Massachusetts Institute of Technology

UNAVCO

Working with command-line systems and GAMIT/GLOBK

T. A. Herring R. W. King M. A. Floyd

Massachusetts Institute of Technology

GPS Data Processing and Analysis with GAMIT/GLOBK/TRACK
UNAVCO Headquarters, Boulder, Colorado
10–14 August 2015

Material from T. A. Herring, R. W. King, M. A. Floyd (MIT) and S. C. McClusky (now ANU)

Quick poll

- Who is using what type of operating system?
  - Linux (brand?)
  - Mac OS X
  - Windows
- Who has installed the software successfully?
- Who has run the example successfully?
- Who has their own data set to process?
- What data do others plan to use?

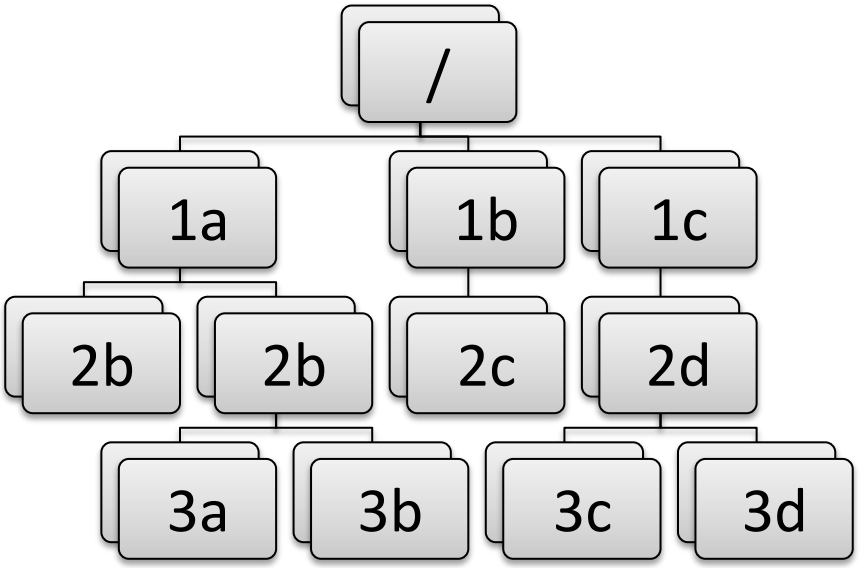
Introduction to command-line computing

- Directory structure and navigation
- Using a command line
- Commands to know
- Introduction to shell scripts

Directory structure and navigation

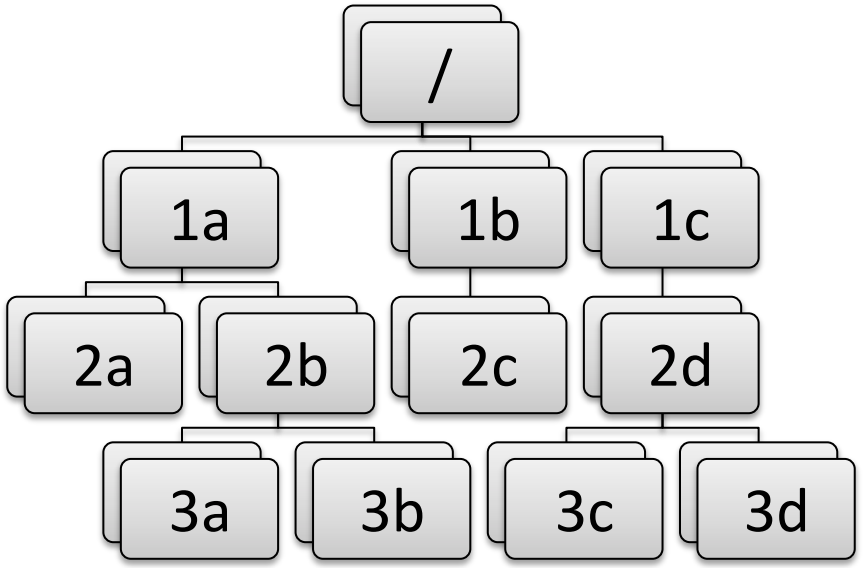
Directory structures

- One must be familiar with the layout of files and directories (or “folders”)
- Once one has a mental “map” of the directory structure, navigating between directories and finding files is easier
- Think of it as a filing cabinet or family tree



Directory structures

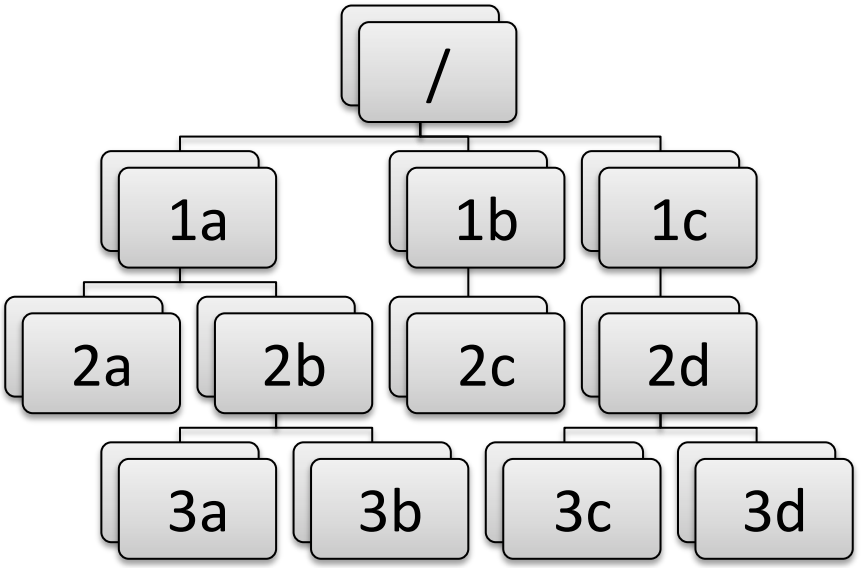
- Top-level (“root”) directory (e.g. “/” on Unix, “C:\” on Windows, etc.)
- User’s current working directory is referred to by the shorthand “.” [dot]
- The “parent” directory is one level above the current working directory in the hierarchy
- Parent directory is referred to by the shorthand “..” [double dot]



Changing directory

Once user knows where they are with "mental map" of directory structure, move around. We can move up or down the hierarchy but not sideways.

- `cd /`
  - Takes user to top-level ("root") directory
- `cd 1b`
  - Takes user to "1b" directory in first level (move down hierarchy)
- `cd 2c`
  - Takes user to "2c" directory in second level, below "1b" (move down hierarchy)
- `cd 2d`
  - Unknown directory. Why?
  - User attempting to move sideways but "2c" not connected directly to "2d".



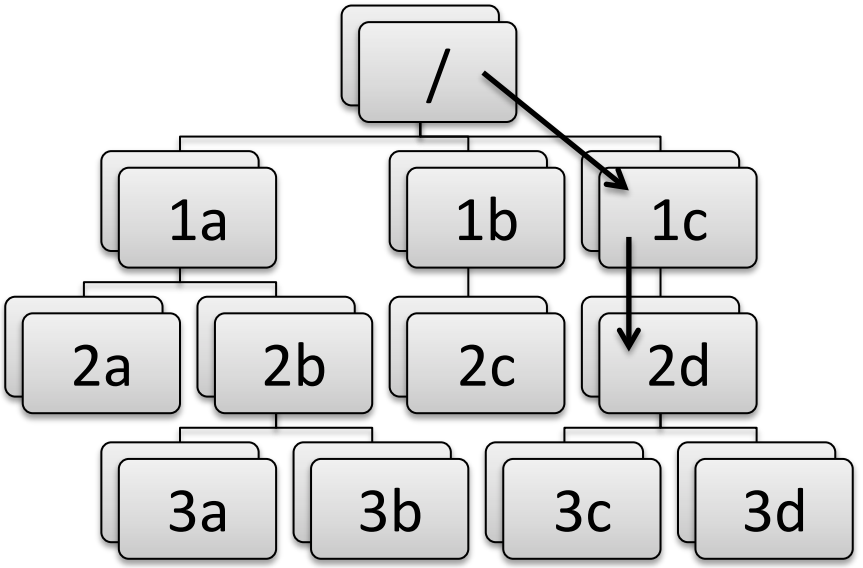
Absolute paths

To move back up the hierarchy to “2d”, one may explicitly start from the top level, e.g.

- `cd /`
- `cd 1c`
- `cd 2d`

or, combined, simply

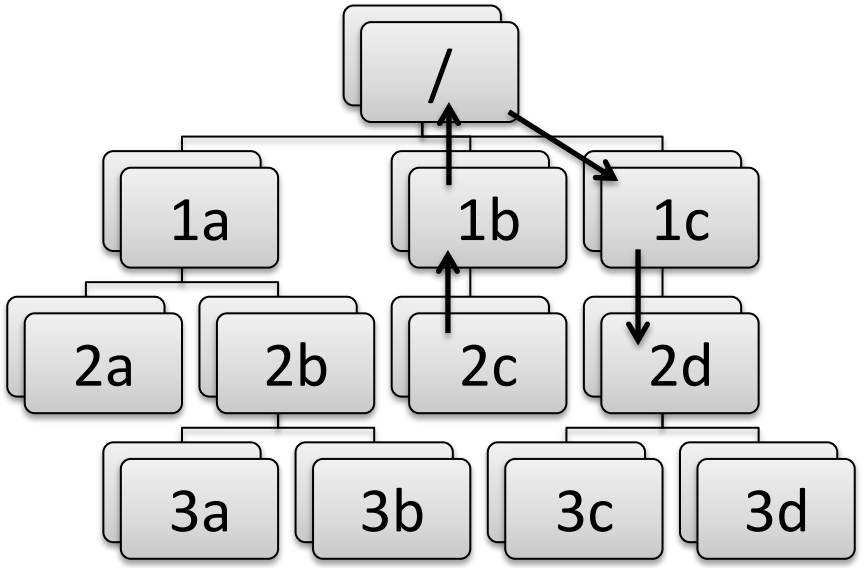
- `cd /1c/2d`
  - Directories are separated by forward slashes



Relative paths

Or:

- Move back up to “1b”
  - `cd ..`
- Move back up to “/”
  - `cd ..`
- Move down to “1c”
  - `cd 1c`
- Move down to “2d”
  - `cd 2d`
- Or, combined, simply:
  - `cd ../../1c/2d`



Using a command line

Using a command line

- Basic syntax is:

  `<command> <options> <argument(s)>`
- `<command>` is the program to run, including directory if not included in PATH environment variable (more in a couple of slides…)
- `<options>` are usually prepended by a dash (e.g. -a)
- `<argument(s)>` are usually input or output files to work on
- Commands may or may not have options or expect arguments

Basic commands

- `cd`
  - Change directory, for navigating the directory structure
- `pwd`
  - Print working directory, to know where you are
- `ls`
  - List directories and files in current working directory (“.”) or directory given after command
- Use the “tab” key to auto-complete options

Environment variables

- A computer must be told information in order to work the way you expect
- Many important settings are kept in "environment variables"
  - $HOME = user's home directory
  - $PATH = list of directories containing programs
  - $SHELL = user's command shell
- `printenv`
  - Prints information on environment variables

Local variables

- To make life easier, one may also set local variables, which may be referred back to at any time
- Useful if one finds the need to write the same thing may times
- sh/bash:
  - `var='Hello'`
  - Instead of writing "Hello", any reference to $var will be equivalent to "Hello"
  - `var=( Hello Goodbye )`
  - Any reference to ${var[0]} will be equivalent to "Hello" and ${var[1]} to "Goodbye"
- csh/tcsh:
  - `set var = 'Hello'`
  - Instead of writing "Hello", any reference to $var will be equivalent to "Hello"
  - `set var = ( Hello Goodbye )`
  - Any reference to $var[1] will be equivalent to "Hello" and $var[2] to "Goodbye"

Commands to know

Everyday commands

- awk
- grep
- sed
- sort
- paste/join
- tr
- echo/cat

awk

Powerful formatted read/write utility, e.g.

- `awk '{print $1,$2,$3}' <file>`
  - Prints first, second and third white-spaced columns ("fields") from each line of <file>
- `awk -v n=3 -F',' '{print $NF/n}' <csv-file>`
  - Prints the last comma-separated field divided by 3 from each line of <csv-file>
- `awk 'BEGIN {sum=0}; {sum=sum+$1}; END {printf("%.1f\n",sum/NR)}' <file>`
  - Calculate mean of first field: sums first field on each line then divides by number of lines ("records")

grep

Pattern-matching command ("**g**eneral **re**gular ex**p**ression")

- `grep 'hello' <file>`
  - Prints all lines from <file> with occurrence of "hello" in them
- `grep –ci '^POS S' <file>`
  - Prints the number ("-c") of lines that begin ("^") with "POS S" in either upper- or lower-case letters ("-i") in <file>
- `grep '^ .* P$' <file>`
  - Print all lines in <file> that begin ("^") with a space, followed by any number of any characters (".*"), and end ("$") with a space followed by P

sed

Basic text editor

- `sed 's/ //g' <file>`
  - Substitute ("s") all ("g") instances of a single whitespace with nothing (i.e. delete all whitespace)
- `sed '/^ *$/d; s/hello/goodbye/1' <file>`
  - Delete ("d") all empty lines and substitute the first instance of "hello" with "goodbye" on each line of <file>

sort

Sorts records

- `sort <file>`
  - Outputs basic alpha-numerically ordered <file>
- `sort -u <file>`
  - Same as above but uniquely sorted (i.e. removes duplicate records)
- `sort -g -k3 <file>`
  - General numeric ordering based on third field of <file>
- `sort -uk2.1,2.4 <file>`
  - Sort based on first character of second field to fourth character of second field and use this as the basis for the uniqueness test

tr

Basic translation

- `tr '[:upper:]' '[:lower:]'`
  - Transposes all upper-case letters to lower-case
- `tr -d '\r'`
  - Deletes all carriage return ("CR") characters (useful for changing a file's line ending from DOS to UNIX format)

echo/cat

Echoes the argument

- `echo 'Help!'`
  - Prints "Help!"
- `cat <file>`
  - Reads out entirety of <file>
- `cat << END`

```
Help!
END
```

  - Same as "echo 'Help!'"

Redirection

- The output from one command may be written to a file…
  - “>” to *overwrite* an existing file
  - “>>” to *append* to an existing file
  - `sort [file] > [sorted file]`
- …or “piped” to another command, effectively forming the second command’s input
  - “|”
  - `grep ‘^ .* P$’ [file] | sort > [grep’d and sorted file]`

Shorthands

- Top-level ("root") directory = "/", e.g.
  - `cd /`
- Your home directory = "~" or "$HOME", e.g.
  - `ls ~`
- "Links" or "shortcuts" may be created, e.g.
  - `ln -s /home/user/gg/10.6 ~/gg`
- This creates a link in the user's home directory called "gg" that points to the directory /home/user/gg/10.6
  - Rather than `cd /home/user/gg/10.6`, one can get to the same place simply with `cd ~/gg`
  - (This is used in GAMIT/GLOBK scripts and must remain in place!)

Useful commands

- `du`
  - Disk usage: useful if you want to know how much space your (or others'!) directories are taking up
- `df`
  - Disk free space: useful if you want to know how much disk space is used and free
- `top`
  - Table Of Processes: useful if you want a real-time overview of running processes
- `ps`
  - List processes: useful if you want to see what processes are running and their process numbers, commands, etc.

Introduction to shell scripts

What is a script?

- Scripts contain a series of commands written in one file and prepended by a “hash-bang”
  - `#!/bin/sh` for original Bourne Shell (usually the same as bash on modern systems)
  - `#!/bin/bash` for Bourne Again Shell
  - `#!/bin/csh` for C Shell (usually the same as tcsh on modern systems)
  - `#!/bin/tcsh` for TENEX C Shell
- The script may then be executed to run all of the commands in sequence as written

Script example

```
#!/bin/bash
echo –n ‘The ISO date is: ‘
date +‘%Y-%m-%dT%H:%M:%S%Z’
echo -n ‘The mean of all numbers between 1 and 10 is: ’
echo 1 10 | awk ‘BEGIN {sum=0; n=0}; {for (i=$1; i<=$2; i++) {sum=sum+i; n++}}; END {print sum/n}’
echo ‘Goodbye!’
```

Installing GAMIT/GLOBK

Sources of prerequisite information

http://web.mit.edu/mfloyd/www/computing/gg/pre/

ftp://guest@chandler.mit.edu/updates/documentation/GAMIT_prerequisites.pdf

http://web.mit.edu/mfloyd/www/computing/mac/gfortran/

http://web.mit.edu/mfloyd/www/computing/mac/gv/

Separation of tasks

- Source code directory
- Installation directory
- Processing directory

Source code directory (optional)

Source code directory

- Users may wish to keep a local copy of source code
  - As backup in case of problems during installation
  - If unable to reconnect to the source code repository (chandler.mit.edu)
- If you wish to do this, keep it separate from where you intend to *install* GAMIT/GLOBK
  - `~/src/gg/10.6`
  - `~/Programs/src/gg/10.6`

Master installation directory

Master installation directory

- Choose a suitable directory for installing the software
  - Suggested place in home directory, e.g. `~/src/gg`, `~/Programs/gg`, etc. (for example, I install GG version 10.6 in `/Users/Mike/Programs/gg/10.6`)
  - Alternative may be your `/usr/local` directory, e.g. `/usr/local/gg/10.6`
  - Take care not to mix source versions, e.g. 10.5 versus 10.6
- Change to this directory to download (or copy) the source code
- This will be the directory that is ultimately linked from your home directory (`~/gg`)

Downloading source via FTP

FTP server

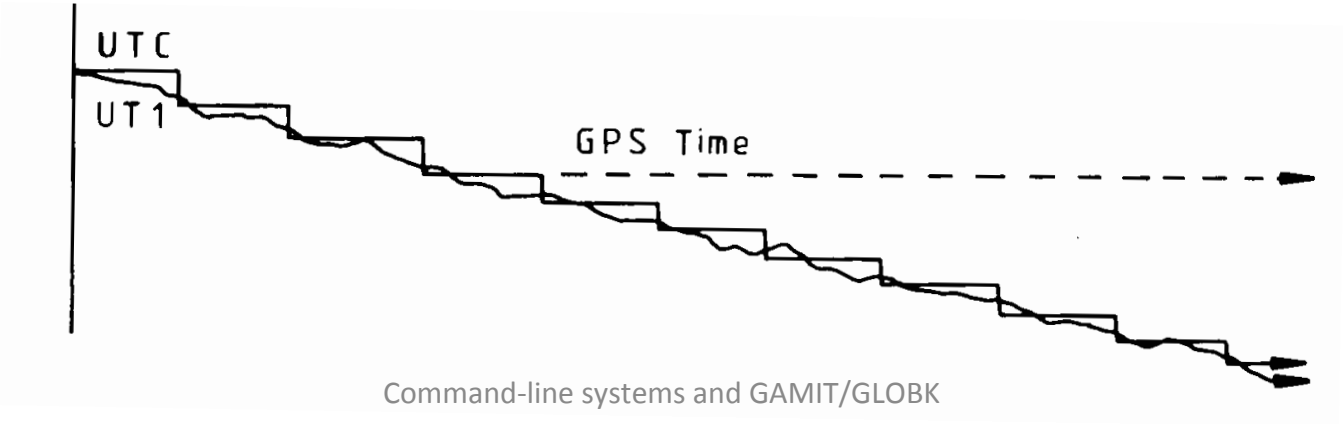
- chandler.mit.edu
  - username: guest
  - password: [changeable]
- Use FTP client, such as `ftp` or `ncftp`
- Alternatively, use internet browser
  - ftp://guest@chandler.mit.edu

Source code

- Change directory to `updates/source/`
- Need *at least*:
  - com
  - gamit
  - help
  - kf
  - libraries
  - tables
  - incremental_updates (if any)
- Also download install_software
- Depending on your processing strategy, may also need to download grids (e.g. ocean-tide loading, atmospheric loading grids, etc.) from ftp://everest.mit.edu/pub/GRIDS

Updates!

- Incremental updates are made available approximately every month, so please check at least
  - Earth orientation parameters (pole.* and ut1.*; or sh_update_eop)
  - SVN-PRN translation tables (svnav.dat)
  - Differential code biases (dcb.dat)
  - Leap seconds (leap.sec)
  - Loading grids (ftp://everest.mit.edu/pub/GRIDS)
- Example: 2015-06-30T23:59:60Z leap second



Documentation

- Top-level "README" file at ftp://guest@chandler.mit.edu/updates/README
- Change directory to `updates/documentation/`
  - GAMIT/GLOBK prerequisites in **GAMIT_prerequisites.pdf** ftp://guest@chandler.mit.edu/updates/documentation/GAMIT_prerequisites.pdf
  - Introductory GPS material in **Intro_GG.pdf** ftp://guest@chandler.mit.edu/updates/documentation/Intro_GG.pdf
  - GAMIT reference manual in **GAMIT_Ref.pdf** ftp://guest@chandler.mit.edu/updates/documentation/GAMIT_Ref.pdf
  - GLOBK reference manual in **GLOBK_Ref.pdf** ftp://guest@chandler.mit.edu/updates/documentation/GLOBK_Ref.pdf

Installing GAMIT/GLOBK etc.

Required tools

Depending on your system, a number of programs may need to be added. One needs:

- A Fortran code compiler
- A C code compiler
- X11 libraries and headers, specifically:
  - libX11.a, libX11.so, libX11.dylib or libX11.la (depending on your system)
  - Xlib.h
- Linux
  - Be sure a C-shell (csh and tcsh) is installed (this is not the case by default with Ubuntu, for instance)
  - X11 libraries and headers may also need to be installed
- Mac
  - Have an Apple ID and download the latest "Command Line Tools for Xcode" (Mac OS X 10.7.3 or later) or "Xcode" (prior to Mac OS X 10.7.3) appropriate to your system from https://developer.apple.com/downloads/index.action
  - X11 was replaced by XQuartz (http://xquartz.macosforge.org/) for Mac OS X 10.8 (Mountain Lion) and later
- Windows (Cygwin)
  - Devel/make
  - Math/bc
  - Shells/tcsh
  - X11/libX11

Notes on known problems

- Very new gfortran releases, especially those with a version number ending in 0 (e.g. 4.9.0), often are buggy and produce compilation problems
  - If this is the case, try compiling a program using only the '-O3' flag or revert to an older, stable version of gfortran
- I currently run gfortran 4.9.2 on my laptop with Mac OS X 10.10 (Yosemite) and 4.7.3 on MIT computers with Ubuntu Linux
  - Note Ubuntu's gfortran 4.8 appears to be buggy

Running install_software

From the master installation directory, where the source tar-files and install_software should be copied

- Run `./install_software`
- As you pass through the installation process, *please read the questions*, e.g.
  - ```
    Searching directories set in libraries/Makefile.config for X11
    installation
    Verified these paths to X11 libs and includes
    X11LIBPATH:
    X11INCPATH:
    Are these paths complete and correct for your system? (y/n)
    ```
- If they are not correct, say “n” then install_software will search or exit and one can then edit libraries/Makefile.config appropriately
    ```

A note here on permissions

- A computer may read (“r”), write (“w”) and/or execute (“x”) a directory or file
- Each action may be allowed by a user (“u”), group (“g”) or others (“o”)
- A computer must follow instructions, called “permissions”, on if it allowed to do any or all of these for any
- Any file that you want to run as a program must be made “executable”
  - `chmod a+x <file>`
  - Change moderations (permissions) so executable (“x”) permissions are added to <file> for all (“ugo”)
- You may find you need to verify that directories and files are readable, writable and/or executable as necessary throughout your UNIX experience

Potentially necessary edits

- libraries/Makefile.config is the main control file for the installation process
- Check:
  - X11LIBPATH (path to libX11)
  - X11INCPATH (path to Xlib.h)
  - MAXSIT (max. number of sites to process simultaneously)
  - MAXSAT (do not change)
  - MAXATM (max. atmospheric estimates per session)
  - MAXEPC (max. epochs per session, e.g. 24 hours at 30 s interval = 2880 measurement epochs)
  - OS block (usually no need to change)

Setting environment variables

- sh/bash (e.g. in ~/.bash_profile or ~/.profile):

```
gg='/usr/local/gg/10.6'
PATH="$gg/com:$gg/gamit/bin:$gg/kf/bin:$PATH" && export PATH
HELP_DIR="$gg/help/" && export HELP_DIR
INSTITUTE='MIT' && export INSTITUTE
```

- csh/tcsh (e.g. in ~/.cshrc):

```
set gg = '/usr/local/gg/10.6'
setenv PATH "$gg/com:$gg/gamit/bin:$gg/kf/bin:$PATH"
setenv HELP_DIR "$gg/help/"
setenv INSTITUTE 'MIT'
```

Additional software

- Generic Mapping Tools (GMT) (http://gmt.soest.hawaii.edu/)
  - Required for plotting scripts to work
  - Scripts in com/ use GMT 5
  - Prepend com_preGMT5/ to $PATH if using GMT 4
    - These scripts are no longer updated, so switch to GMT 5!
- Tom's GGMatlab tools (http://www-gpsg.mit.edu/~tah/GGMatlab/)
  - tsview
  - velview

GMT

Install netCDF (http://www.unidata.ucar.edu/downloads/netcdf/current) first:

- If unable to install via, e.g. Ubuntu Software Manager then…
- Download latest source code to suitable directory (e.g. ~/src)
  - `wget http://www.unidata.ucar.edu/downloads/netcdf/ftp/netcdf-4.3.0.tar.gz`
- Expand tar-file
  - `tar xvfz netcdf-4.3.0.tar.gz`
- Change directory and configure *without* netcdf-4 support (unless you have required HDF5 and zlib installed) and install in /usr/local
  - `cd netcdf-4.3.0`
  - `./configure --disable-netcdf-4`
- Run the usual make sequence to install in /usr/local (configure's default)
  - `make`
  - `make check`
  - `sudo make install`

GMT

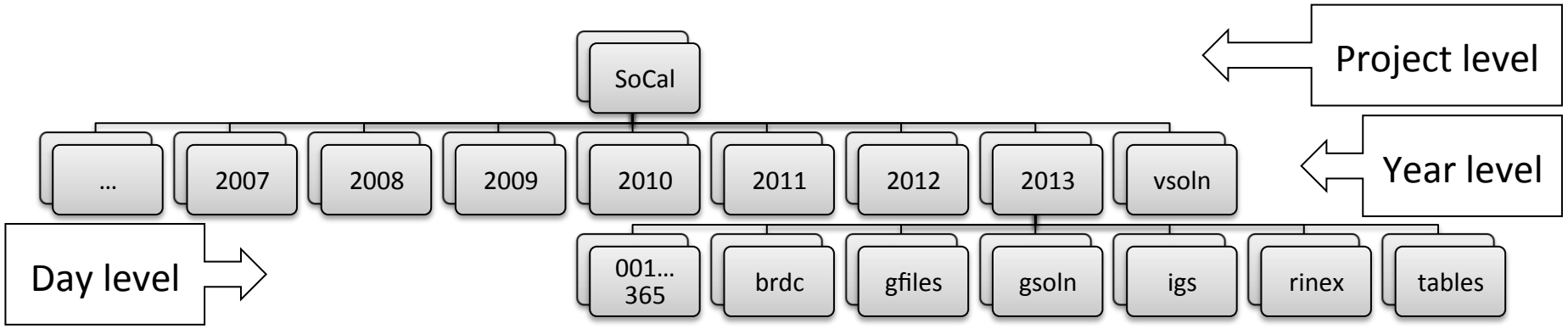
- Download and execute install_gmt.sh (http://gmt.soest.hawaii.edu/gmt/install_gmt.sh)
- Answer the questions appropriately (most defaults settings are adequate)
- Default configuration installs netCDF in /usr/local/lib, /usr/local/include, etc. (previous slide)
- Suggested installation directory for GMT is /usr/local/GMTX.Y.Z (where X.Y.Z is currently 4.5.13 or 5.1.2)
- Be sure to follow the instructions regarding setting *environment variables* (PATH, MANPATH)

Processing directories

Processing directory

- The *processing* directory will not have the same structure as the *master installation* directory
- Choose a different location, do not process in your master installation directory
- We will, however, be copying or linking to the master installation tables (via symbolic link or "shortcut" `~/gg/tables`)

Example continuous GPS structure



Example survey GPS structure