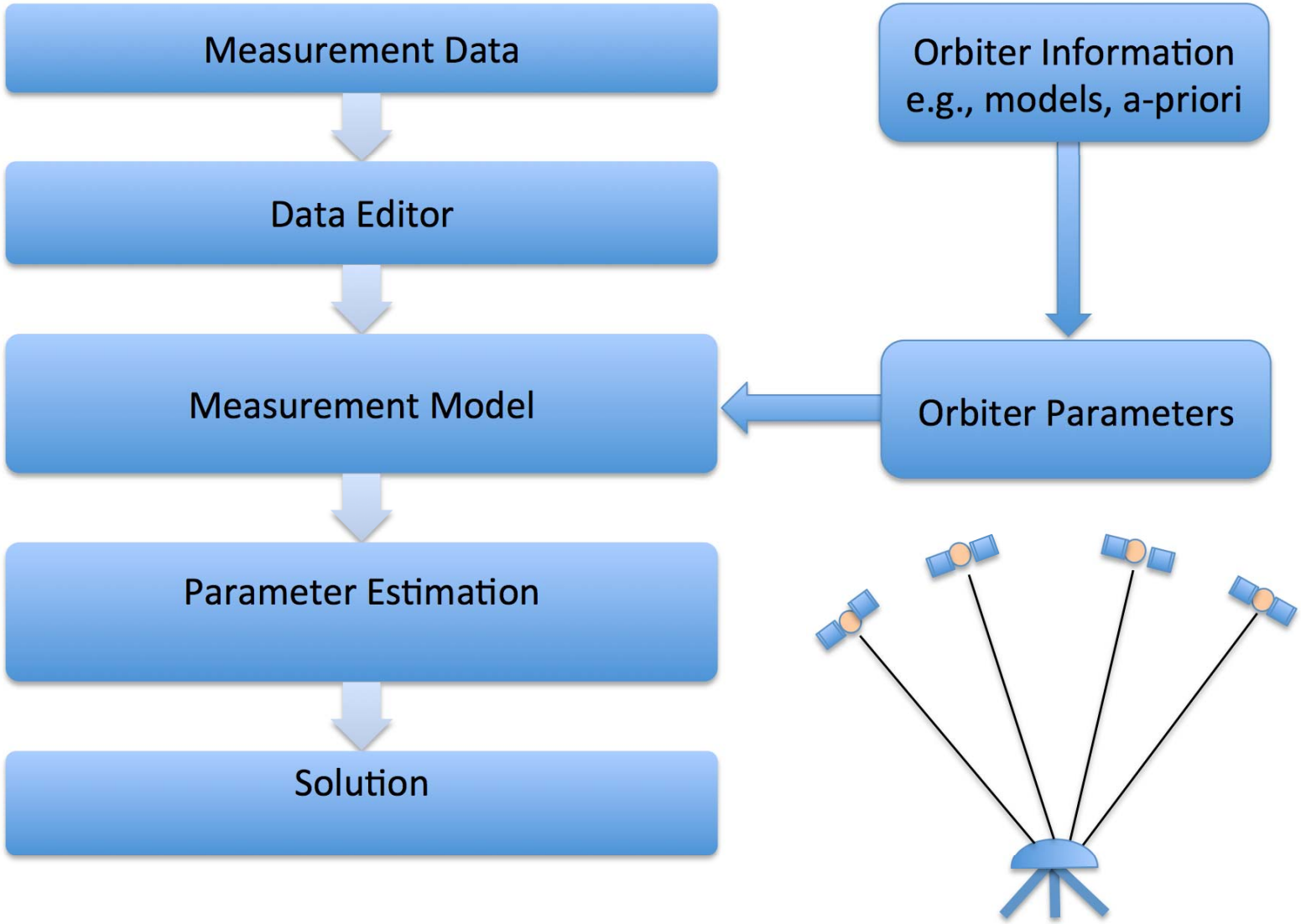
Jet Propulsion Laboratory
California Institute of Technology



GIPSY Input and Output



Typical Processing Flow for GNSS-Based Terrestrial Positioning



GNSS Observables

- GNSS satellites provide two types of observables:
  - Pseudorange (e.g., P1, P2) with accuracy of 30-100 cm.
  - Carrier phase (e.g., L1, L2) with accuracy of ~1 cm or better.
    - ≈ Range biased by an unknown integer number of cycles, i.e., phase biases.
  - Precision geodetic GNSS-based positioning leverages phase measurement accuracy.
    - Single receiver positioning typically estimates phase biases as real values.
    - Improved positioning accuracy realized through "ambiguity resolution":
      - Resolving the integer number of cycles in the phase measurements.
      - Traditionally performed using double differences between measurements.
        - » Two receivers with two commonly viewed GPS satellites.
  - Usually use dual-frequency combinations of range and phase as observable to eliminate first order ionospheric delays in measurements.
    - PC = 2.54*P1 – 1.54*P2
    - LC = 2.54*L1 – 1.54*L2

GNSS Clock Biases and Atmospheric Delay

- Range and phase measurements based on difference between time stamp of transmission and time stamp at reception.
  - Precise positioning needs to account for errors in clocks of transmitters and receivers.
  - Transmitter clock usually provided by orbit/clock product for satellites.
  - Minimum of 4 satellites in view needed to resolve position and clock of receiver.
- The range and phase measurements are "delayed" by the neutral atmosphere (i.e., Dry and wet troposphere) and ionosphere.
  - Precise positioning requires that these effects be modeled or estimated from the data.
  - Ionospheric delays eliminated (to first order) by using dual-frequency data.
    - Ionospheric models required when using single frequency data.

GIPSY Parameter Types

- Parameters can be estimated as constant or time-varying piecewise constants with user-specified update time:
  - Constant
    - Single value over entire data window
  - White noise process
    - Time series of independent Gaussian random values.
    - Neighboring values are uncorrelated.
  - Random walk process
    - Time series of values with white noise update to value at previous time step.
    - Neighboring values are correlated.
  - First order Gauss-Markov process
    - Time series of values, each correlated with neighboring value by a correlation time and steady-state standard deviation.
    - Neighboring values are correlated.
    - White noise and random walk are the two limits of Gauss-Markov.

GIPSY Capabilities

- Measurement type
  - Undifferenced measurements
    - Pseudorange, carrier phase, dual frequency, single frequency, doppler.
- Precise centimeter-level GNSS positioning and timing:
  - Space platforms (GNSS constellations and low-Earth orbiters)
    - Precise orbit determination (POD)
  - **Precise Point Positioning**
    - **Terrestrial (ground) stations, static or moving (kinematic)**
    - Airplanes
  - Network Positioning
  - Time transfer
- Simulation capabilities
  - Satellite orbital motion simulation
  - GPS measurement data simulation

GIPSY/OASIS – First Stop on the Web

- **https://gipsy-oasis.jpl.nasa.gov/**

Current and Upcoming GIPSY Versions

- **Refer to website for current release.**
- Version 6.3 is current release.
  - Addition of antenna thrust models for GPS satellites.
  - Updated solar radiation pressure force models for GPS satellites (GSPM13).
  - Added GPT2 troposphere models.
  - Add ocean load pole tide displacement model.
- Version 6.4 anticipated in 2015.

Licensing

- All GIPSY users should maintain an active license.
  - Typical research license duration is 4 years.
  - Helps justify continued NASA support and development.
  - Provides continued user support.
    - GIPSY Forum, User Group meetings, classes, announcements.
    - Ongoing software updates.
      - Typically, at least 1 release per year.
- Processing time:
  - Allow 6-12 weeks for processing (new or renewals).
    - **Apply for renewal before current license expires.**
- Where?
  - GIPSY website
  - https://gipsy-oasis.jpl.nasa.gov

User Support Through GIPSY Website

- Software License Request and Download
  - Request GIPSY License
  - Download software (with approved license)
- Forum
  - **Users help each other.**
  - JPL monitors discussion and intervenes when necessary.
  - Post questions to forum so all users can benefit from user or JPL response.
    - Typical response time: < 5 working days.
- Documentation
  - Online training videos.
  - Release notes.
  - Simple software patches.
  - Presentations from User Group Meetings and classes.
    - Typically 1 user group meeting/year at Fall American Geophysical Union Meeting.

GIPSY System Requirements

- **Always refer to release notes for each release.**

Operating System	**Linux** **Most extensively tested on Red Hat Enterprise 4 and 5**
	Some testing on Ubuntu and Fedora, but supported on best efforts only.
Memory	**2 GB**
Third Party Software	**Perl 5.8.8**
	Python 2.4.1, 2.4.3, 2.6, or 2.7
	Numpy 1.3.0, 1.4.1, or 1.5.1
	Tcsh 6.12 or newer
	Gnuplot 4.0
	Ncompress 4.2.4
	teqc (Recommended from UNAVCO)
	convert (only needed to generate gif plots)

Installation

- **Always refer to release notes for each release.**
- Following uses GIPSY 6.4 as an example, ***path_exe*** as user-defined location of software installation, ***path_auxdata*** as user-defined location of auxiliary data installation.
  - ***path_exe*** and ***path_auxdata*** could be the same.
- Generate and send to JPL a public key.
  - JPL will send encrypted file with password for download of software.
- Download two gzipped tarballs:
  - Software (GOA): e.g., goa-6.3-RHEL5-64.tgz
  - Auxiliary Data (GOA_VAR): e.g., goa-var.tgz
- Extract auxiliary data tarball to directory of choice (e.g. *path_auxdata*/goa-var)
  - Make sure this directory is read and writeable
- Extract software tarball to installation directory of choice (e.g. *path_exe*/goa-6.3)
  - Rootless GIPSY since version 5.X and higher.
  - Make this directory read-only.

Setting Up GIPSY Paths

- Following uses GIPSY 6.3 as an example.
- Define GIPSY paths through environment variables in the provided "rc" script:
  - For C-shell: *path_exe*/goa-6.3/rc_gipsy.csh
  - For Bash: *path_exe*/goa-6.3/rc_gipsy.sh
  - Manually edit rc_gipsy.csh (or .sh) in installation directory to set two envronment variables.

    setenv GOA *path_exe*/goa-6.3

    setenv GOA_VAR *path_auxdata*/goa-var
  - Or, run following command to set these paths in the rc_gipsy file of choice.

_path_exe_/goa-6.3/admin/fix_rc_gipsy.py –g _path_exe_/goa-6.3 –v _path_auxdata_/goa-var

- **Source the rc script of choice (C-shell or Bash) before running GIPSY.**
  - E.g., For C-shell could include the following line in your .cshrc file:

    source _path_exe_/goa-6.3/rc_gipsy.csh
  - Verify paths for two environment variables exist, e.g., use the **env** Unix command.
    - Make sure **GOA** and **GOA_VAR** environment variables are properly set to point to the GIPSY executable and auxiliary data locations.

Updating GIPSY's Auxilliary Data

- Keep auxilliary files in **GOA_VAR** updated to process current data.
  - E.g. set up a cron to run once a day.
  - Cron should include the following command, assuming environment variable GOA (pointing to GIPSY software) exists.

    $GOA/crons/update_gipsy_files.py
  - This script updates some of the auxilliary data through a JPL server.
    - CA-P bias table.
    - GPS Yaw rates
    - GPS Yaw biases
    - Earth orientation parameter tables
    - Leap seconds table
    - Receiver types table
    - PRN to GPS conversion tables.
    - Antenna calibrations.

Running Verify Tests

- Verify tests provide some basic examples to test for expected results.
  - Includes basic point positioning, precise orbit determination, and utility applications.
- Source start-up file of choice (C-shell or Bash):

  source *path_exe*/goa-6.3/rc_gipsy.csh

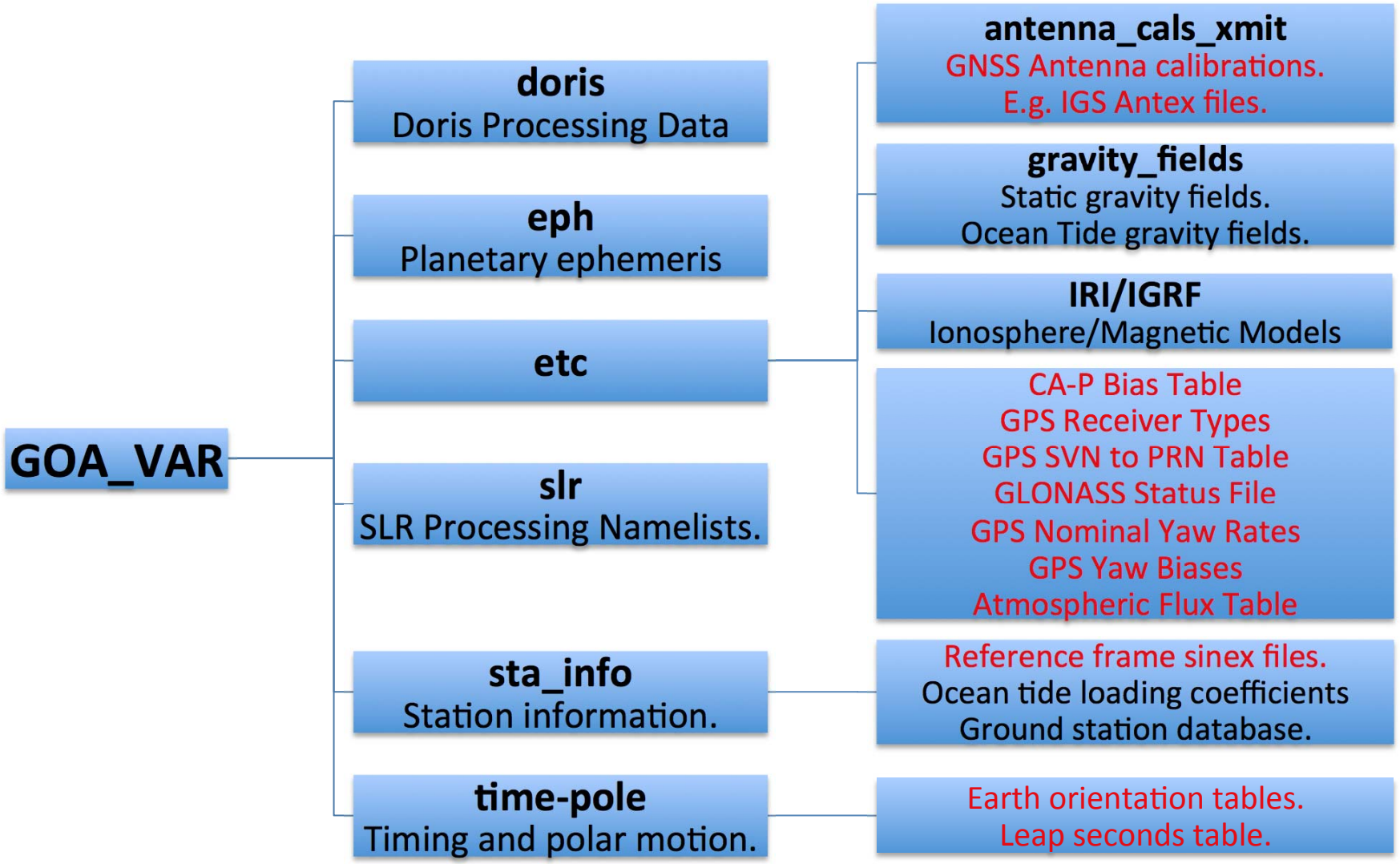
  (or source path_exe/goa-6.3/rc_gipsy.sh)
- Current working directory should not have an existing "verify" directory
- Extract verify tarball.

  tar xvfz $GOA/verify.tar.gz
- Run the verify tests (can take up to 15 minutes):

  cd verify

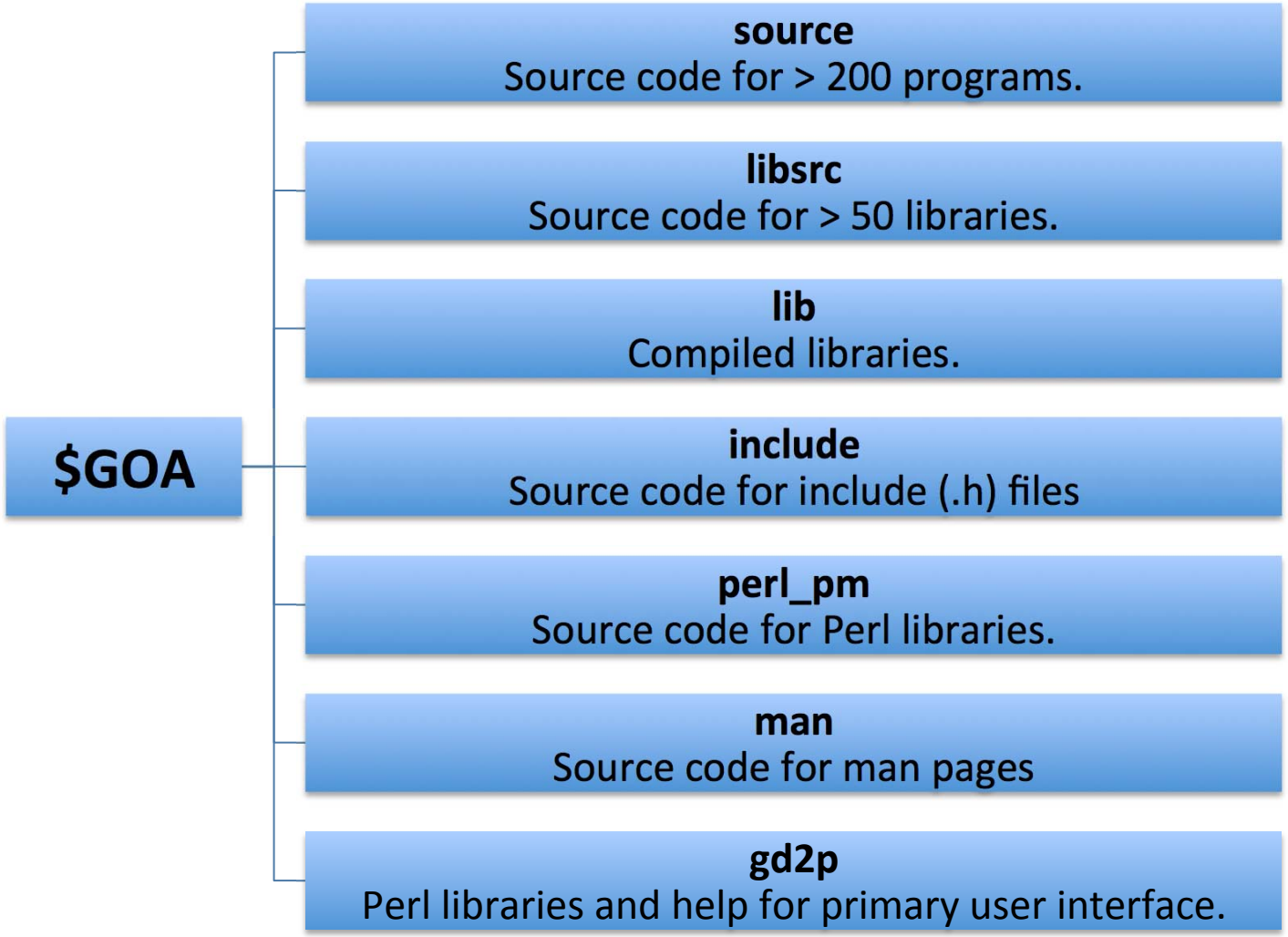
  ./verify.py
- Make sure that all tests pass ("FAIL" indicates failure).
- Verify tests are self-contained, except one that tests ftp connection to JPL.
  - Decide importance of ftp connection. (e.g. to update GOA_VAR)

GOA_VAR Directory Structure



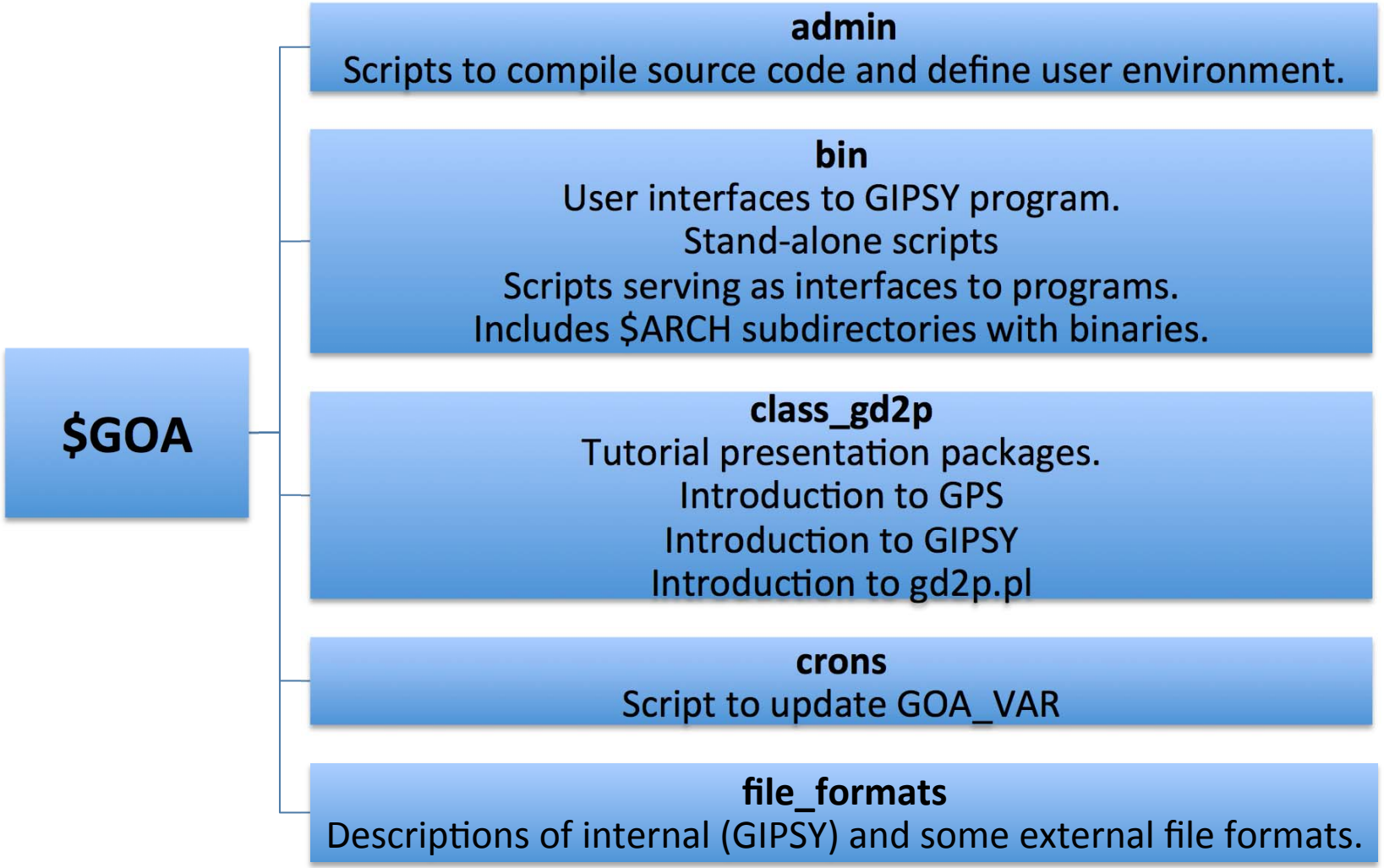
Files updated by cron.
Needed for current processing.

GOA Source Code Structure

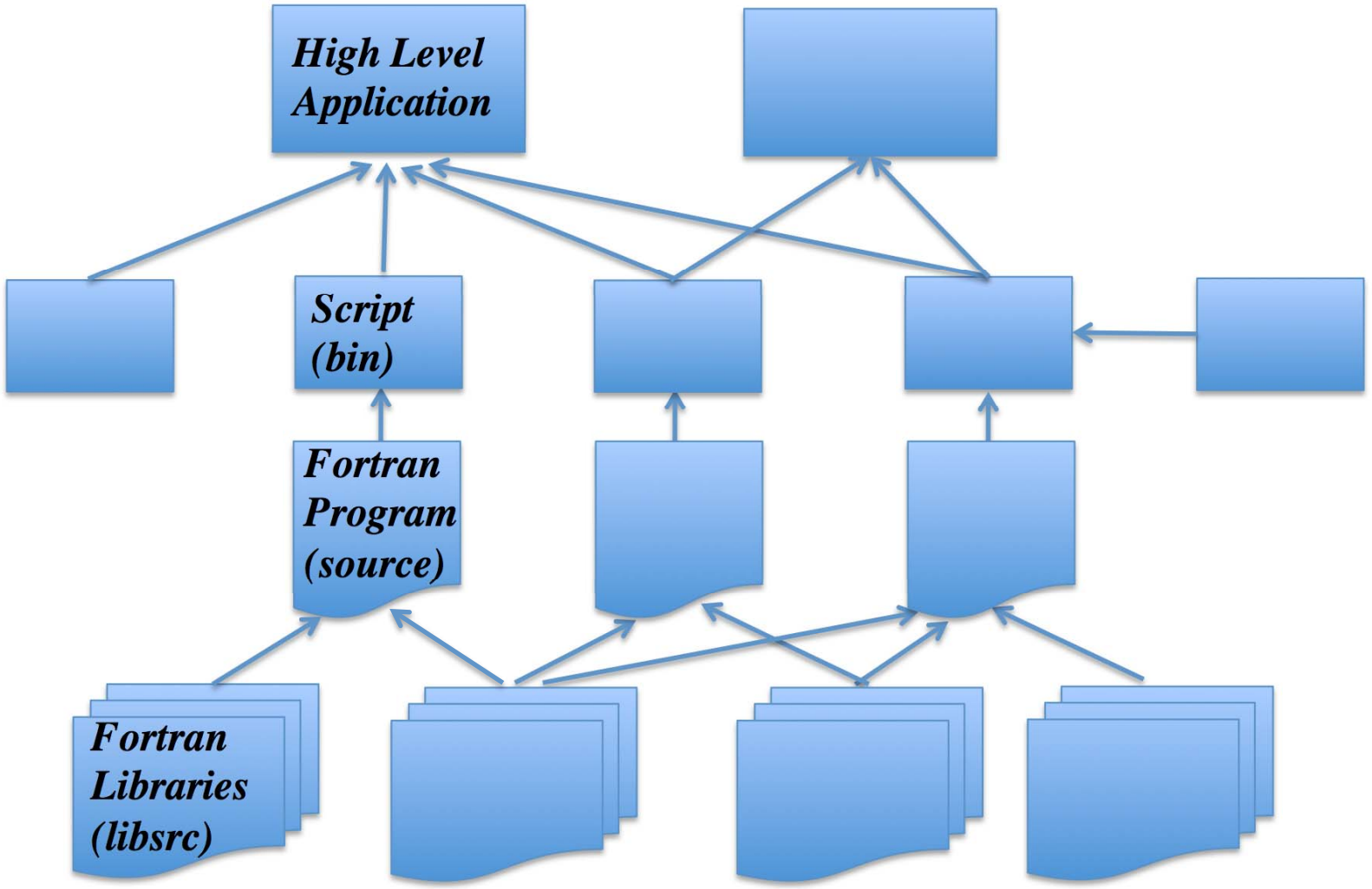


NOTE: Fortran source code not provided even though directories exist.

GOA User Tools



GOA Program Structure



$GOA/file_formats

- Defines format of internal and external files.

File	Type	Description
JPL_GPS_Products	ASCII	JPL's orbit and clock products.
rinex	ASCII	Receiver independent exchange format for GNSS data.
qmfile	BINARY	Quick measurement, contains tracking data.
pos_goa	ASCII	Position and Velocity Table
tdp/TDPfile	ASCII	Time Dependent Parameters Table
antenna_cal	ASCII	Transmitter and Receiver Antenna Calibrations (xyz files)
tp_array	ASCII	Earth Orientation Parameter "tpnml" namelist
geop	ASCII	Earth Orientation Parameter table
wlpblist	ASCII	Widelane and phase bias table
ocnld_formats	ASCII	Ocean load tide coefficients
sta_cov	ASCII	Station positions and covariance matrix

NAVIO Files

- NAVIO = NAVigation Input and Output
  - Binary format to exchange large volume of I/O beween various modules.
  - $GOA/libsrc/navio – Library to handle these files
  - Utilities to manipulate NAVIO files.
    - **niodump** – Dump to standard system output file.
    - **nio2text** – Dump to text file.
    - **niocomp** – Compares two NAVIO files.

Module	NAVIO File	Description
qregres	regres	Observation (regression) equations.
filter	accume	Information matrix and scaled sensitivity matrix.
filter	smooth	Smoothing coefficients
smapper	smapper_solution	Smoothed solution (smsol)
smapper	smapper_covariance	Smoothed covariance (smcov)
oi	nio	Orbit states and partials

Online GIPSY Help

- **Command line help (recommended)**
  - Access: 'command', '*command –h*', '*command –H*', '*command –help*', '*command help*'
    - Depends on module. Not fully consistent across all modules.
  - Examples: '**gd2p.pl**', '**gd2p.pl –h**', '**yymmmdd2sec –H**', '**antex2xyz.py –help**'
- Man pages
  - Access: '*man command*' or '*man –k keyword | grep 1g*'
  - Examples: '**man gd2p.pl**', '**man oi**', '**man filter**', '**man –k gd2p | grep 1g**'
- Source code area text files
  - Users guide: $GOA/source/*module*/USERSGUIDE.TXT
    - Examples: *module* = filter, prefilter, preprefilter, smapper
  - Namelist descriptions: $GOA/source/*module*/*module*.nml
    - Examples: *module* = qregres, trajedy (e.g., $GOA/source/qregres/qregres.nml)
  - Include files: $GOA/libsrc/*library*/*.inc
    - Examples: *library* = oi (e.g., $GOA/libsrc/oi/*.inc)
      - E.g., gravity.inc, tides.inc, solar_press.inc

Typical GIPSY Usage

- Unix/Linux command lines:
  - E.g., *command –i input_file –n namelist_file –p option –o outputfile*
  - *input_file* – Name of input data file to be processed.
    - Some programs use environment variables to define input files.
  - *namelist_file* – Name of FORTRAN namelist file
    - Defines processing parameters, locations of models, etc.
  - *option* – Options for program, e.g. method, model, file
    - Most programs have more than one processing option.
  - *output_file* – Name of output file (product of program).
    - Some programs print output to standard output.

Some Useful Utilities for Tabulated Data

- **cl** – Script to manipulate tabulated data.

  cl –help

  – Example: Extract columns 1 and 4 of leap seconds file:

  cat $GOA_VAR/time-pole/LEAPSECS | cl 1 4

- **stats** – Compute statistics of a column of data.

  – Reports number, mean, standard deviation, minimum, maximum, root-mean-square (RMS) of a set of numbers from STDIN.

  man stats

  – Example: Compute statistics on leap seconds since 1979.

  cat $GOA_VAR/time-pole/LEAPSECS | cl 4 | stats

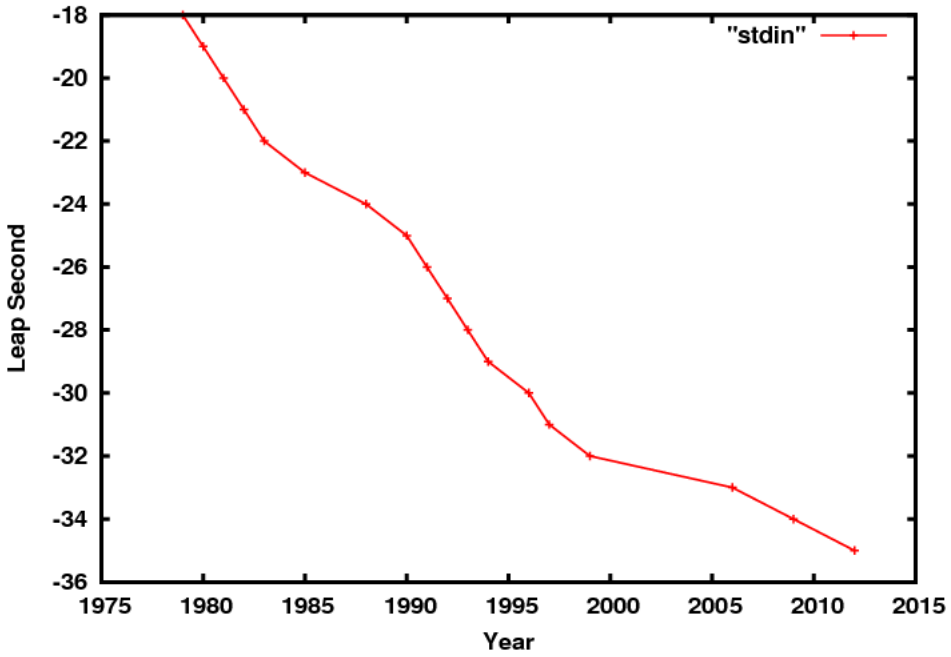
```
18  -26.5000000000000   5.33853912601566   -35.0000000000000   -18.0000000000000   27.0030862433661
```

Plotting Data Using GIPSY's gnup

- **gnup** – Script to plot data from unix command line.
  - Uses open source **gnuplot**.

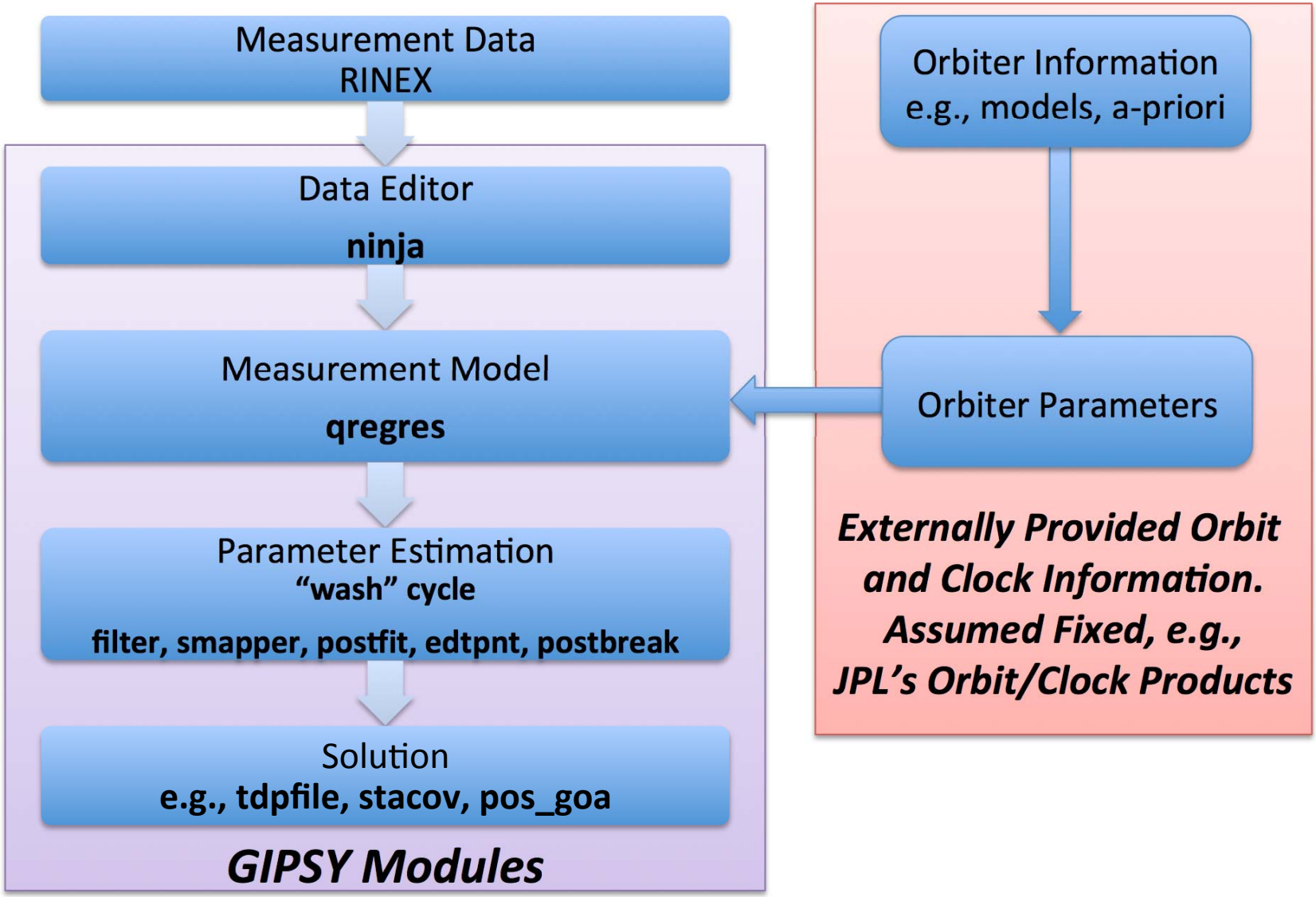
  gnup –help
- Example: Plot time series of leap seconds by year:

  cat $GOA_VAR/time-pole/LEAPSECS | cl 1 4 | awk '{printf "%d %d\n", substr($1,8,4), $2}' | gnup -lp -xl 'Year' -yl 'Leap Second'



- **Note: Could use gnup to generate a gif file with the plot using –gif option.**
  - **e.g. –gif leapsec.gif**
- **"convert" utility must be installed to generate gif file.**

Typical GIPSY Flow for GNSS-Based Terrestrial Positioning



Features of JPL's Orbit and Clock Products

- **Includes information to enable single receiver ambiguity resolution with the GIPSY-OASIS software.**
  - **Wide-Lane Phase Bias ("wlpb") files.**
- All products span 30-hours.
  - Ultra-Rapid is moving window updated (overwritten) every hour.
  - Rapid and Final centered at noon of each day.
- Provided in formats native to GIPSY (e.g. pos_goa in Earth-Center-Earth-Fixed).
- Orbit estimates provided with respect to the center of mass of entire Earth system (solid Earth, atmosphere, oceans).
  - **Recommend using ocean load tide corrections with respect to combined center of mass of solid Earth and oceans.**
- All three products are automatically fetched with gd2p.pl.
  - See –orb_clk flag. (ultra, qlR, fllinnR)
- Presently, JPL orbit/clock products include GPS only.
  - On-going work to include other GNSS constellations.

JPL's GPS Orbit and Clock Products

Product	Latency	3-D RMS Accuracy (cm)	Product Location (anonymous ftp)	High-Rate (30-second)Clock Product
Ultra-Rapid	< 2 hours	5	*/Ultra	No
Rapid	Next-day (16:00 UTC)	3.5	*/Rapid	Yes
Final	< 14 days	2.5	*/Final	Yes

* = ftp://sideshow.jpl.nasa.gov/pub/JPL_GPS_Products

- **All three product families include availability of "wlpb" files to enable single-receiver ambiguity resolution.**
  - Benefit of using GIPSY format products instead of IGS format products.
- Final products span August 16, 1992-present.
  - High-rate clock products only available from May 5, 2000.
    - Selective Availability (SA) discontinuation.

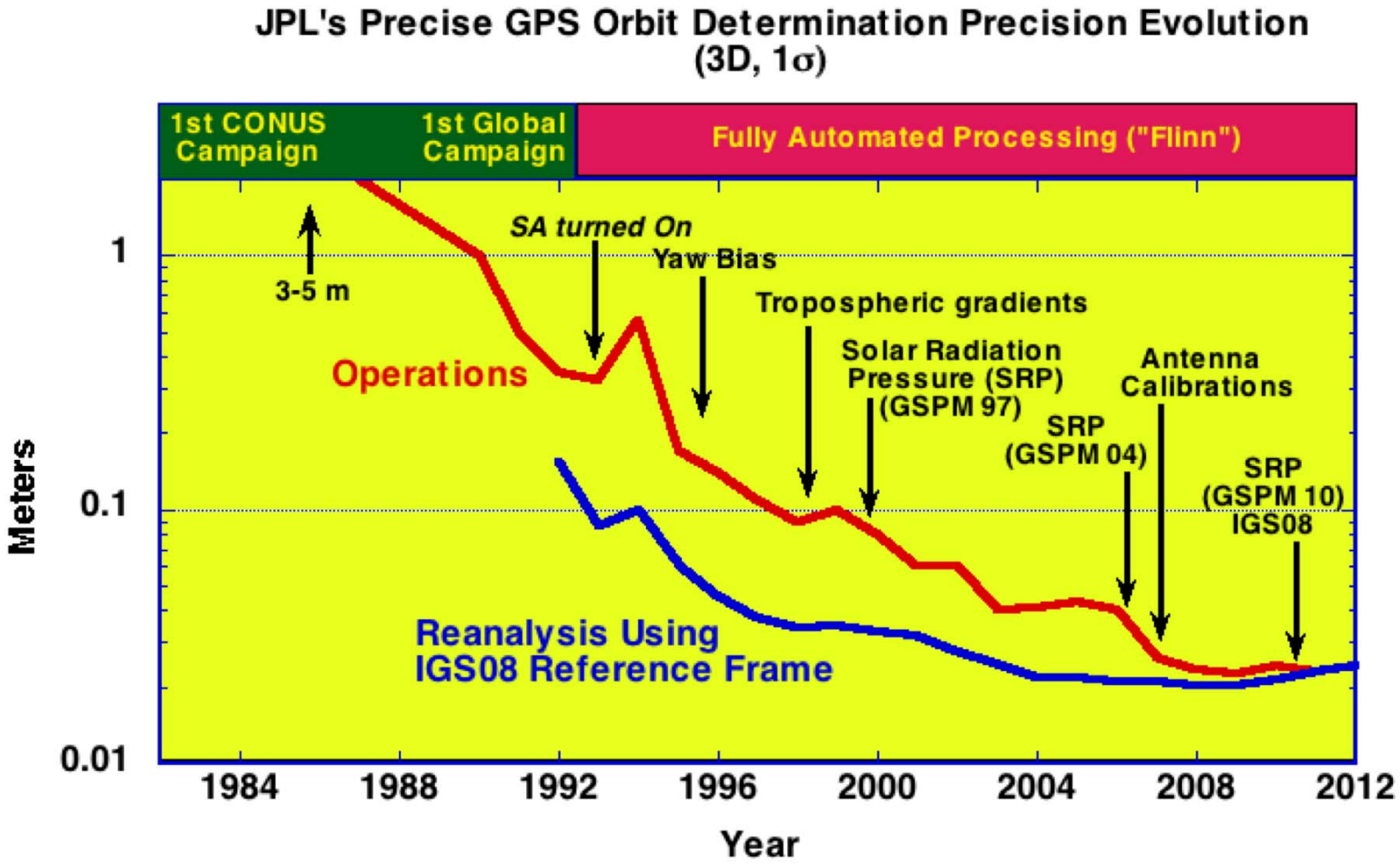
Fiducial Orbit and Clock Product Files

Description	Filename	Product Family
Fiducial Orbits	yyyy-mm-dd.pos.gz	ALL Products Ultra-Rapid Rapid Final **Fiducial means that the products are defined in the reference frame identified in the .frame file.** **(No high-rate clock for Ultra-Rapid)**
Fiducial Clocks (5-minute)	yyyy-mm-dd.tdp.gz	
Fiducial Clocks – High Rate (30-sec)	yyyy-mm-dd_hr.tdp.gz	
Fiducial Earth Orientation	yyyy-mm-dd.eo.gz	
Fiducial "WLPB"	yyyy-mm-dd.wlpb.gz	
Shadow	yyyy-mm-dd.shad.gz	
Frame	yyyy-mm-dd.frame.gz	
Transmitter Antenna Calibration	yyyy-mm-dd.ant.gz	

No-Net Rotation (NNR) and Non-Fiducial (NF) Orbit and Clock Product Files for Finals

Description	Filename	Description
No Net Rotation Orbits	yyyy-mm-dd_nnr.pos.gz	Final Products Only **No-Net Rotation (*_nnr) means no-net rotation with respect to the reference frame. Rotations in Helmert transformation are forced to zero, while translation and scale are free.** **Translation and scale will be similar to *_nf products.** **Expected availability: mid-2014.**
No Net Rotation Clocks (5-minutes)	yyyy-mm-dd_nnr.tdp.gz	
No Net Rotation Clocks - Hi Rate (30-sec)	yyyy-mm-dd_nnr_hr.tdp.gz	
No Net Rotation Earth Orientationn	yyyy-mm-dd_nnr.eo.gz	
No Net Rotation "WLPB"	yyyy-mm-dd_nnr.wlpb.gz	
No Net Rotation Helmert Transformation	yyyy-mm-dd_nnr.x.gz	
Fiducial Free Orbits	yyyy-mm-dd_nf.pos.gz	Final Products Only **Non-fiducial (*_nf) , or Fiducial-free, means no tie to the reference frame. GPS satellites determine 'frame of the day'.**
Fiducial Free Clocks (5-minute)	yyyy-mm-dd_nf.tdp.gz	
Fiducial Free Clocks – High Rate (30-sec)	yyyy-mm-dd_nf_hr.tdp.gz	
Fiducial Free Earth Orientation	yyyy-mm-dd_nf.eo.gz	
Fiducial Fee "WLPB"	yyyy-mm-dd_nf.wlpb.gz	
Helmert Tranformation	yyyy-mm-dd.x.gz	

Evolution of JPL's "Final" GPS Orbit and Clock Products



- Solar radiation pressure likely to be limiting error source.
- IGS08 is reanalysis released in early 2012. (A second IGS08 reanalysis is being performed in 2014).

Defining a Base Working Directory for Examples

- In this package, examples assume a working directory defined by the user with **absolute** path name: ***WORKDIR***.
- Examples will be relative to this user-defined working directory.

  mkdir *WORKDIR*

  cd *WORKDIR*

Tool to Fetch JPL Orbit/Clock Products

- goa_prod_ftp.pl – Fetches specified JPL orbit/clock products.
  - Example: Pick up "Final" orbits and clocks, including high-rate clocks (-hr option) and place in directory "jplorbclk".

mkdir jplorbclk

cd jplorbclk

goa_prod_ftp.pl -d 2010-01-10 -s flinnR –hr

  - Or copy output from example from $GOA:

cp $GOA/class_gd2p/orbits_forclass/* .

ls

2010-01-10.ant.gz 2010-01-10.frame.gz 2010-01-10.pos.gz 2010-01-10.tdp.gz
2010-01-10.eo.gz 2010-01-10_hr.tdp.gz 2010-01-10.shad.gz 2010-01-10.wlpb.gz

WARNING: For following examples use products provided with release (orbits_forclass). Revised products are likely to be released at the ftp site in 2014.

Antenna Calibrations Used to Generate Products

- The exact transmitter and receiver antenna calibrations used to generate the products are explicitly defined in the .ant file.
  - SAT line indicates transmitter calibration file used to generate orbit/clock product, and location on JPL server.
  - GRN line indicates receiver calibration file used to generate orbit/clock product, and location on JPL server.

zcat 2010-01-10.ant.gz

SAT igs08_1604.xyz ftp://sideshow.jpl.nasa.gov/pub/gipsy_files/gipsy_params/antenna_cals_xmit/igs08_1604.xyz.gz

GRN igs08_1604.atx ftp://sideshow.jpl.nasa.gov/pub/gipsy_files/gipsy_params/antenna_cals_xmit/igs08_1604.atx.gz

– Best positioning achieved by using consistent receiver calibration.
  - Primary consideration for consistency is reference frame identifier (e.g., igs08) in the SAT/GRN line.
  - Version numbers (e.g., 1604) usually means addition of new launches or new (usually not revised) antenna calibrations in same reference frame.

Handling IGS Orbit and Clock Products

- igs_orbits_clks_fetch.pl – Fetches IGS combined orbits/clocks.
  - Example: Fetch final orbits and clocks and place in directory igsorbclk.

```
cd ..
mkdir igsorbclk
cd igsorbclk
igs_orbits_clks_fetch.pl -b 2010-01-10 -e 2010-01-10
ls
igs15660.clk.Z   igs15660.sp3.Z
```

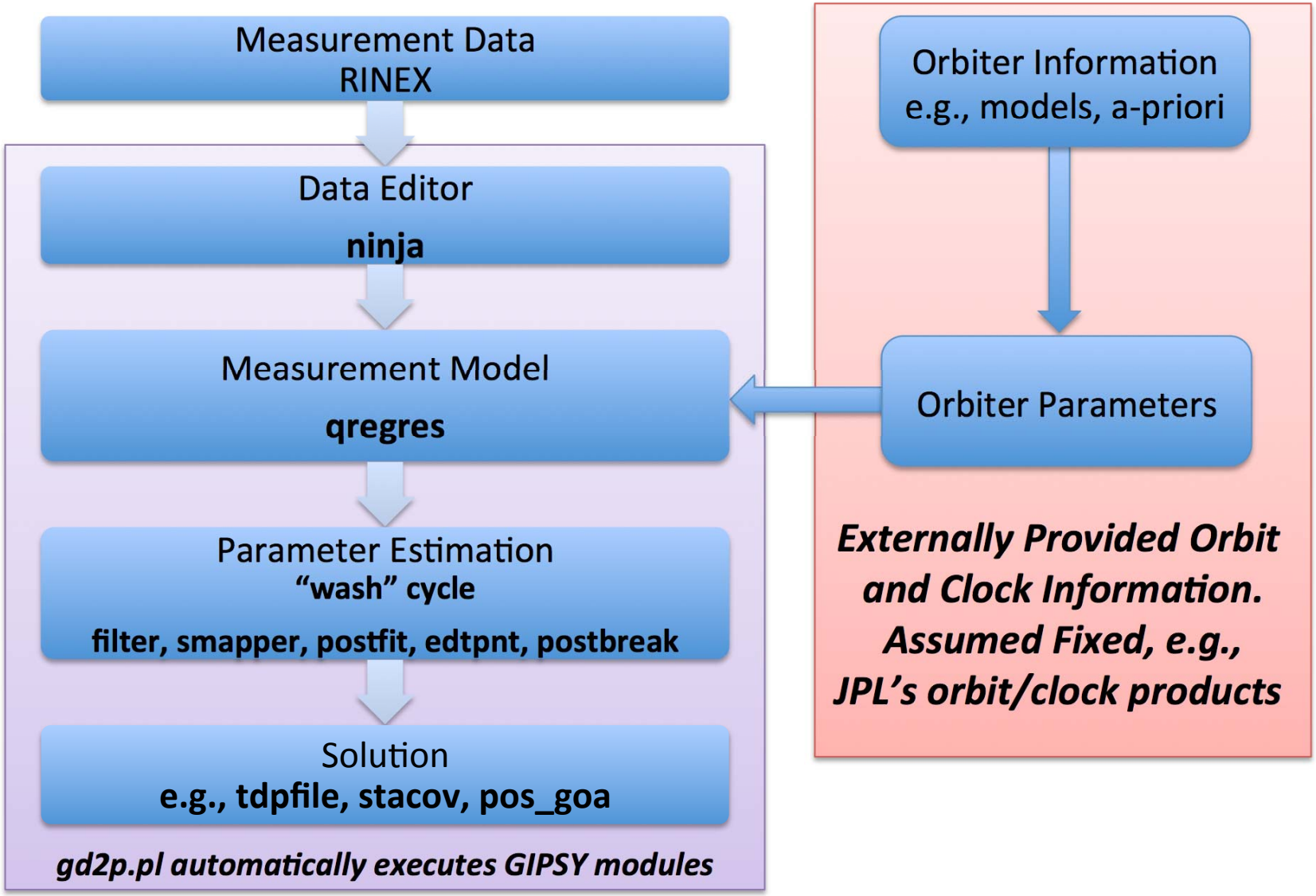
- igs2goa.pl – Convert IGS format products to native GIPSY formats.
  - Assumes IGS format products in current directory.
  - Some care needed here. Scripts may not correctly reflect transmitter antenna calibrations used to generate products.

```
igs2goa.pl -b 2010-01-10 00:00 -e 2010-01-10 23:55
ls
igs15660.clk.Z    igs15660.sp3.Z    pos    tdp_clk_yaw    tpnml
```

What is gd2p.pl?

- gd2p – GNSS Data to Position
  - Recommended high level GIPSY interface for processing data from a single GNSS receiver.
  - Simplifies positioning.
  - Captures typical processing flow, including automated fetching of required transmitter orbit and clock products if not using local database.
    - Examples shown in this package use the database of products generated earlier for 2010-01-10 in **jplorbclk** directory.
  - Flexible user-defined processing control.
  - Better error detection.
  - Static and kinematic point positioning.
  - Precise orbit determination.

Typical GIPSY Flow for GNSS-Based Terrestrial Positioning Using gd2p



gd2p Example 1

- Simple Example: Use 5-minute GPS data and JPL's Final (fiducial) GPS orbits and clocks to estimate static station position, troposphere, receiver clock.

  cd ..

  mkdir example1

  cd example1

  (gd2p.pl –i $GOA/class_gd2p/gol20100.10o –n GOL2 –r 300 –type s –d 2010-01-10 –orb_clk "flinnR *BASEPATH*/jplorbclk" > gd2p.log) >& gd2p.err

  – ***BASEPATH*** = Absolute path to location of previously created work directory with sub-directory (jplorbclk) containing JPL orbit/clock products for 2010-01-10.

  – Standard output (STDOUT) sent to gd2p.log
    - Provides log of each of GIPSY programs executed, and how (i.e. options)

  – Standard error (STDERR) sent to gd2p.err
    - Catches errors from the run.

  – Creates executable script, **run_again**, to exactly reproduce run.
    - Can either edit this file, or copy and modify to re-run with different settings.

Example 1: Checking for errors

- Many errors are caught in “standard error”.
  - **gd2p.err** in this case

cat gd2p.err
Don't forget that this wash script modifies the rgfile

  - This is a warning that gd2p modified the regres file (**rgfile**)
    - regres file contains measurement equations.
    - Useful if using gd2p’s start_at features.
    - Post-fit outliers were marked in the regres file, by setting data weights to negative.
      - Can be restored from negative to positive with **rgclean**. (**man rgclean**)
  - No obvious run-time errors.
    - Solution might still be problematic.

Example 1: Checking Post-Fit Residuals

- RMS of post-fit residuals, in km, are in **Postfit.sum**

cat Postfit.sum

start time	stop time	type	user	postfit sigma	npts	outliers
10JAN09 23:59:45	10JAN10 23:54:45	LC	GOL2	6.6321E-06	2170	0
10JAN09 23:59:45	10JAN10 23:54:45	PC	GOL2	3.7481E-04	2170	0

  - LC = Dual-frequency ionosphere-free phase.
    - Typical RMS of post-fit = 0.5-1 cm.
  - PC = Dual-frequency ionosphere-free range.
    - Typical RMS of post-fit = 30-80 cm.
  - Run is probably OK.

Example 1: Station Position

- Station position solution in **tdp_final.**
  - See time dependent parameter file description ($GOA/file_formats/tdp).
  - Column 2 contains nominal (first guess provided to gd2p).
  - Column 3 is estimate+nominal.
- Extracting the station position solution (units are km).

grep 'STA [XYZ]' tdp_final

```
316353600.0000   3676.97640700000    3676.97641962698    2.112E-06  STA Z   GOL2
316353600.0000  -4641.38532100000   -4641.38528882247    2.737E-06  STA Y   GOL2
316353600.0000  -2353.61442100000   -2353.61439370980    1.944E-06  STA X   GOL2
```

- How much did the station move from the nominal, in mm?

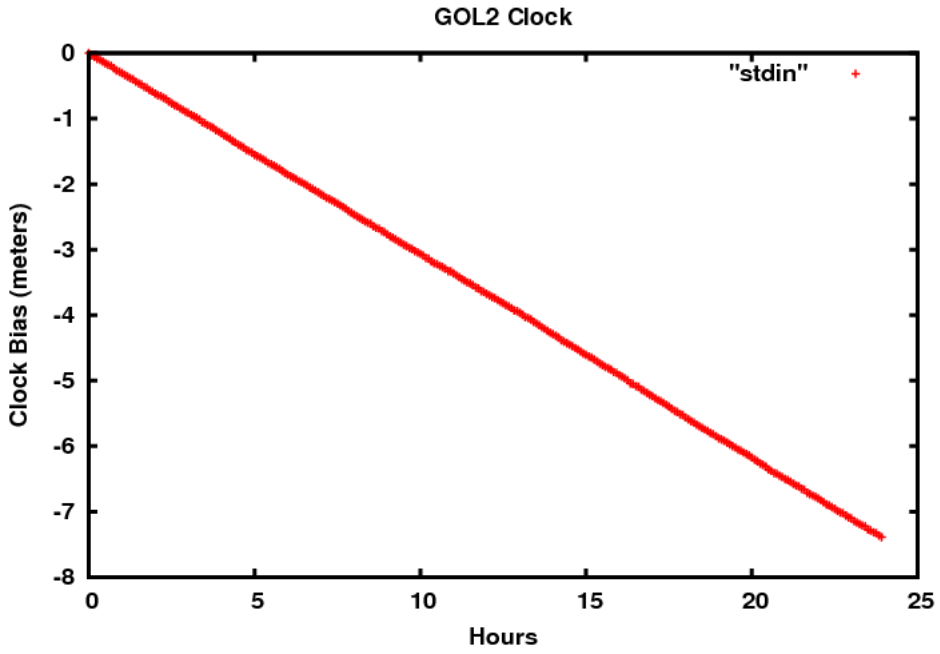
grep 'STA [XYZ]' tdp_final | cl '(c3-c2)*1e6'

```
12.6269796965062
32.177529647015
27.290200250718
```

Example 1: Receiver Clock Solution

- What was the clock behavior, in meters?
  - STA BIAS parameters have units of km in tdp file.
  - Convert time unit to hours since first estimate.

grep 'STA BIASGOL2' tdp_final | cl '(c1-f1)/3600' '(c3-f3)*1e3' | gnup –t 'GOL2 Clock' –xl 'Hours' –yl 'Clock Bias (meters)'



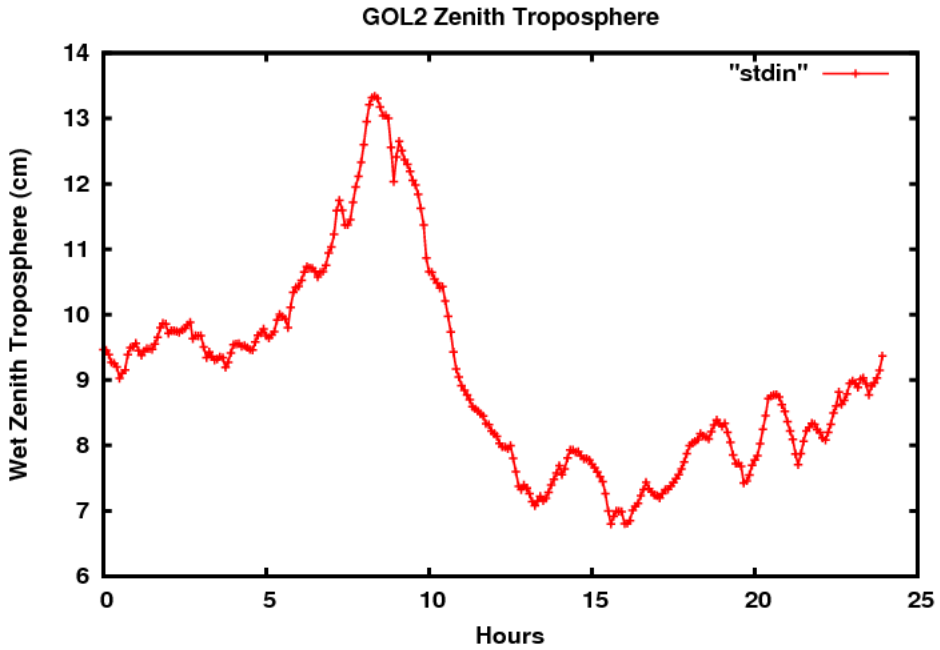
Could add option:
-gif GOL2_clk.gif
to send plot to a GIF file

Example 1: Zenith Troposphere Solution

- What was the zenith troposphere solution, in cm?
  - WETZTROP parameters have units of meters in tdp file.
  - Convert time unit to hours since first estimate.

grep WETZTROPGOL2 tdp_final | cl h1 '1.0e2*c3' | gnup -t 'GOL2 Zenith Troposphere' -xl 'Hours' -yl 'Wet Zenith Troposphere (cm)' –lp

  - NOTE: cl h1 is equivalent to cl '(c1-f1)/3600'



GIPSY: Under the Hood

Example 1: Orbit/clocks used in solution

- gd2p.log indicates orbit/clock products used in solution.
  - Daily orbit/clock product selected from –d option.

At Mon Mar 4 12:11:27 2013 local time, the following files were used
to set up the qregres run

2010-01-10.eo.gz
2010-01-10.pos.gz
2010-01-10.tdp.gz
2010-01-10.wlpb.gz
2010-01-10.frame.gz
2010-01-10.shad.gz
2010-01-10.ant.gz

Antenna Calibration file identified in flinnR,qlR, or flinnR_nf will be used:
/var/opt/goa-var/etc/antenna_cals_xmit/igs08_1604.xyz
in addition to any specified with -AntCal

The source of the orbit/clock files was:

Look in a local directory */workdir/jplorbclk

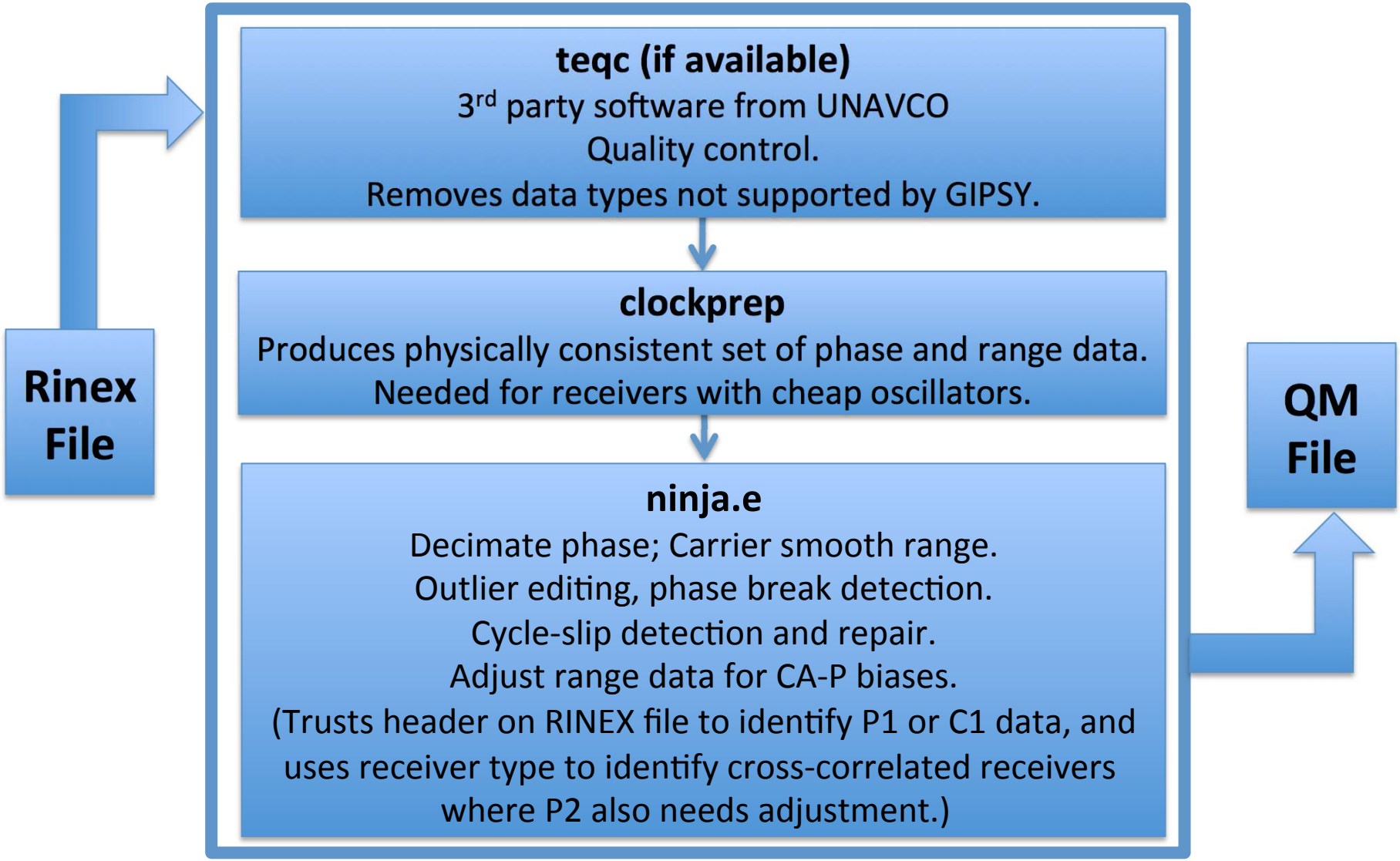
Example 1: GOA modules used to process data

- STDOUT captured by gd2p.log indicates all modules executed.

grep Starting gd2p.log

```
Starting /bin/rm -f qmfile
Starting ninja -a 10 -LC -PC -F -t 300 -n GOL2 -i /opt/goa-6.2/class_gd2p/
          gol20100.10o -q qmfile
Starting gps_shadow -scr /tmp shadow qmfile > /dev/null
Starting weed_qm qmfile pos qmfile_weed
Starting qregres -i qmfile -o rgfile -n qregres.nml  -scr SCRATCH_REGRES -shad
           shadow -time_dep tdp_clk_yaw -sc pos
            -iri_dir /GIPSY_source/goa-var/etc/iri          -pr logs/qregres1.log
Starting wash    -post_wind 0.01 0.002  -edtpnt_max 6 -pb_min_slip 0.0002
```

GNSS Data Editor - **ninja**



ninja: Input – RINEX

```
 2.20           OBSERVATION DATA    GPS                 RINEX VERSION / TYPE
gps1x2rnx           wib                 2004-05-13 10:45:31 PGM / RUN BY / DATE
GRACE A                                                 MARKER NAME
GRACE A             NASA                                OBSERVER / AGENCY
RECNUM              RECTYPE             RECVERS         REC # / TYPE / VERS
ANTNUM              ANTTYPE                             ANT # / TYPE
        0.0000        0.0000        0.0000              APPROX POSITION XYZ
        0.0000        0.0000        0.0000              ANTENNA: DELTA H/E/N
     1     1     0                                      WAVELENGTH FACT L1/2
     9    L1    L2    C1    P1    P2    LA    SA    S1    S2# / TYPES OF OBSERV
    10                                                  INTERVAL
  2002    05    04    20    59   00.000000      GPS     TIME OF FIRST OBS
SNR is mapped to signal strength [0,1,2-9]              COMMENT
 SNR: >500 >100  >50  >10   >5   >3  >1  >0  bad  n/a   COMMENT
 sig:    9    8    7    6    5    4   3   2    1    0   COMMENT
 Loss of Lock Indicator is set for all phase observations COMMENT
 in case a phase break/cycle slip is detected.          COMMENT
 Loss of Lock Indicator is also set at acquisition of a PRN COMMENT
 Undetected cycle slips may remain in this file.        COMMENT
                                                        END OF HEADER
 02 05 04 20 59 00.0000000  0  6 04 05 07 09 24 28
 104251475.70348  81234942.40148  19838395.65649  19838396.22648  19838402.60348
 104251473.19549       791.00049       436.00048       437.00048
 128544047.70646 100164220.57346  24461118.92648  24461119.04546  24461125.54246
 128544049.19648       154.00048        21.00046        20.00046
 111981106.48848  87258034.73448  21309296.14648  21309296.00348  21309303.20948
 111981106.98648       406.00048       125.00048       117.00048
 118098805.44247  92025071.64547  22473455.27448  22473455.50347  22473462.18647
 118098803.93348       338.00048        96.00047        68.00047
 107894201.35548  84073434.91748  20531583.65349  20531583.82848  20531591.42848
 107894199.84849       677.00049       330.00048       244.00048
 118957860.10147  92694461.84947  22636927.93948  22636928.07847  22636934.24447
 118957859.59048       314.00048        69.00047        68.00047
```

ninja: Output – LC and PC

Data Types and Measurements

- GPS observables:
  - C_1,P_1, P_2, L_1 & L_2
  - Linearly combined to obtained the ionospheric-free combinations

$$\begin{aligned} LC &= a_2 L_1 - a_1 L2 \ , \\ PC &= a_2 P_1 - a_1 P_2 \ , \end{aligned}$$

where:

$$\begin{aligned} a_1 &= \frac{{f_2}^2}{({f_1}^2 - {f_2}^2)} = 1.5457 \text{ and} \\ a_2 &= a_1 + 1 = 2.5457, \end{aligned} \tag{28}$$

with f_1 and f_2 being the GPS frequencies at L_1 and L_2.

Note: I'm calling L_1 & L_2 what is sometimes referred as Φ_1 & Φ_2.

ninja: Output – Wind Lane

Other GPS Combinations: The "Wide Lane"

$$
\begin{aligned}
L_w &= \frac{f_1 L_1 - f_2 L_2}{f_1 - f_2}, \\
&= \rho + d_{iono} + \frac{f_1 f_2}{f_1^2 - f_2^2} + \lambda_w b_w \ ; \\
P_w &= \frac{f_1 P_1 + f_2 P_2}{f_1 + f_2}, \\
&= \rho + d_{iono} + \frac{f_1 f_2}{f_1^2 - f_2^2} \ ;
\end{aligned}
$$

where

$$
b_w = N_1 - N_2, \quad \lambda_w \equiv \frac{c}{f_1 - f_2} \approx 86.2 \text{ cm}.
$$

ninja – Example 1

- Using GOL2 RINEX file from gd2p example, and same ninja command:

  cd ..

  mkdir ninja

  cd ninja

  ninja -a 10 -LC -PC -F -t 300 -n GOL2 -i $GOA/class_gd2p/gol20100.10o -q gol2.qm_lr >& gol2_lr_ninja.log

- Look at the log file:

  tail -3 gol2_lr_ninja.log

  ***** PROCESSING COMPLETE *****

  Total number of input epochs = 27126

  Total number of output epochs = 2546

- Number of output epochs much lower than number of input epochs.
  - Original data interval in input RINEX file is 30 seconds.
  - -t 300 option resulted with output data in QMFILE at 5-minute intervals.
    - decimate phase, and carrier-smooth range.

ninja – Example 2

- Using GOL2 RINEX file from gd2p example, but excluding all edit options:

  ninja -a 0 -L1 -L2 -LC -P1 -P2 -PC -t 30 -n GOL2 -i $GOA/class_gd2p/ gol20100.10o -q gol2.qm_hr -notedit -nosane -noedit >& gol2_hr_ninja.log

- Look at the log file:

  tail -3 gol2_hr_ninja.log

  ***** PROCESSING COMPLETE *****

  Total number of input epochs = 27126

  Total number of output epochs = 27126

- All data now in output qm file.
  - No editing based on arc length. (-a 0)
  - No editing checks. (-notedit, -nosane, -noedit)
  - Output interval at original data rate (-t 30). (No smoothing of range).

QM File – Using dump_qm to convert to ASCII

- QM file is binary file.
- Utilities available to extract data.
- E.g., **ls $GOA/bin/*qm*** or **man –k qm | grep 1g | less**
- Most commonly used is **dump_qm**:
  - Generate ASCII stream of data in qm file.
    - See $GOA/file_formats/qm_dump
  - Look at header information of qm file.

    dump_qm -h gol2.qm_lr
  - Look at body (data):
    - Low rate example.

    dump_qm –b –n –d gol2.qm_lr > gol2.dumpqm_lr
    - High rate example.

    dump_qm –b –n –d gol2.qm_hr > gol2.dumpqm_hr

QM File – Looking at the data

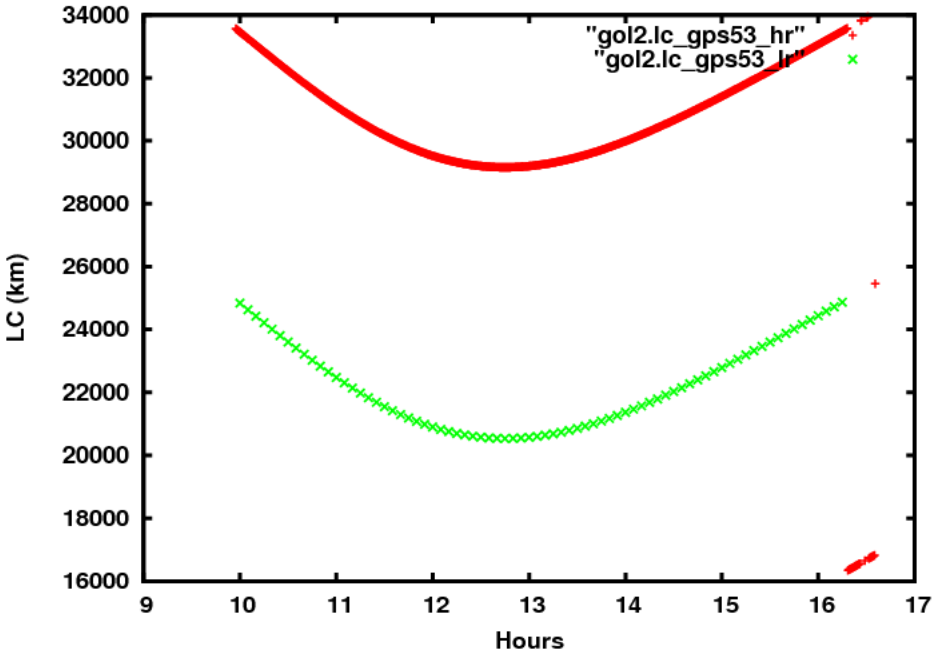
- Use the ASCII dump of the QM file to look at the data.

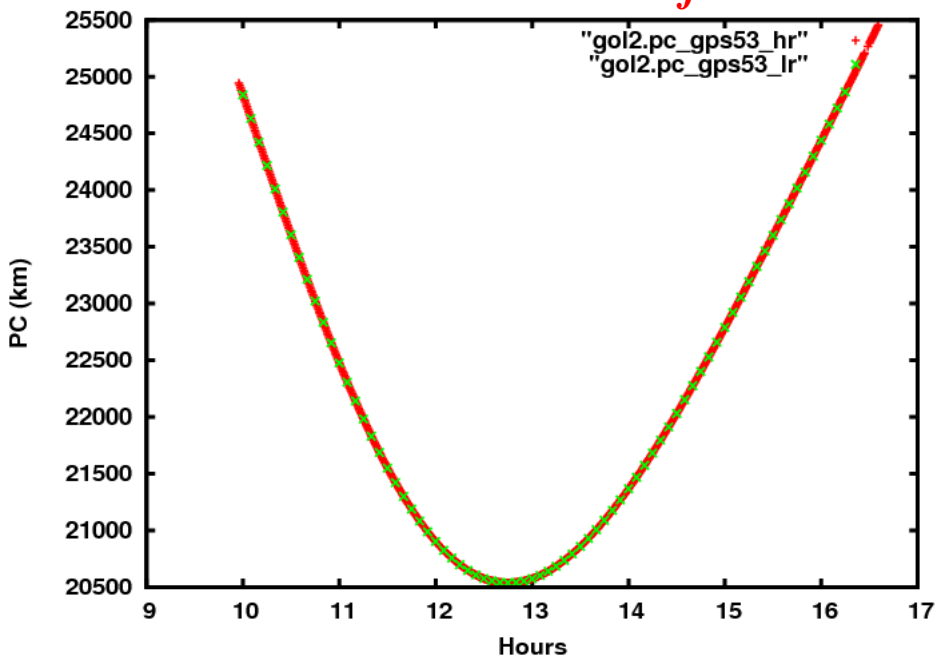
cat gol2.dumpqm_lr | grep LC | grep GPS53 | cl 'c1/3600' 9 > gol2.lc_gps53_lr

cat gol2.dumpqm_hr | grep LC | grep GPS53 | cl 'c1/3600' 9 > gol2.lc_gps53_hr

gnup -p gol2.lc_gps53_hr gol2.lc_gps53_lr -xl 'Hours' -yl 'LC (km)'

Similar commands for PC data





QM File – Other Utilities

- Other utilities available to work with QM file data.
  - **ls $GOA/bin/*qm***)
- Most commonly used QM file utilities in table below.

QM File Utility	Purpose
qm_sta	List participating stations.
qm_sat	List participating satellites.
dump_qm	Convert go ASCII table of data.
qm_dump_to_qm	Convert ASCII stream to binary.
del_qm	Delete specific data points from QM file.
merge_qm	Merge multiple QM files into a single file.
weed_qm	Remove satellites which are not in orbit file.
gps_shadow	Remove data during satellite shadow events.

QM File – Example of Utilities

- Following are commands executed in gd2p example.
- Remove data in shadow.
  - Use shadow file downloaded with orbit/clock products.

cp ../jplorbclk/2010-01-10.shad.gz .

gunzip 2010-01-10.shad.gz

gps_shadow 2010-01-10.shad gol2.qm_lr

- Remove satellites not in orbit file.
  - Use orbit file from gd2p example.

cp ../jplorbclk/2010-01-10.pos.gz .

gunzip 2010-01-10.pos.gz

weed_qm gol2.qm_lr 2010-01-10.pos gol2.qm_lr_weed

qregres – Regression equations

- Formulate the linearized measurement equations for the least squares estimation problem.

$$z(t) = A(t)x(t) + \varepsilon,$$

- z(t) = Pre-fit residual, Observed – Computed ("O-C").
  - Observations from tracking data, e.g., QM file
  - Computed, based on models and nominal (first guess) values.
- x(t) = Parameters to be estimated.
  - **NOTE: GIPSY is estimating "corrections" to the nominal values.**
    - GIPSY linearizes about a nominal.
- A(t) = Partial derivatives of observations with respect to estimated parameters (at nominal values).
- **qregres computes z(t) and A(t).**

qregres – Parameter Types

- Three different types of parameters in qregres:

$$z(t) = A_x(t)\, x(t) + A_p(t)\, p(t) + A_q(t)\, q + \varepsilon$$

  - Orbit pseudo epoch state:

$$\mathbf{x_{j+1} = x_j + \Phi_p(j) p_j}$$

  - Stochastic process:

$$\mathbf{p_{j+1} = M_j p_j + w_j}$$

$$M_j = \exp[-(t_{j+1}-t_j)/\tau]$$

$$E(w^2) = (1 - M_j^2)\sigma^2$$

White Noise: $\tau=0$, $M=0$, $E(w^2) = \sigma^2$

Random Walk: $\tau=\infty$, $M=1$, $E(w^2) = (t_{j+1}-t_j)\sigma^2$

  - Constant parameter:

$$\mathbf{q_{j+1} = q_j}$$

qregres – Observation equation parameters

- Compute expected measurements based on models and nominal values (e.g., station and satellite position (r) and velocity (r'), troposphere etc.).

$$\rho^{*i}(t) = G^i\,(t, \mathbf{q}^*, \mathbf{r}_0^*, \mathbf{r'}_0^*, \mathbf{p}^*)$$

- Compute the pre-fit residual, as difference between actual observations and computed.

$$z^i = \rho^i(t) - G^i\,(t, \mathbf{q}^*, \mathbf{r}_0^*, \mathbf{r'}_0^*, \mathbf{p}^*)$$

- **x(t)** represents parameters to be estimated as "corrections" to nominal values and models:

$$\mathbf{x} = [(\mathbf{r}_0 - \mathbf{r}_0^*)^T, (\mathbf{r'}_0 - \mathbf{r'}_0^*)^T, (\mathbf{p} - \mathbf{p}^*)^T, (\mathbf{q}-\mathbf{q}^*)^T]^T$$

- Observation equation partial derivatives are:

$$\mathbf{A} = [\partial\mathbf{G}/\partial\mathbf{q}, (\partial\mathbf{G}/\partial\mathbf{r})(\partial\mathbf{r}/\partial\mathbf{r}_0), (\partial\mathbf{G}/\partial\mathbf{r})(\partial\mathbf{r}/\partial\mathbf{r'}_0), (\partial\mathbf{G}/\partial\mathbf{r})(\partial\mathbf{r}/\partial\mathbf{p})]$$

- **$\rho^{*i}(t)$, z(t), and A(t) are computed by qregres.**

qregres – Stochastic Models

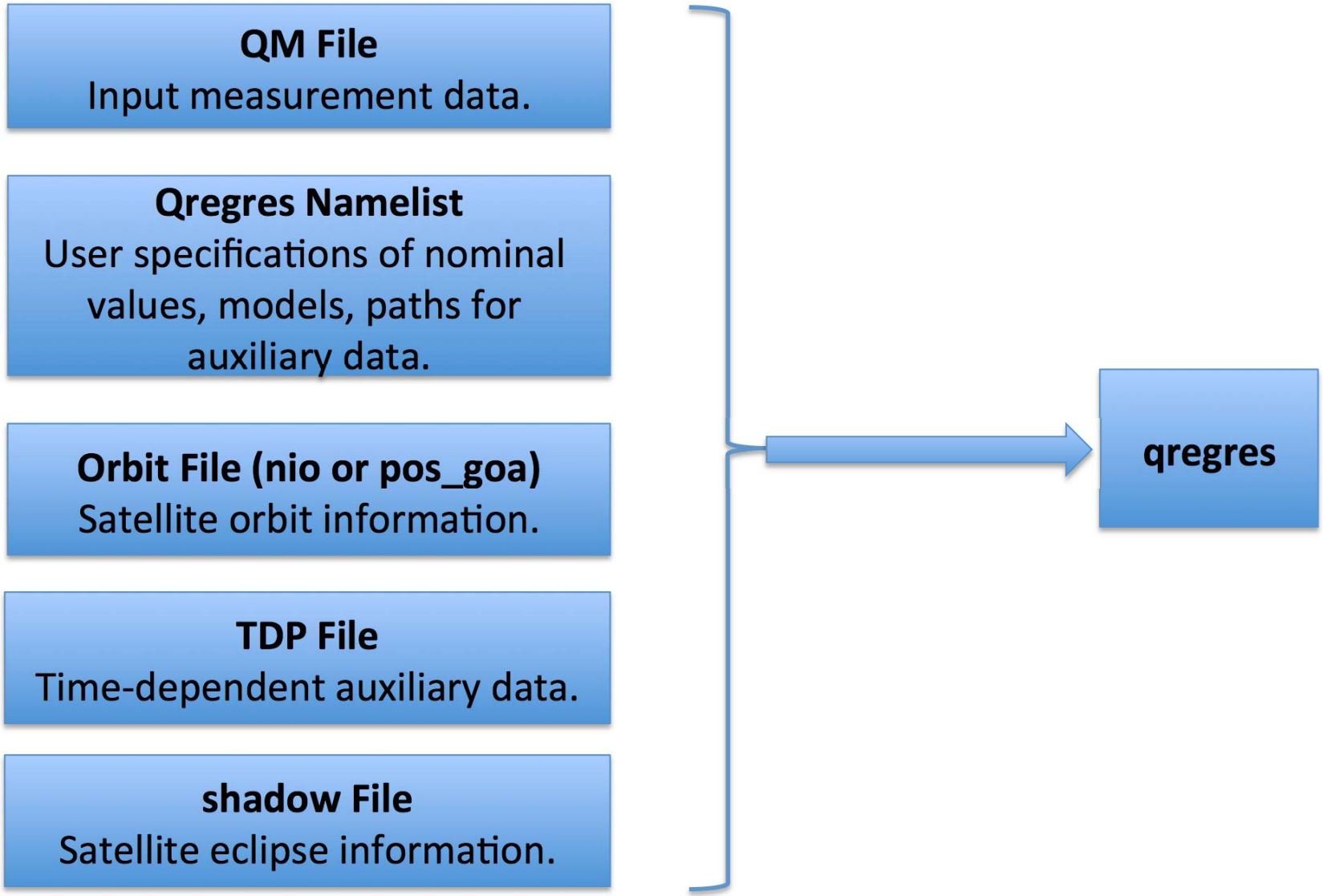
- White noise parameters:
  - Typically clocks and phase biases.
- Random walk parameters:
  - Typically parameters that vary slowly in time.
  - Tropospheric delay.
- Gauss-Markov parameters:
  - Colored noise processes.
  - Typically dynamic parameters such as solar scale and empirical orbit parameters.

qregres – Measurement Models

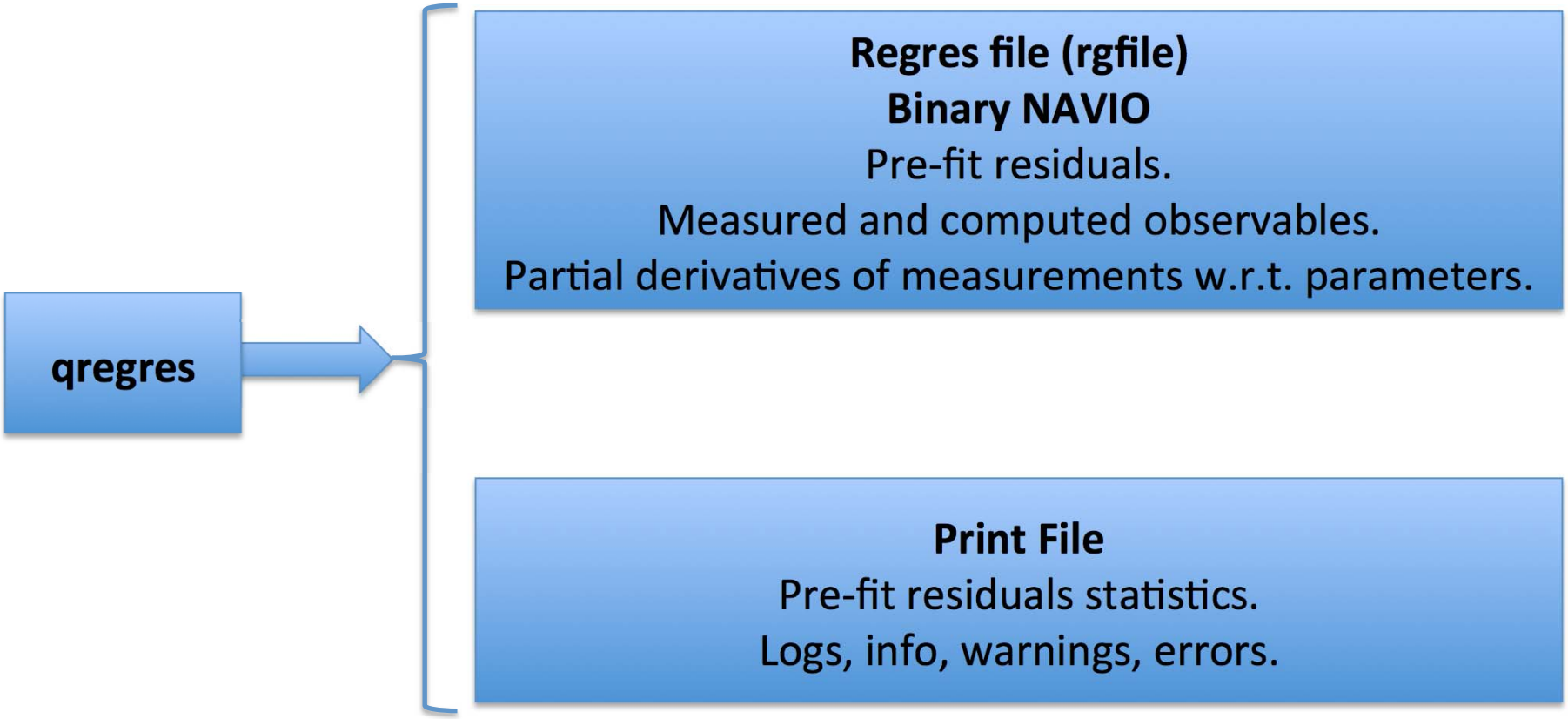
- Examples of models and available parameter partial derivatives:

Effect	Computed measurement	Parameter partials
Orbit initial state	Position, velocity	YES
Clock bias	Transmitter, receiver	YES
Earth orientation	UT1, PM & rates	YES
Tides on station position	Solid, pole, loading	Love numbers
Ionosphere delay	1st order VTEC (BENT, GIM …)	Scale factor
Troposphere delay	YES (NIELL, GPT, GMF, VMF)	Zenith delay
Troposphere gradient	YES	YES
Carrier phase ambiguity	YES	YES
Carrier phase windup	YES	NO
Relativity on clock	YES	NO

qregres – Input



qregres – Output



qregres – Creating the namelist

- Create working directory for qregres examples.

  cd ..

  mkdir qregres

  cd qregres

- Create the qregres namelist file, e.g.,

  s2nml GOL2 -d 2010 01 10 -sta_only -trop_map NIELL -id $GOA_VAR/sta_info/sta_id -pos $GOA_VAR/sta_info/sta_pos -svec $GOA_VAR/sta_info/sta_svec -pc $GOA_VAR/sta_info/pcenter -tides PolTid WahrK1 > gol2_qregres.nml

  – Uses station information database ($GOA_VAR/sta_info) to generate nominal information for receiver (NML_Monument and NMLSite).
    - NML_Monument from station position in sta_pos
    - NMLSite is sum total of:
      – Vector from monument to antenna reference point (ARP) from sta_svec
      – Vector from ARP to antenna phase center from pcenter
    - Apply solid Earth and pole tide models to station position.

qregres Namelist from s2nml

- Need to surround this information with station_model namelist block.
  - Add new first line in gol2_qregres.nml with **$station_model**
  - Add new last line in gol2_qregres.nml with **$end**
- **s2nml** does not provide all necessary information.
  - Difference with namelist used by gd2p

    diff gol2_qregres.nml ../example1/qregres.nml

qregres – Adding Earth Orientation and Antenna Calibration Information.

- Timing and polar motion information (tp_nml) missing. Use from orbit and clock products.

  zcat ../jplorbclk/2010-01-10.eo.gz >> gol2_qregres.nml

- Transmitter antenna calibration not automatically added by s2nml.
  - Currently no available script to add this information.
  - gd2p.pl has built-in perl module to add this information.
    - **Should use same transmitter antenna calibrations as used to generate orbit/clock products.**
    - **See YYYY-MM-DD.ant.gz file.**

qregres – Adding ocean load tide information to qregres namelist.

- **add_ocnld** – Add ocean load tide coefficients to qregres namelist.

  add_ocnld -i gol2_qregres.nml -o gol2_qregres2.nml

  – Adding coefficients to NMLOcnLoad_Amp and NMLOcnLoad_Phs in station_model block of qregres namelist.

  – By default, script is using an ocean loading coefficient file database:

    - $GOA_VAR/sta_info/ocnld_coeff_cm_got48ac_wtpxo8ofunc

  – User could maintain their own database.

    - See for example: http://froste.oso.chalmers.se/loading/
    - An alternative local database from the FES04 ocean tide model is: $GOA_VAR/sta_info/ocnld_coeff_cm_fes04
    - **When using JPL's orbit and clock products be sure to use load tide coefficients with respect to center of mass of sum total of solid Earth and oceans**.

qregres - Example

- Use QM file from ninja example, and qregres namelist from gd2p example.
  - Use orbit/clock products as formatted by gd2p.

qregres -i ../ninja/gol2.qm_lr_weed -o rgfile -n ../example1/qregres.nml -scr SCRATCH_REGRES -shad ../example1/shadow -time_dep ../example1/tdp_clk_yaw -sc ../example1/pos –iri_dir $GOA_VAR/etc/iri -pr qregres1.log

- Output should be the same as from gd2p example.

diff qregres1.log ../example1/logs/qregres1.log

  - Only differences found are in command line and paths to files.

Regres File Utility

- Regres file (rgfile) can be converted to ASCII using oaregres_dump.

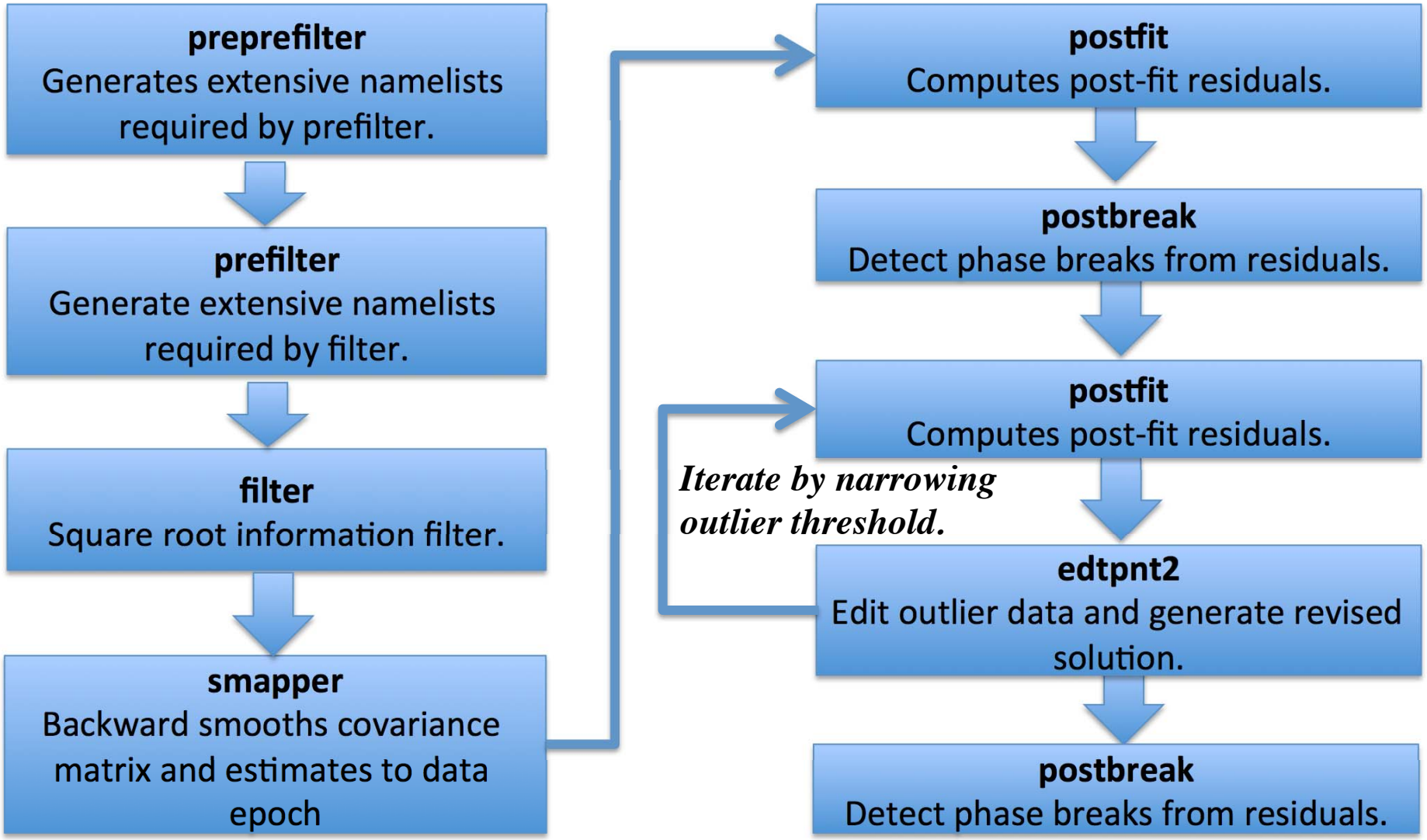
  oaregres_dump rgfile a > rgfile.dump

- Data dumped by measurement groups.

The "Wash" Cycle

- Achieve solution by iteratively "wash"ing out the outliers and missed phase breaks.
  - Determine solution.
  - Compute post-fit residuals.
  - Identify previously missed phase breaks from post-fit residuals.
  - Recompute post-fit residuals.
  - Edit outliers and generate new solution.
  - Gradually narrow outlier detection threshold and iterate.
  - Check for any remaining missed phase breaks.
  - Generate final solution.
- gd2p wash script in $GOA/gd2p/scripts/wash.pl
  - During a run, copied to run directory and named "wash" with input namelist "wash.nml".
  - Log from wash run in logs/wash*.log

gd2p's "wash" cycle



Preparing for the filter

- **preprefilter**
  - Generate extensive input namelist required by prefilter.
  - Creates text file of the stochastics that will drive prefilter.
  - See: $GOA/source/preprefilter/USERGUIDE.TXT

- **prefilter**
  - Generate extensive input namelist required by filter.
  - Creates text file (batch.txt) of the stochastic events that will drive filter.
  - See: $GOA/source/prefilter/USERGUIDE.TXT
  - Usually use preprefilter to generate prefilter input namelist.
    - Input namelist is large.

filter

- Square Root Information Filter
  - Initializes diagonal covariance matrix.
  - Processes data from regres file.
  - At each time step, updates satellite states and adds stochastic process noise at specified event times.
  - At each stochastic update, writes smoothing coefficients to NAVIO file (smooth.nio)
  - At final epoch, writes entire covariance matrix to NAVIO file (accume.nio)
  - Stochastic attributes and structure are defined by input “batch.txt”.
    - Output file from prefilter.
  - See $GOA/source/filter/USERGUIDE.TXT.

smapper

- Maps final epoch covariances and estimates from filter (accume.nio) to the data epoch.
  - Generates smoothed solution (smsol.nio)
- Backward smooths filter solution using smoothing coefficients (smooth.nio).
  - Optionally resolves phase ambiguities backwards.
- Write smoothed solution to NAVIO file (smsol.nio)
- Writes smoothed covariance matrix to NAVIO file (smcov.nio)
- See $GOA/source/smapper/USERGUIDE.TXT

Post-Filter Functions

- **postfit**
  - Computes post-fit residuals of the measurement equation.
- **postbreak**
  - Detects missed phase breaks from discontinuities in post-fit residuals.
- **edtpnt2**
  - Edit data from normal equations based upon post-fit residuals and specified outlier threshold.
- **flag_qm**
  - Mark outliers in QM file based upon post-fit residuals.

wash Example

- Use rgfile from qregres example.
- Use wash script and namelist from gd2p example.

  cp ../example1/wash

  cp ../example1/wash.nml

  wash -post_wind 0.01 0.002 -edtpnt_max 6 -pb_min_slip 0.0002

  – Defines post-fit window.
  – Defines maximum number of edtpnt cycles.
  – Defines threshold for phase break detection by postbreak.
  – Defaults to expected input files “rgfile” and “wash.nml”.
- Solution should be identical to that from gd2p example.

  diff tdp_final ../example1/tdp_final

postpostfit – Statistics of Postfit Residuals

- postpostfit generates statistical summary of postfit residuals.
  - Requires DATAWGHT and LIMITS namelist blocks.
    - Can use blocks from wash.nml.
  - Example:

postpostfit wash.nml postfit.nio Postfit.sum

cat Postfit.sum

start time	stop time	type	user	postfit sigma	npts	outliers
10JAN09 23:59:45	10JAN10 23:54:45	LC	GOL2	6.6321E-06	2170	0
10JAN09 23:59:45	10JAN10 23:54:45	PC	GOL2	3.7481E-04	2170	0

  - Units are km.

residuals.pl – Time series of postfit residuals

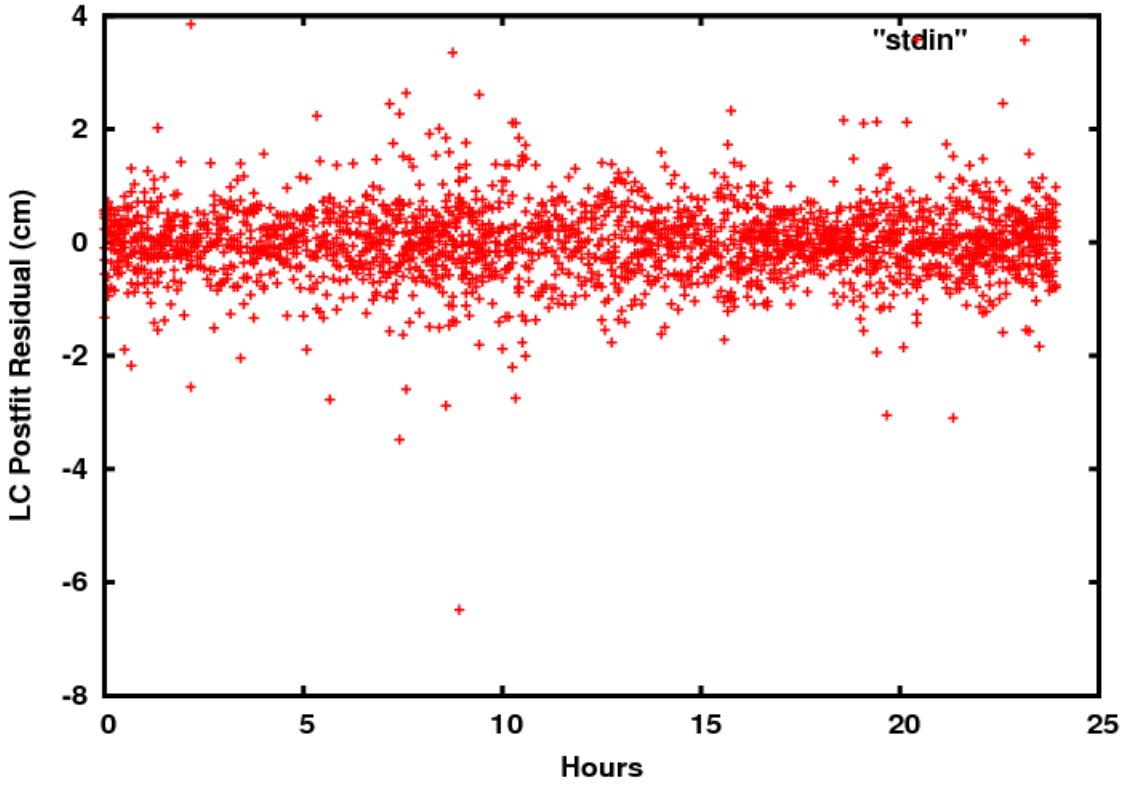
- **residuals.pl** generates tabulated text file with post-fit residuals from binary output of postfit in units of cm.
  - Runs GIPSY program **rdpostfit_dump**
    - See $GOA/source/rdpostfit_dump/USERGUIDE.TXT
  - Default is no options.
    - Use '–h' or 'help' options for help page and output format.
  - Assumes a file named *postfit.nio* exists. (output of postfit)
    - Or use '–p' option to identify specific postfit NAVIO file.
  - Generates file named *residuals.txt*. (format from residuals.pl –h)
    - Excludes outlier points unless '–rej' option used.
  - Output file uses GIPSY codes for data types:
    - See $GOA/source/filter/USERGUIDE.TXT

Data Type	PC	P1	P2	LC	L1	L2	AZ	EL
GIPSY Code	110	111	112	120	121	122	181	182

residuals.pl - Example

residuals.pl -p postfit.nio

cat residuals.txt | awk '{if ($4 == 120) print $0}' | cl '(c1-f1)/3600' 5 | gnup -p -xl 'Hours' -yl 'LC Postfit Residual (cm)'



Time Conversion Utilities

- GIPSY includes many tools to convert common time and date formats.
  - "Seconds" means seconds since Jan 1, 2000 12:00:00 (J2000)

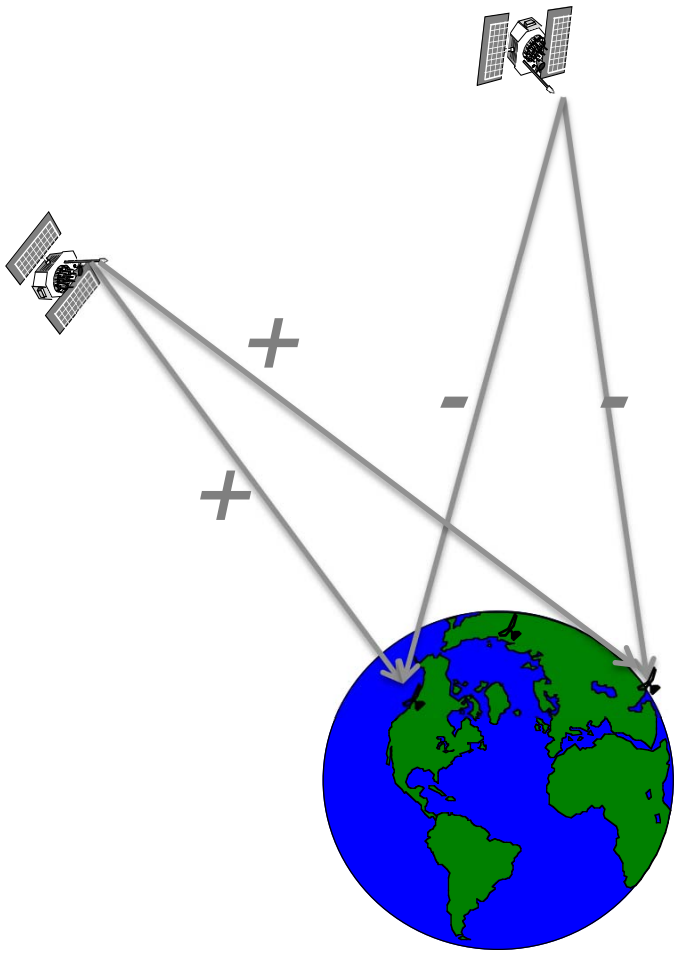
Command	Command	Description
cal2doy	doy2cal	Calendar (YYYY MM DD) to/from day of year.
cal2sec	sec2cal	Calendar (YYYY MM DD HH MN SS) to/from seconds.
char2sec	sec2char	Character (DD-MMM-YYYY HH:MM:SS) to/from seconds.
date2ws	ws2date	Date (YYYY MM DD HH MM SS) to/from GPS week and second.
doy2sec	sec2doy	Day of year (YYYY DDD) to/from seconds.
gps2utc	utc2gps	GPS seconds to/from UTC seconds.
gpsws2sec	sec2gpsws	GPS week and seconds (WWW SS) to/from seconds.
jd2sec	sec2jd	Julian Date (DDDDDDDD.FFFF) to/from seconds.
mjd2sec	sec2mjd	Modified Julian Date (DDDDDDDD.FFFF) to/from seconds.
yymmmdd2sec	sec2yymmmdd	Year/month/day (e.g. 12MAY23) to/from seconds.
yymmmdd2doy		Year/month/day (e.g., 12MAY23) to/from day of year.
taiutc		TAI seconds to UTC seconds.

More Information Under the Hood

- The GIPSY website documentation area provide online videos that have more details about the core GIPSY modules.
  - https://gipsy-oasis.jpl.nasa.gov/docs/index.php?
  - Click on: **GIPSY-OASIS Online Training Course**
  - Topics include:
    - ninja
    - qregres
    - filter and smoother
    - oi (Orbit integrator)
    - Data editing
    - Stacov manipulation and Analysis
    - Strategies

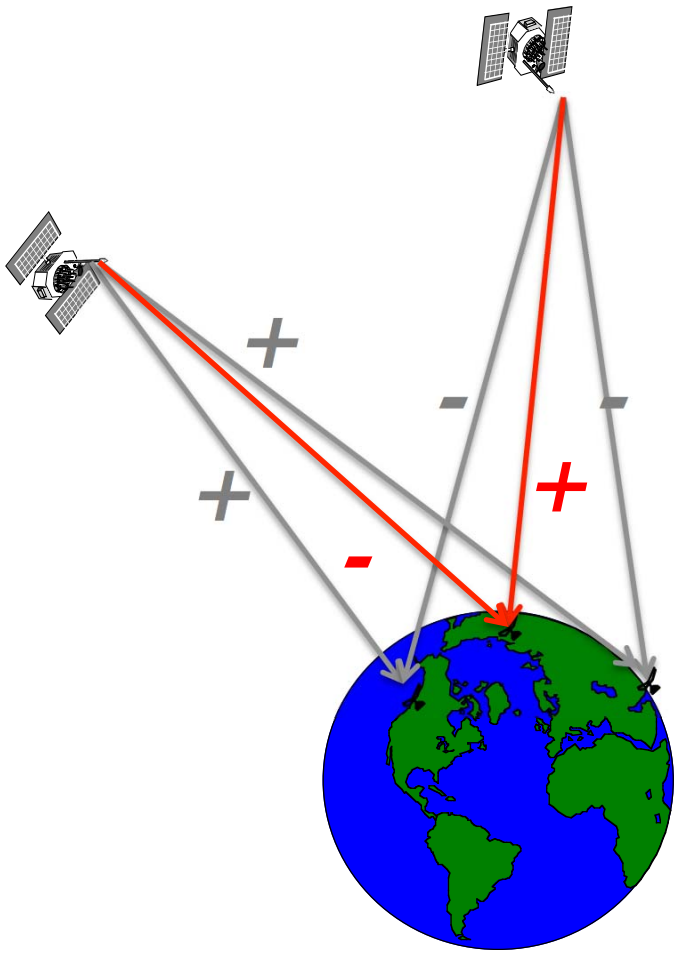
Single Receiver Ambiguity Resolution with GIPSY

- Determine orbit and clock states of GPS satellites using global network of ground stations.
  - Resolves integer ambiguities using double differences of data from ground network/GPS satellites.
    - Includes determining wide-lane and phase biases for each station/satellite pair.
    - Wide-lane=Dual frequency (L1/L2, P1/P2) data combination with long wavelength.
- ALL published JPL products from network solution include:
  - Orbits and clocks of GPS satellites.
  - **Wide-lane and phase bias estimates for each receiver/satellite continuous phase arc.**
    - **This information is in the "wlpb" file.**
  - WLPB file available in Ultra-Rapid, Rapid, and Final products.

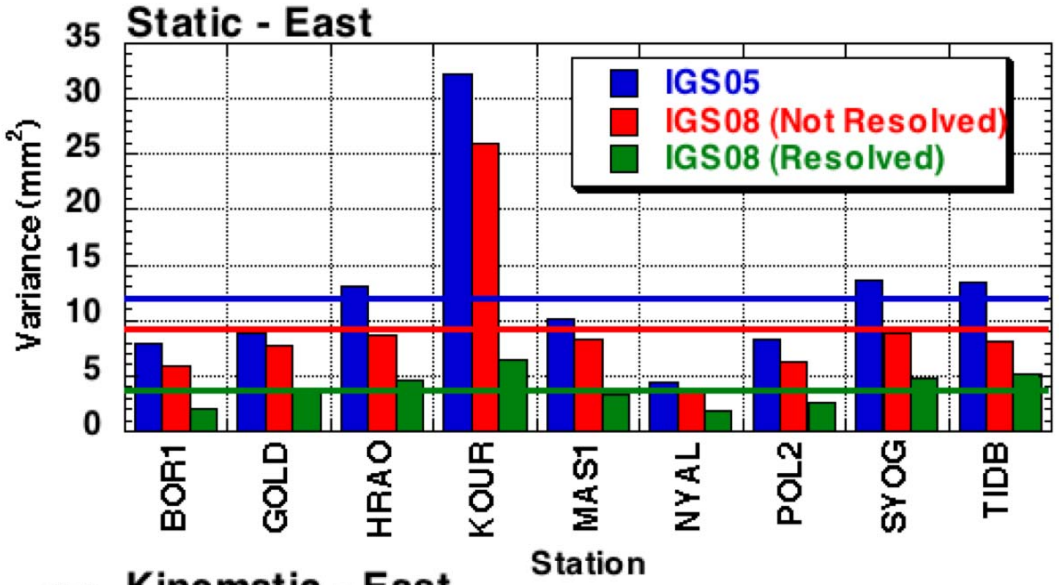


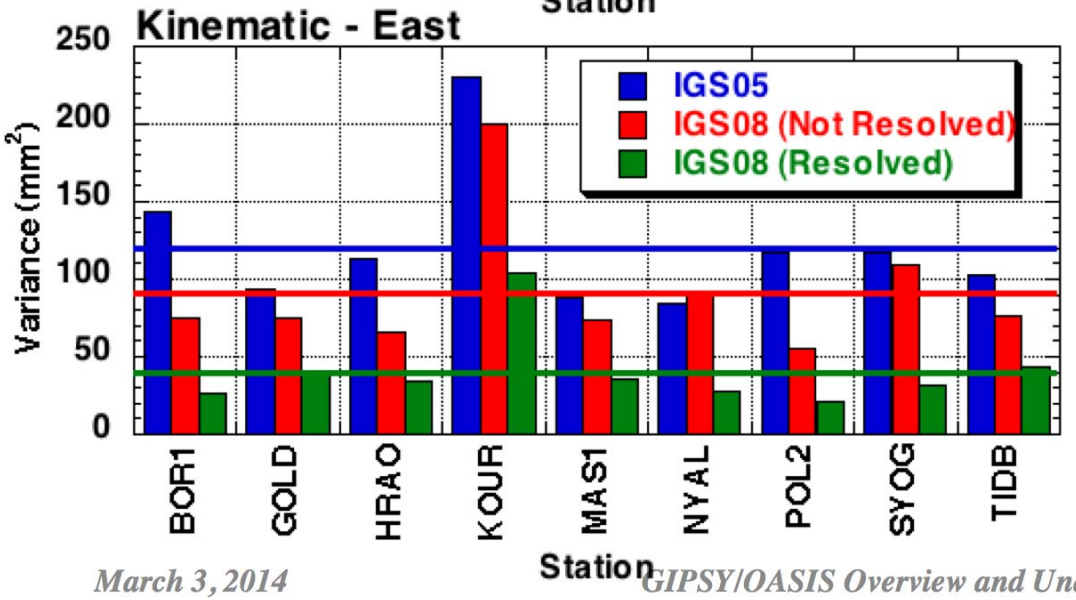
Single Receiver Ambiguity Resolution with GIPSY

- Single receiver ambiguity-resolved positioning:
  - Use published orbits and clocks of GPS satellites.
    - As usual for single receiver positioning.
  - Estimate wide-lane and phase biases for single receiver.
  - Resolve ambiguities using **double differences of wide-lane and phase biases estimates:**
    - Single receiver with respect to all available stations from network solution with common satellites in view. (>> 2).
  - Apply soft constraints to allow for some errors.
  - Iterate for ambiguities.
- Reference: Bertiger et al., 2010.
- Option in gd2p.pl to automatically perform single receiver ambiguity resolution. (-amb_res).



Station Repeatability - East



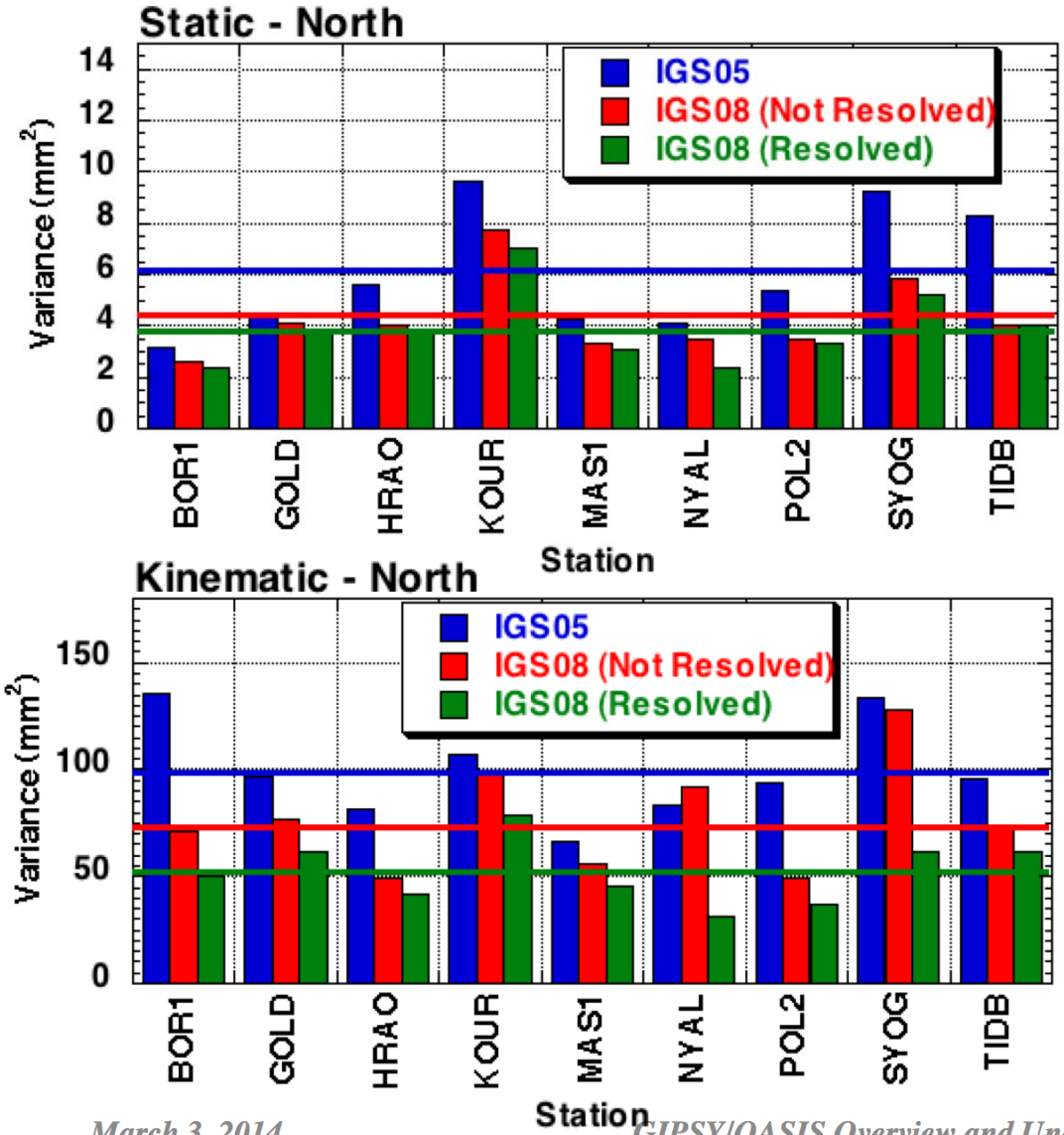


Average Variance (mm^2)

	Static	Kinematic
IGS05	12.4	121.0
IGS08	9.3	91.0
IGS08 (with wlpb)	3.9	40.1

- 25% variance reduction from IGS05 to IGS08.
- 65-70% variance reduction with single receiver ambiguity resolution.
  - **Station repeatability:**
    - **2 mm – Static**
    - **6.3 mm - Kinematic**

Station Repeatability - North

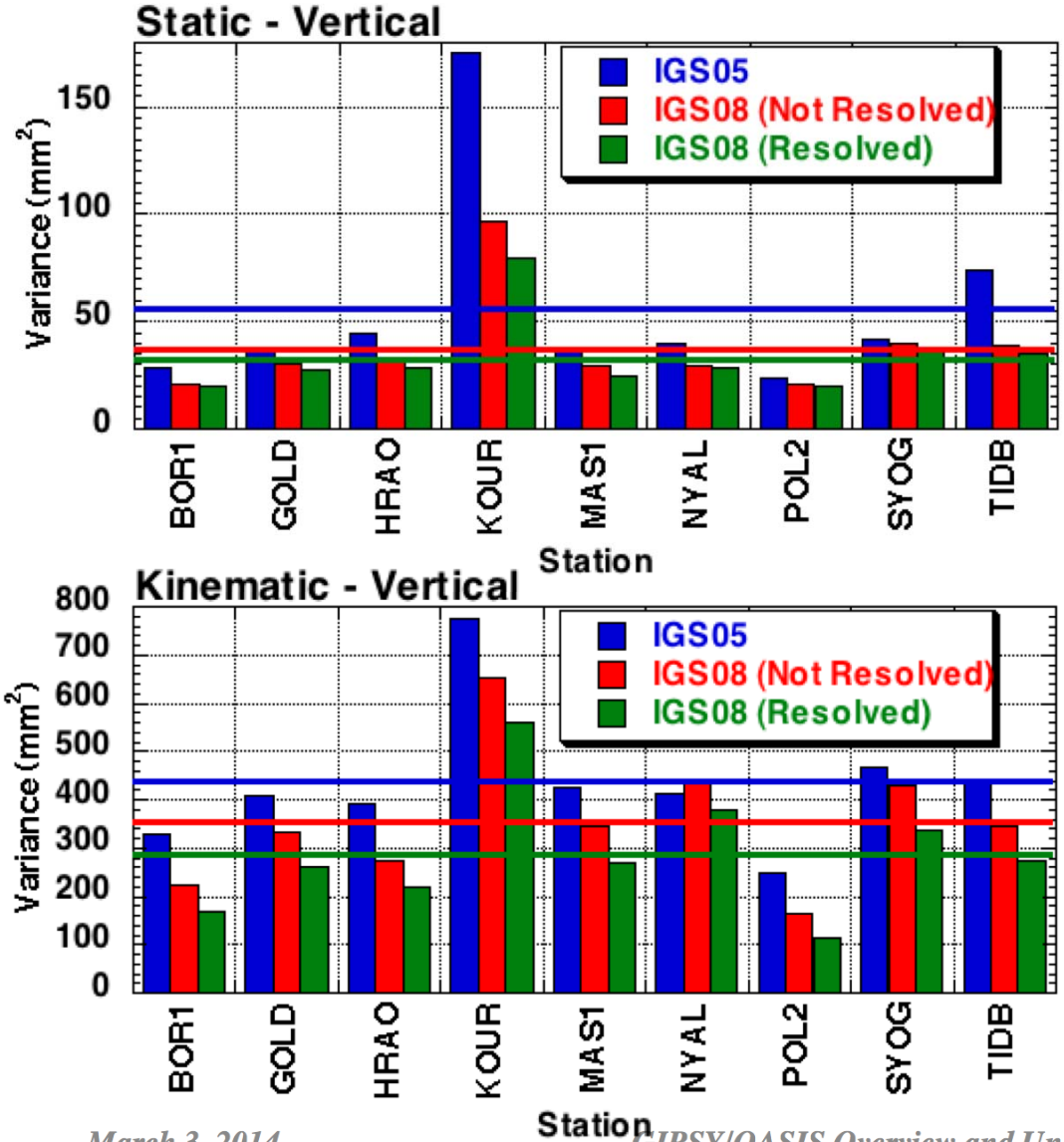


Average Variance (mm^2)

	Static	Kinematic
IGS05	6.0	99.3
IGS08	4.3	76.9
IGS08 (with wlpb)	3.9	52.3

- 23-28% variance reduction from IGS05 to IGS08.
- 35-47% variance reduction with single receiver ambiguity resolution.
  - **Station repeatability:**
    - **2 mm – Static**
    - **7.2 mm - Kinematic**

Station Repeatability - Vertical

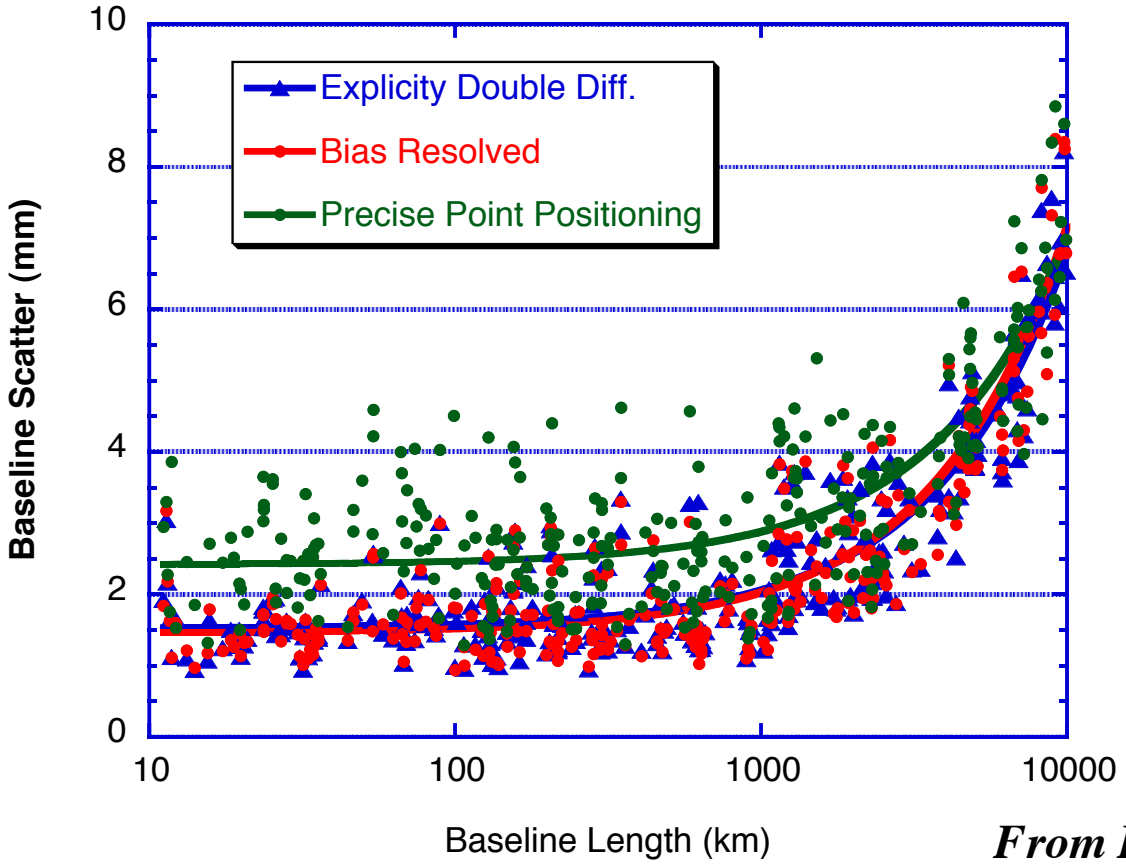


Average Variance (mm²)

	Static	Kinematic
IGS05	56.0	433.3
IGS08	37.8	357.1
IGS08 (with wlpb)	33.3	287.0

- 20-32% variance reduction from IGS05 to IGS08.
- 34-40% variance reduction with single receiver ambiguity resolution.
  - **Station repeatability:**
    - **5.8 mm – Static**
    - **16.9 mm - Kinematic**

Comparing "wlpb" to Explicit Double Differencing



From Bertiger et al., 2010

- 209 IGS global stations June 1-November 30, 2008.
- Single receiver "wlpb" approach is competitive with explicit double differencing approach…..**but simpler**.

Single Receiver Ambiguity Resolved Positioning Summary

- Single receiver ambiguity-resolved static and kinematic point positioning easily applied (plug and play) using:
  - GIPSY
  - JPL's orbit, clock, **AND wide-lane/phase bias (wlpb) products.**
  - WLPB files available with all JPL GPS orbit/clock products.
    - **Including "FINAL" IGS08-based products from 1992-present.**
  - Care should be taken when positioning errors approach 10 cm.
    - E.g., Using wlpb with airplane positioning shown to be risky.
- Results have been shown to be competitive with bias-fixing (double difference) approach (Bertiger et al., 2010).
- Variance reduction of station repeatability in all components.
  - Most significant improvement in East component.
    - Likely due to north/south geometry of GPS satellite orbits.

References

- Bertiger W., S. Desai, B. Haines, N. Harvey, A. Moore, S. Owen, J. Weiss. 2010. Single Receiver Phase Ambiguity Resolution With GPS Data. *J. of Geodesy*, *Vol. 84, No. 5, 327-337,* 2010.
- Bierman, G. J., *Factorization Methods for Discrete Sequential Estimation*, Academic Press, San Diego, CA, 1977.
- Blewitt, Geoffrey, "Carrier phase ambiguity resolution for the Global Positioning System applied to geodetic baselines up to 2000 km", *J. Geophys Res, Vol. 94., No. B8,* 10187-10203, 1989.
- Blewitt, G., "An automatic editing algorithm for GPS data", *Geophys. Res. Let., 17*, pp 199-202, 1990.
- Jefferson, David C., Michael B. Heflin and Ronald J. Muellerschoen, "Examining the C1-P1 Pseudorange Bias", *GPS Solution, Vol. 4, No. 4, 25-30*, 2001.
- Lichten, Stephen M., "Estimation and filtering for high-precision GPS positioning applications", *Manuscripta Geodaetica,* 15:159-176, 1990.
- Wu, J. T., S.C. Wu, G.A. Hajj, W.I. Bertiger, and S.M. Lichten, "Effects of Antenna Orientation on GPS Carrier Phase", *Manuscripta Geodaetica, Vol. 18, No. 2,* 1993, pp. 91-98.
- See also: https://gipsy-oasis.jpl.nasa.gov/index.php?page=references