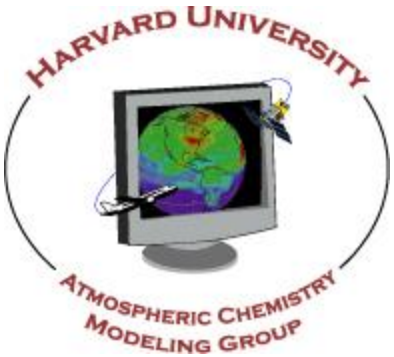
GEOS-Chem Model Clinic

Melissa Payer and Sajeev Philip
GEOS-Chem Support Team
geos-chem-support@as.harvard.edu



The 6th International GEOS-Chem Meeting, Harvard University
06 May 2013

Overview

1) Getting Started
2) Git Version Control Demo
   - Creating Branches
   - Committing Changes
   - Sharing Code with Others
3) Compiling GEOS-Chem Using Makefile Options
4) Downloading Data/Run Directories
5) Restart Files
6) Running GEOS-Chem
7) Visualizing Output with GAMAP
8) Q & A

GEOS-Chem Documentation

- GEOS-Chem Website:
  http://www.geos-chem.org/
- GEOS-Chem Public Wiki:
  http://wiki.geos-chem.org/
- GEOS-Chem Users' Guide:
  http://www.geos-chem.org/doc/man
- GAMAP Users' Guide:
  http://acmg.seas.harvard.edu/gamap/doc/

Getting Started

Make sure that you meet the minimum system requirements

Install the netCDF library

- GEOS-Chem v9-01-03 and higher versions require that you install netCDF on your computer system
- Instructions can be found on the Installing libraries for GEOS-Chem wiki page

Download the source code using Git:

```
git clone git://git.as.harvard.edy/bmy/GEOS-Chem Code.v9-01-03
```

See Ch. 2 of the GEOS-Chem manual for detailed info

Git Version Control Software

Git version control software is a distributed source code management system

- Created by Linus Torvalds to do the version control for Linux

Offers many improvements over previous source code management software (e.g. CVS, Subversion)

- Allows you to keep an identical copy (“clone”) of the GEOS-Chem source code repository on your own system
- Lets you easily combine code from several developers
- Contains easy-to-use graphical tools

GEOS-Chem source code, GEOS-Chem run directories, and GAMAP visualization software are distributed with Git

See Ch. 2 and Ch. 8 in GEOS-Chem manual for detailed info

Downloading and Updating

git clone

- Gives you a clean copy of the “master” repository
- You get source code files + total version history
- Use this to download a new version of the code

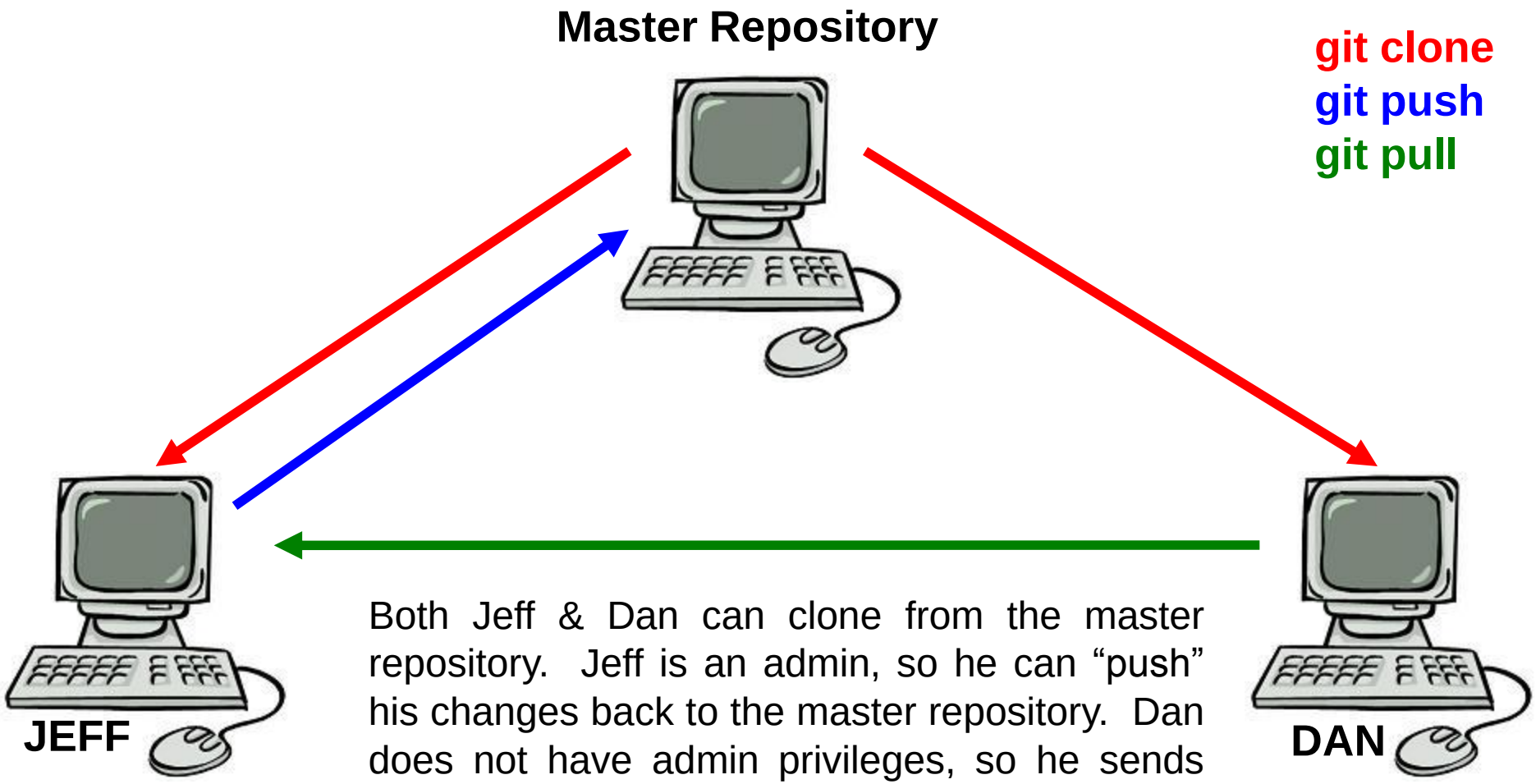
git pull

- Gets updates from the “master” repository and puts them into your existing local repository
- Use this to apply bug fixes (aka “patches”) to your code, or to get code from another user's local repository

git push

- Uploads code from your local repository to the “master” repository
- Typically this will only be used by people w/ admin privileges

Downloading and Updating



Both Jeff & Dan can clone from the master repository. Jeff is an admin, so he can “push” his changes back to the master repository. Dan does not have admin privileges, so he sends his updates to Jeff, who then “pushes” them to the master repository.

Git's Graphical Tools

gitk

- Lets you browse the complete revision history
- For each commit, you can see the changes made to each file
- You can see the branching history of the repository

git gui

- Graphical user interface for most Git commands
- Create new branches of development
- Commit (and document!) changes to the repository
- Merge branches back into the "master" line of development

NOTE: We recommend using the git gui rather than typing commands at the Unix prompt. The git gui greatly simplifies things.

Git's Branching Commands

Git allows you to split model development from the main line of development (aka "master branch") into one or more branches.

We recommend that you do not make code changes in the master branch. You can create a new branch in which to make updates.

Creating a branch

- In git gui: Branch / Create
- Branch is also checked out after it is created

Checking out a branch

- In git gui: Branch / Checkout menu
- The checkout command switches the files in your source code directory so that they correspond to the branch you requested
- To avoid confusion, you should always commit your updates to the current branch before checking out a new branch

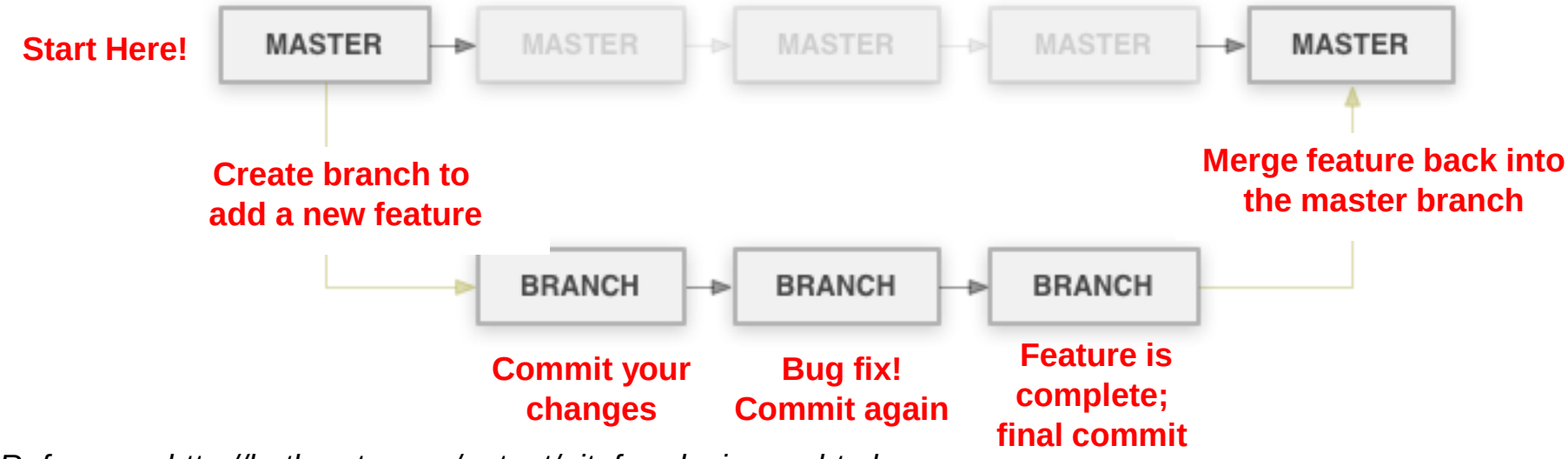
Git's Branching Commands

Merging branches together

- In git gui: Merge / Local Merge menu
- This will merge code from the branch that you pick into the currently-checked-out branch

Deleting branches

- In git gui: Branch/Delete menu
- Once you have merged a branch you can delete it



Reference: http://hoth.entp.com/output/git_for_designers.html

Committing Changes with Git

Committing is best done in the Git GUI

- ***Unstaged changes*** (top-left GUI window): Files w/ updates that Git does not know about yet. Click on them to stage.
- ***Staged changes*** (bottom-left GUI window): Files that will be added to the local repository when you commit.
- ***Commit message*** (lower-right GUI window): You can type a message describing the files/directories that are being committed. The first line of the message will show up as the commit title in the GitK browser.
- ***Sign-off button*** (lower-right GUI window): Click this button to add your name & email to the commit message.
- ***Commit button*** (lower-right GUI window): Click this button to commit your updated files/directories to the repository.
- ***Amend Last Commit*** (lower-right GUI window): Check this box if you want to update the message from the last commit.

Sharing Code with Others

Git pull

- When you can “see” the other repository on disk or via internet
- Recipient “pulls” code form sender’s repository

Send a patch file via email

- ***Patch file*** = file containing the “diffs”, or changes between commits
- The sender creates a patch file with this command at the Unix prompt:

```
git format-patch -2 --stdout > my_patch_file.txt
```

- The text file can be sent to the recipient by email
- To ingest changes into their local repository, the recipient will type this at the Unix prompt:

```
git am < my_patch_file.txt
```

*NOTE: For a patch file to work, you have to know its **parent commit** (i.e. the commit just prior to the point where the patch file was made).*

Good Coding Practice

The more comments, the better

- Copious comments to your source code will make sure that current and future GEOS-Chem users will be able to understand the modifications that you have added

Automatic documentation with ProTeX

- Header at the top of each subroutine, function, and module lists arguments, modification history, and references
- ProTeX is a perl script developed by NASA/GMAO that can be used to generate a GEOS-Chem reference manual
- Documentation is automatically generated when you compile with “make doc”

For more tips, see Appendix 7 of the GEOS-Chem manual

Compiling GEOS-Chem

Some important compiling commands:

- ***make help*** : shows help screen with all possible Makefile options
- ***make -j4***: compiles 4 files simultaneously (you can pick a different #)
- ***make clean***: Removes *.o *.mod files in source code subdirs only
- ***make realclean***: Removes all *.o *.mod *.a *.lib etc. files everywhere
- ***make -j4 DEBUG=yes TRACEBACK=yes***: Compile for use w/ debugger
- ***make -j4 BOUNDS=yes***: Compile w/ array-out-of-bounds checking
- ***make –j4 OMP=no***: Compile with OpenMP parallelization turned off
- ***make doc***: Builds the G-C reference docs with the ProTeX script
- ***make -j4 lib***: Only compiles the code but does not build executable
- ***make -j4 exe***: Builds the executable

See Ch. 3 of the GEOS-Chem manual for detailed info

Debugging Tips

- Check the GEOS-Chem wiki for any recent bug fixes
- Turn on the ND70 diagnostic to print debug output to the log file
- Compilation options to use:
  - ***BOUNDS=yes*** turns on array out-of-bounds error checking
  - ***TRACEBACK=yes*** prints out a list of routines that were called when the error occurred
- Use a debugger (Totalview, IDB, DBX)
- Isolate the error to a particular operation by turning on/off transport, chemistry, drydep, etc.
- Check for math errors
  - GC routine CHECK_STT checks for NaN or infinite values
  - GC routine SAFE_DIV avoids division by zero errors

Report any bugs in the standard code to the GC Support Team and post on the GC wiki for other users to see

Submitting Code Updates

- Test your code thoroughly and make sure that it works properly
  - Test other grid resolutions and offline simulations when applicable
- Contact the GEOS-Chem Support Team and request that your changed be included in the standard code
- When submitting updates be sure to include the following:
  - Source code
  - Data files
  - Documentation
- The GEOS-Chem Support Team asks that you provide a ***Git patch file*** containing your code modifications to minimize errors during code transfer process

Data Directories

The GEOS-Chem shared data directories contain various data that GEOS-Chem will read in during the course of a simulation:

- Meteorological data used to drive GEOS-Chem
- Emissions inventories for GEOS-Chem
- Other data

You may download the GEOS-Chem shared data directories from one of two archives:

1. Harvard archive (ftp.as.harvard.edu)

```
wget -r -nH "ftp://ftp.as.harvard.edu/gcgrid/geos-chem/data/DIRNAME/"
```

2. Dalhousie archive (http://rain.ucis.dal.ca)

```
wget -r -nH "ftp://rain.ucis.dal.ca/DIRNAME/”
```

Run Directories

Download via Git:

```
git clone git://git.as.harvard.edu/bmy/GEOS-Chem-rundirs/DIRNAME
```

Where DIRNAME is a combination such as:

```
4x5/geos5/standard
4x5/geos5/SOA
4x5/geos5/dicarbonyls, etc.
```

The most important input files in the run directories are ***input.geos*** (sets run options), ***globchem.dat***, and ***mglob.dat*** (chemical mechanism)

You also need to copy the executable file from the source code directory to the run directory

See Ch. 2 and Ch. 5 of the GEOS-Chem manual for detailed info

Setting Run Options

Check the following settings in your ***input.geos*** file:

- Start and end dates of run
- Directories where data files are located
- Restart file names
- Transport options
- Convection options
- Emissions options
- Deposition options
- Chemistry options
- Diagnostic options
- Timeseries diagnostic options
- Nested-grid options (if necessary)

Restart Files

Availability

- Each of the GEOS-Chem run directories contains a sample restart file
- GAMAP routine ***make_restart.pro*** can be used to create a “fake” restart file
- We highly recommend running GEOS-Chem for at least one year to generate a spun-up restart file for your simulations

Regridding

- Restart files may be regridded or cropped using the following GAMAP routines:
  - regridh_restart.pro
  - create_nested.pro

Visualizing Output with GAMAP

Use ***GAMAP*** for most GEOS-Chem output visualization tasks

- Easy-to-use interactive interface fulfills basic plotting needs
- See http://acmg.seas.harvard.edu/gamap for more info

GAMAP subroutines can be used as standalones in IDL:

- ***CTM_GET_DATA:*** gets several data blocks
- ***CTM_GET_DATABLOCK:*** gets 1 data block; cuts it to size
- ***TVMAP:*** plots data on a lat-lon world map
- ***TVPLOT:*** plots lat-alt or lon-alt data
- ***CTM_PLOT:*** reads data from disk and plots it
- ***CTM_SUM_EMISSIONS:*** sums up emissions totals

Join GEOS-Chem Working Groups

All GEOS-Chem users are invited to join the Working Groups that corresponds to their own particular areas of research

Users are also encouraged to sign up for the relevant WG email list and add their project to the group's page on the GEOS-Chem wiki

- Adjoint Model and Data Assimilation: geos-chem-adjoint@seas.harvard.edu
- Aerosols: geos-chem-aerosols@seas.harvard.edu
- Carbon Gases and Organics: geos-chem-carbon@seas.harvard.edu
- Chemistry–Climate: geos-chem-climate@seas.harvard.edu
- Hg and POPs: geos-chem-hg-pop@seas.harvard.edu
- Oxidants and Chemistry: geos-chem-oxidants@seas.harvard.edu
- Nested Model: geos-chem-regional@seas.harvard.edu
- Sources and Sinks: geos-chem-emissions@seas.harvard.edu

Q & A

Topics that you want to discuss?

Demos that you'd like to see?