Getting Started with High Performance GEOS-Chem

Lizzie Lundgren
GEOS-Chem Support Team
geos-chem-support@as.harvard.edu

June 2017

Overview

What is GCHP and why use it?

- GCHP features the same science as GEOS-Chem using the standard "classic" capability except:
  - It operates on a cubed-sphere grid
  - It is parallelized using a message-passing interface (MPI) implementation
- GCHP improves upon GCC by:
  - Enabling more accurate transport
  - Providing efficient scaling across many cores and multiple nodes

Common Misconceptions about GCHP

- I need a high performance compute cluster for GCHP
  - Not true! You can run GCHP on as little as one machine with 6 cores.
- I can only perform high-resolution runs with GCHP
  - Not true! GCHP can run with c24, the cubed-sphere equivalent of 4°x5°.
- I need met fields at the same resolution as my run
  - Not true! GCHP can use 2°x2.5° met fields for up to at least c180 (0.5° res), although we recommend keeping the met resolution to no more than twice your run resolution equivalent to ensure quality output. If the met wind fields are too coarse relative to your internal resolution then polar divergence will occur.

Useful Tip #1: Grid Resolutions

- Cubed-sphere resolution “cN” means each of the six faces are divided into N x N segments.
- An easy rule-of-thumb for resolution mapping is to divide 90 by N to determine the approximate lat-lon degree resolution.

Standard lat-lon resolution	**Approximate CS equivalent(s)**
4° x 5°	c24
2° x 2.5°	c48, c45
1° x 1.25°	c96, c90
0.5° x 0.625° [1]	c192, c180
0.25° x 0.3125° [2]	c384, c360
0.125° x 0.15625°	c720 [3]

[1] Native resolution of MERRA-2 product from GMAO
[2] Native resolution of GEOS-FP product from GMAO
[3] Native cubed-sphere resolution of GEOS-5

Useful Tip #2: Resource Allocation

- Number of nodes and number of faces are independent
- Number of cores and number of faces are NOT independent
- Total number of cores must be divisible by 6!
- How does it work?
  - Each NxN face is divided into NX x NY/6 segments, each comprised of approximately N/NX x N*6/NY cubed-sphere grid cells.
  - Each segment is processed by a single core
  - NX * NY must therefore equal the total number of cores
  - NX * NY/6 would ideally be square to minimize required communication between cores
- NX and NY are manually set in config file GCHP.rc but are over-written by NX and NY of your choosing in utility script runConfig.rc. Setting them will soon be automatic.
- More on this topic later in the presentation.

GCHP Tutorial

1. Downloading Source Code
2. Create a Run Directory
3. Load Environment
4. Compiling GCHP
5. Configure Run
6. Run a Simulation
7. Analyze Output Data
8. Reusing a Run Directory

Step 1: Download Source Code

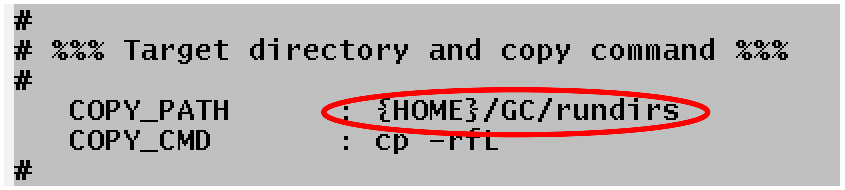
- You need two repositories for GCHP:
  1. GEOS-Chem “classic” (GCC) code as your main directory
  2. GCHP code as a subdirectory within it

- Use the GC and GCHP master branches

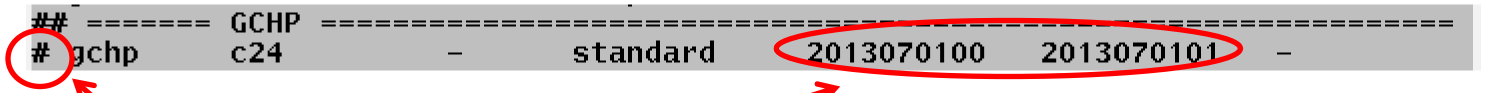
```
git clone -b master https://bitbucket.org/gcst/gc_bleeding_edge Code.v11-02_gchp
cd Code.v11-02_gchp
git clone -b master https://bitbucket.org/gcst/gchp GCHP
```

Step 2: Create a Run Directory

- Download a GCHP run directory from GEOS-Chem Unit Tester
- Use the UT_Bleeding_Edge master branch
  - If you have the repository already, check out the branch:
    - `git pull`
    - `git checkout master`
  - If you do not have the repository:
    - `git clone -b master https:/bitbucket.org/gcst/ut_bleeding_edge UT`
- Run directory set up for c24 (~4°x5°), 1 hour, standard simulation
- To download, modify UT/perl/CopyRunDirs.input:



```
#
# %%% Target directory and copy command %%%
#
   COPY_PATH          : {HOME}/GC/rundirs
   COPY_CMD           : cp -rfL
#
```



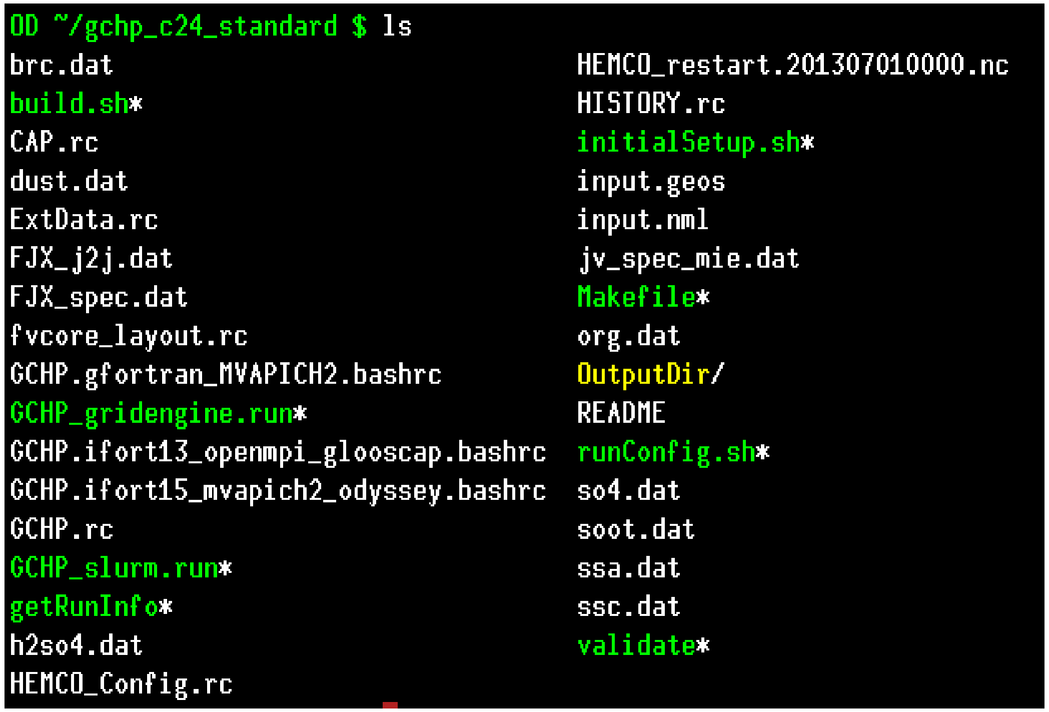
```
## ======= GCHP ============================================================
# gchp     c24         -        standard     2013070100    2013070101    -
```

Uncomment
(delete)

Ignore
(not yet functional)

GCHP Run Directory: Out-of-the-box Contents

1. Config files
2. Standard sim `*.dat` files
3. Sample `.bashrc` files
4. Sample run scripts
5. Utility bash scripts
6. Output data subdirectory
7. README
8. Makefile
9. Files to ignore:
   – `getRunInfo`
   – `input.nml`
   – HEMCO restart file (not used by GCHP)



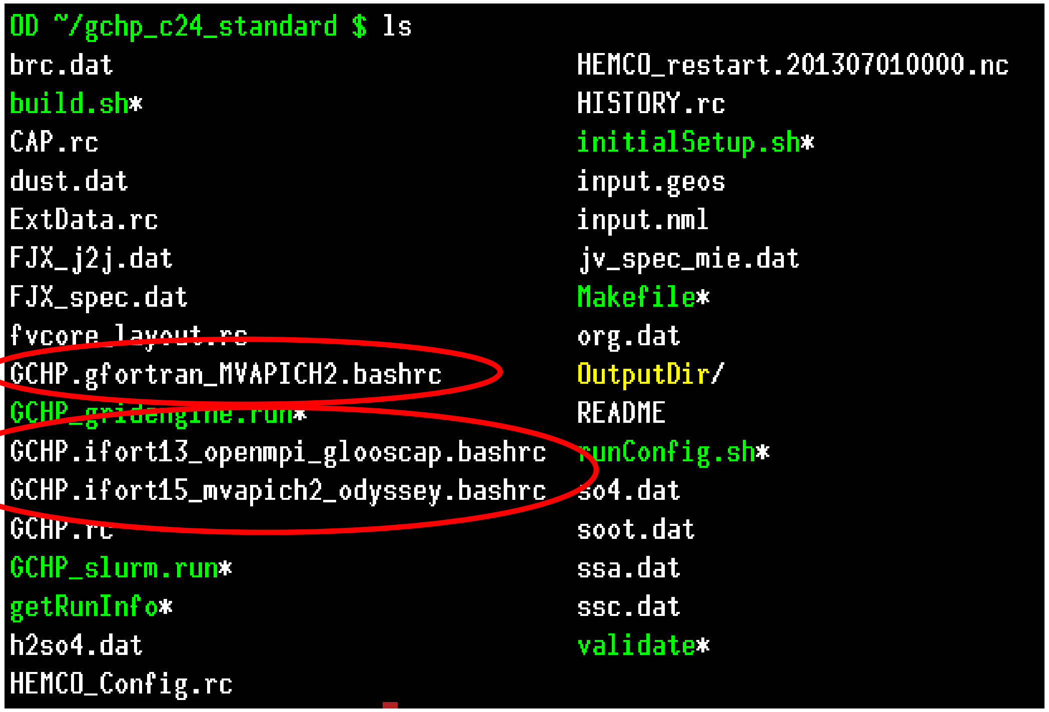
```
OD ~/gchp_c24_standard $ ls
brc.dat                             HEMCO_restart.201307010000.nc
build.sh*                           HISTORY.rc
CAP.rc                              initialSetup.sh*
dust.dat                            input.geos
ExtData.rc                          input.nml
FJX_j2j.dat                         jv_spec_mie.dat
FJX_spec.dat                        Makefile*
fvcore_layout.rc                    org.dat
GCHP.gfortran_MVAPICH2.bashrc       OutputDir/
GCHP_gridengine.run*                README
GCHP.ifort13_openmpi_glooscap.bashrc  runConfig.sh*
GCHP.ifort15_mvapich2_odyssey.bashrc  so4.dat
GCHP.rc                             soot.dat
GCHP_slurm.run*                     ssa.dat
getRunInfo*                         ssc.dat
h2so4.dat                           validate*
HEMCO_Config.rc
```

WARNING: do not use the GCHP run directory out-of-the-box! Initial setup is required (more on this later).

GCHP Run Directory: .bashrc Files

Three examples provided:

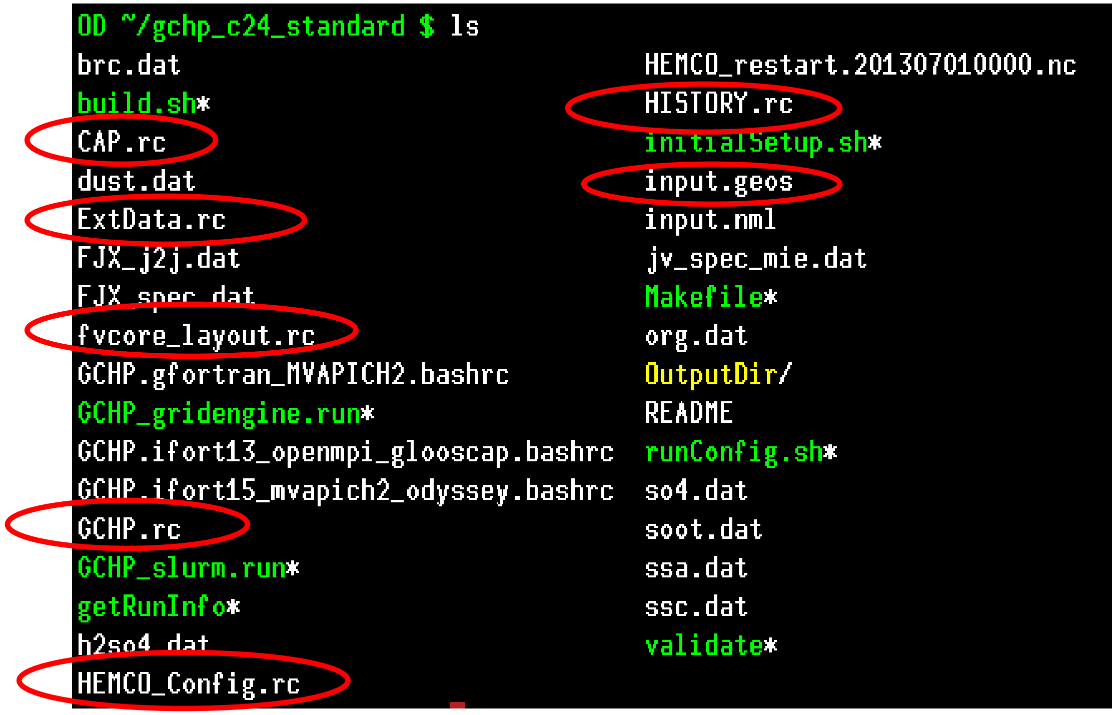
- For Odyssey (Harvard):
  - ifort15, MVAPICH2
  - gfortran, MVAPICH2
- For Glooscap (Dalhousie):
  - ifort13, OpenMPI
- Other Systems
  - Use these as examples to build your own



```
OD ~/gchp_c24_standard $ ls
brc.dat                               HEMCO_restart.201307010000.nc
build.sh*                             HISTORY.rc
CAP.rc                                initialSetup.sh*
dust.dat                              input.geos
ExtData.rc                            input.nml
FJX_j2j.dat                           jv_spec_mie.dat
FJX_spec.dat                          Makefile*
fvcore_layout.rc                      org.dat
GCHP.gfortran_MVAPICH2.bashrc         OutputDir/
GCHP_gridengine.run*                  README
GCHP.ifort13_openmpi_glooscap.bashrc  runConfig.sh*
GCHP.ifort15_mvapich2_odyssey.bashrc  so4.dat
GCHP.rc                               soot.dat
GCHP_slurm.run*                       ssa.dat
getRunInfo*                           ssc.dat
h2so4.dat                             validate*
HEMCO_Config.rc
```

GCHP Run Directory: Config Files

- GCHP Specific:
  - **`Cap.rc`**
    - start/end dates, and more
  - **`ExtData.rc`**
    - external data information
  - **`fvcore_layout.rc`**
    - transport-related settings
  - **`GCHP.rc`**
    - general settings
  - **`HISTORY.rc`**
    - output data settings
- Same as GCC:
  - **`input.geos`**
  - **`HEMCO_Config.rc`**
- **WARNINGS:**
  - Not all fields in **`input.geos`** and **`HEMCO_Config.rc`** are used.
  - Some settings must be set in multiple files (**use runConfig.sh for sanity!!!**)



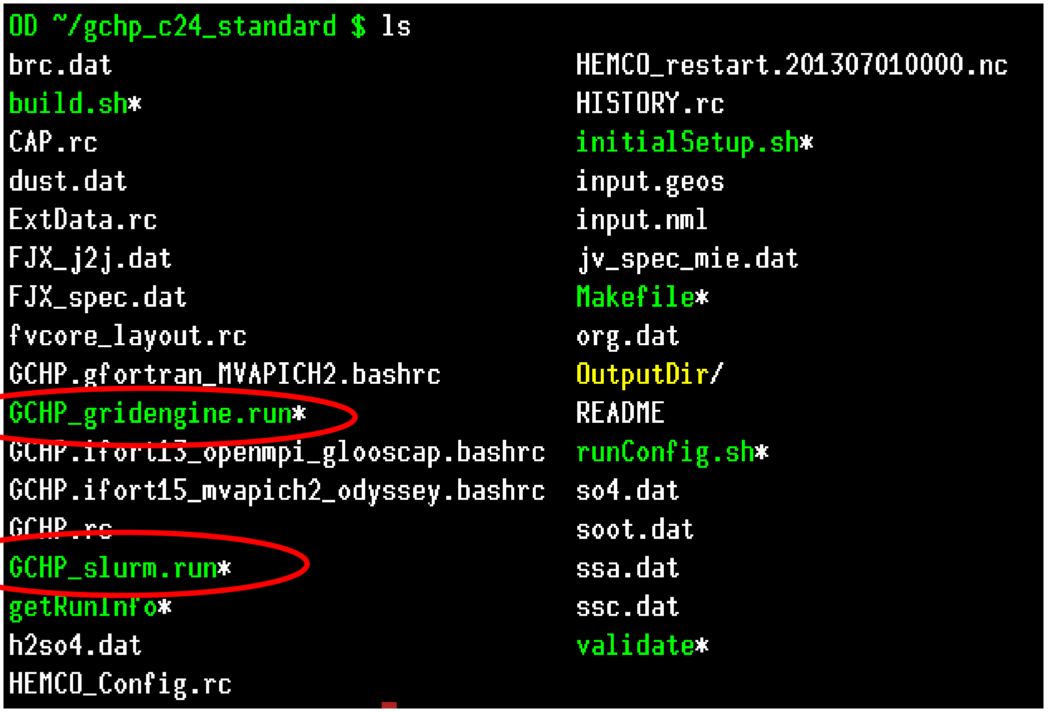
```
OD ~/gchp_c24_standard $ ls
brc.dat                                HEMCO_restart.201307010000.nc
build.sh*                              HISTORY.rc
CAP.rc                                 initialSetup.sh*
dust.dat                               input.geos
ExtData.rc                             input.nml
FJX_j2j.dat                            jv_spec_mie.dat
FJX_spec.dat                           Makefile*
fvcore_layout.rc                       org.dat
GCHP.gfortran_MVAPICH2.bashrc          OutputDir/
GCHP_gridengine.run*                   README
GCHP.ifort13_openmpi_glooscap.bashrc   runConfig.sh*
GCHP.ifort15_mvapich2_odyssey.bashrc   so4.dat
GCHP.rc                                soot.dat
GCHP_slurm.run*                        ssa.dat
getRunInfo*                            ssc.dat
h2so4.dat                              validate*
HEMCO_Config.rc
```

> 90% of GCHP errors are due to incorrect or inconsistent config file settings. Use bash script **`runConfig.sh`** to avoid common errors.

GCHP Run Directory: Run Scripts

Two examples provided:

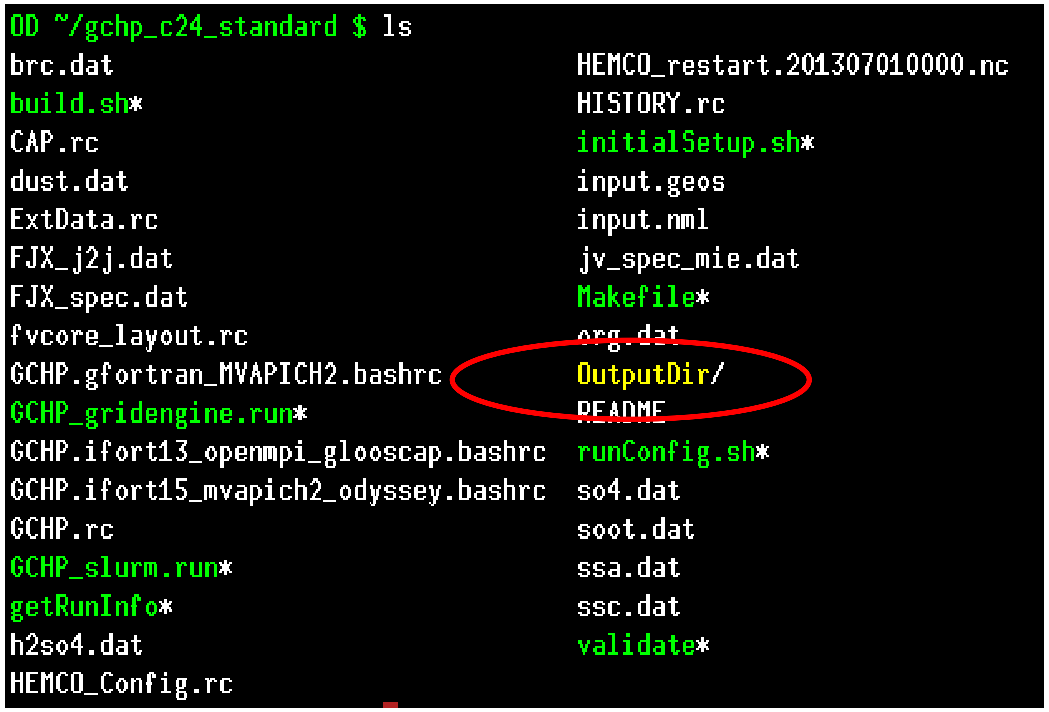
- For Odyssey (Harvard):
  - **`GCHP_slurm.run`**
- For Glooscap (Dalhousie):
  - **`GCHP_gridengine.run`**
- Other Systems
  - Use these as examples to build your own



```
OD ~/gchp_c24_standard $ ls
brc.dat                                 HEMCO_restart.201307010000.nc
build.sh*                               HISTORY.rc
CAP.rc                                  initialSetup.sh*
dust.dat                                input.geos
ExtData.rc                              input.nml
FJX_j2j.dat                             jv_spec_mie.dat
FJX_spec.dat                            Makefile*
fvcore_layout.rc                        org.dat
GCHP.gfortran_MVAPICH2.bashrc           OutputDir/
GCHP_gridengine.run*                    README
GCHP.ifort13_openmpi_glooscap.bashrc    runConfig.sh*
GCHP.ifort15_mvapich2_odyssey.bashrc    so4.dat
GCHP.rc                                 soot.dat
GCHP_slurm.run*                         ssa.dat
getRunInfo*                             ssc.dat
h2so4.dat                               validate*
HEMCO_Config.rc
```

GCHP Run Directory: Output Data Storage

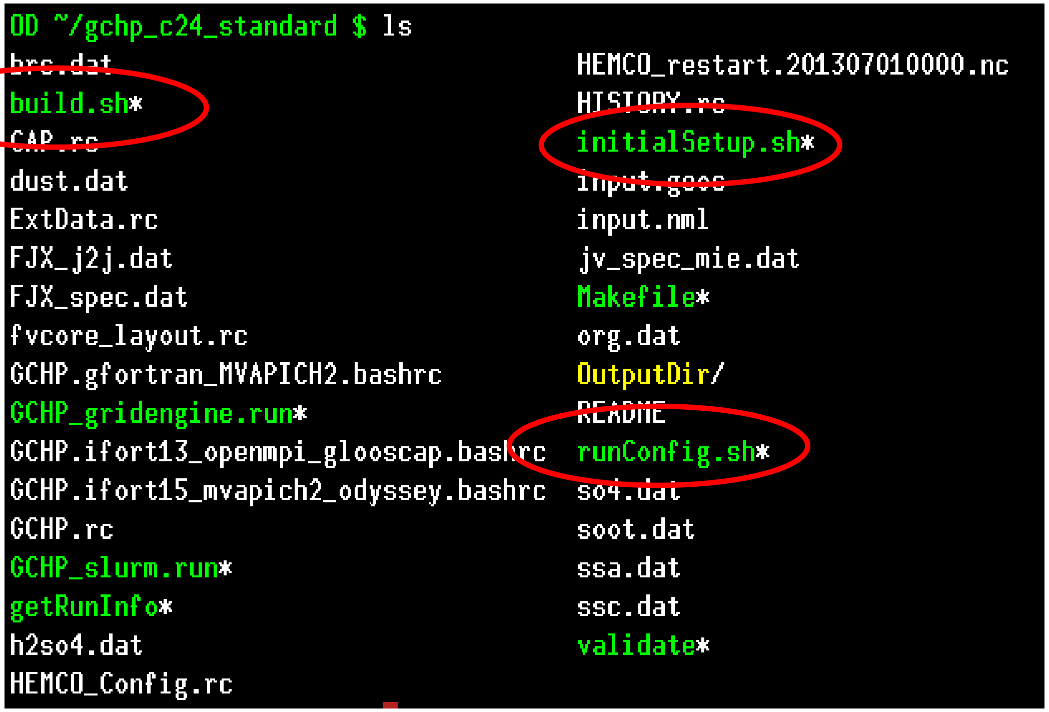
- **`OutputDir/`**
  - All GCHP output data configured in **`HISTORY.rc`** are saved here
  - Restart file is not saved here
  - Do not remove or rename! GCHP will hang without a helpful error message



```
OD ~/gchp_c24_standard $ ls
brc.dat                               HEMCO_restart.201307010000.nc
build.sh*                             HISTORY.rc
CAP.rc                                initialSetup.sh*
dust.dat                              input.geos
ExtData.rc                            input.nml
FJX_j2j.dat                           jv_spec_mie.dat
FJX_spec.dat                          Makefile*
fvcore_layout.rc                      org.dat
GCHP.gfortran_MVAPICH2.bashrc         OutputDir/
GCHP_gridengine.run*                  README
GCHP.ifort13_openmpi_glooscap.bashrc  runConfig.sh*
GCHP.ifort15_mvapich2_odyssey.bashrc  so4.dat
GCHP.rc                               soot.dat
GCHP_slurm.run*                       ssa.dat
getRunInfo*                           ssc.dat
h2so4.dat                             validate*
HEMCO_Config.rc
```

GCHP Run Directory: Utility Scripts

- **`initialSetup.sh`**
  - creates symlinks to data
  - IMPORTANT: run once after rundir download
- **`build.sh`**
  - cleans and compiles code
  - executed in Makefile
- **`runConfig.sh`**
  - single location to update common run settings
  - overwrites config files
  - executed in run scripts



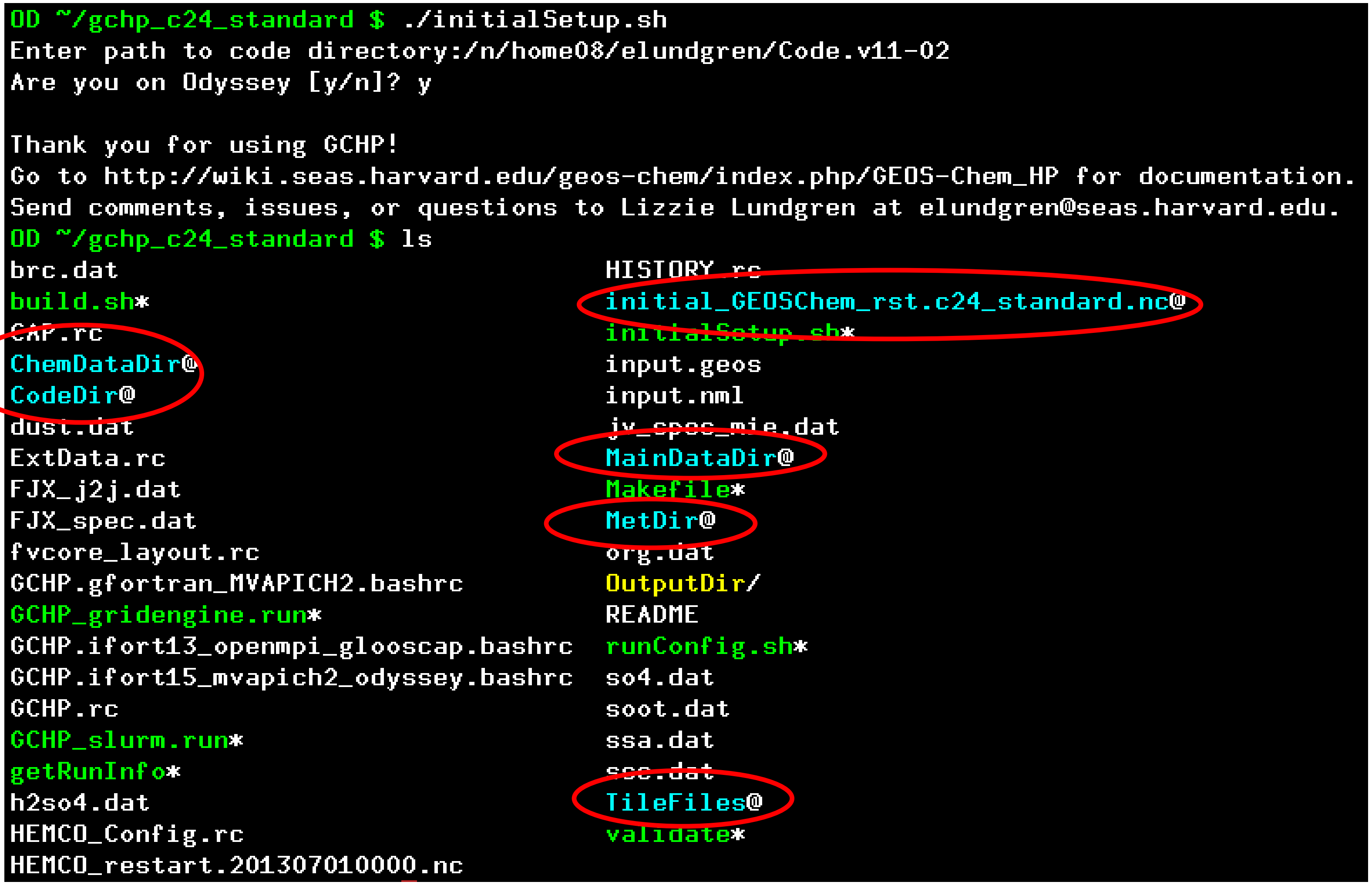
```
OD ~/gchp_c24_standard $ ls
brc.dat                                  HEMCO_restart.201307010000.nc
build.sh*                                HISTORY.rc
CAP.rc                                   initialSetup.sh*
dust.dat                                 input.geos
ExtData.rc                               input.nml
FJX_j2j.dat                              jv_spec_mie.dat
FJX_spec.dat                             Makefile*
fvcore_layout.rc                         org.dat
GCHP.gfortran_MVAPICH2.bashrc            OutputDir/
GCHP_gridengine.run*                     README
GCHP.ifort13_openmpi_glooscap.bashrc     runConfig.sh*
GCHP.ifort15_mvapich2_odyssey.bashrc     so4.dat
GCHP.rc                                  soot.dat
GCHP_slurm.run*                          ssa.dat
getRunInfo*                              ssc.dat
h2so4.dat                                validate*
HEMCO_Config.rc
```

Step 3: One-time Run Directory Setup

- One-time setup of your run directory after downloading is required
- Run bash shell script **`initialSetup.sh`** to set symbolic links:
  - You will be prompted for your source code location (set as symlink **`CodeDir`**)
  - The rest is automatically set for you if on Odyssey (do manually elsewhere)
    - **`ChemDataDir`** – ExtData/CHEM_INPUTS
    - **`MainDataDir`** – ExtData/HEMCO
    - **`MetDir`** – meteorology data
    - **`TileFiles`** – ExtData/GCHP/TileFiles
    - **`initial_GEOSChem_rst.c24_standard.nc`** – GCHP restart file at c24 (cubed-sphere equivalent of 4°x5°)
- Things to note:
  - Use path for your GC base code and not the GCHP subdirectory
  - Do not include symbolic links in your source code path
  - Unlike GCC, do not edit the Makefile with your source code path
  - Config files assume **`MetDir`** points to 2°x2.5° GEOS-FP meteorology

Step 3: One-time Run Directory Setup

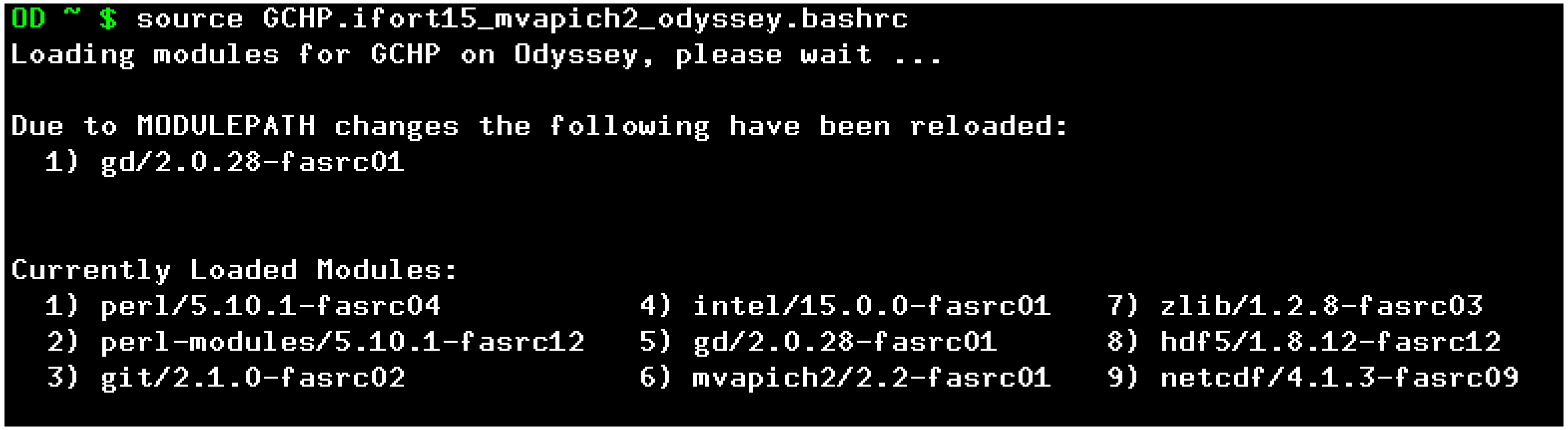
- Following initial setup, your run directory should look like this:



```
OD ~/gchp_c24_standard $ ./initialSetup.sh
Enter path to code directory:/n/home08/elundgren/Code.v11-02
Are you on Odyssey [y/n]? y

Thank you for using GCHP!
Go to http://wiki.seas.harvard.edu/geos-chem/index.php/GEOS-Chem_HP for documentation.
Send comments, issues, or questions to Lizzie Lundgren at elundgren@seas.harvard.edu.
OD ~/gchp_c24_standard $ ls
brc.dat                               HISTORY.rc
build.sh*                             initial_GEOSChem_rst.c24_standard.nc@
CAP.rc                                initialSetup.sh*
ChemDataDir@                          input.geos
CodeDir@                              input.nml
dust.dat                              jv_spec_mie.dat
ExtData.rc                            MainDataDir@
FJX_j2j.dat                           Makefile*
FJX_spec.dat                          MetDir@
fvcore_layout.rc                      org.dat
GCHP.gfortran_MVAPICH2.bashrc         OutputDir/
GCHP_gridengine.run*                  README
GCHP.ifort13_openmpi_glooscap.bashrc  runConfig.sh*
GCHP.ifort15_mvapich2_odyssey.bashrc  so4.dat
GCHP.rc                               soot.dat
GCHP_slurm.run*                       ssa.dat
getRunInfo*                           ssc.dat
h2so4.dat                             TileFiles@
HEMCO_Config.rc                       validate*
HEMCO_restart.201307010000.nc
```

Step 4: Load GCHP Environment

- Set up your environment prior to compiling and/or running
- On Odyssey:



```
OD ~ $ source GCHP.ifort15_mvapich2_odyssey.bashrc
Loading modules for GCHP on Odyssey, please wait ...

Due to MODULEPATH changes the following have been reloaded:
  1) gd/2.0.28-fasrc01


Currently Loaded Modules:
  1) perl/5.10.1-fasrc04          4) intel/15.0.0-fasrc01   7) zlib/1.2.8-fasrc03
  2) perl-modules/5.10.1-fasrc12  5) gd/2.0.28-fasrc01      8) hdf5/1.8.12-fasrc12
  3) git/2.1.0-fasrc02            6) mvapich2/2.2-fasrc01   9) netcdf/4.1.3-fasrc09
```

- Elsewhere:
  - Create a **`.bashrc`** file based on sample files in the run directory
  - Using the libraries above is recommended but other combos are possible
    - OpenMPI or Intel MPI
    - Gfortran
    - Other NetCDF library versions

Step 5: Compile GCHP

- Like GCC, compile GCHP from the run directory using the Makefile

- First time compilation (30-60 min): **`make clean_compile`**
  - Warnings, error messages, and pauses are normal
  - Signs of successful compilation:
    - **`“### GCHP compiled Successfully ###”`**
    - The following files exist:
      - **`GCHP/ESMF/esmf.install`**
      - **`GCHP/FVdycoreCubed_GridComp/fvdycore.install`**
      - **`GCHP/Shared/mapl.install`**

- Subsequent compilation: **`make clean_standard`**
  - For updates to GC base code or GCHP top-level directory
  - Not for updates to GCHP subdirectories (e.g. GCHP/Shared)

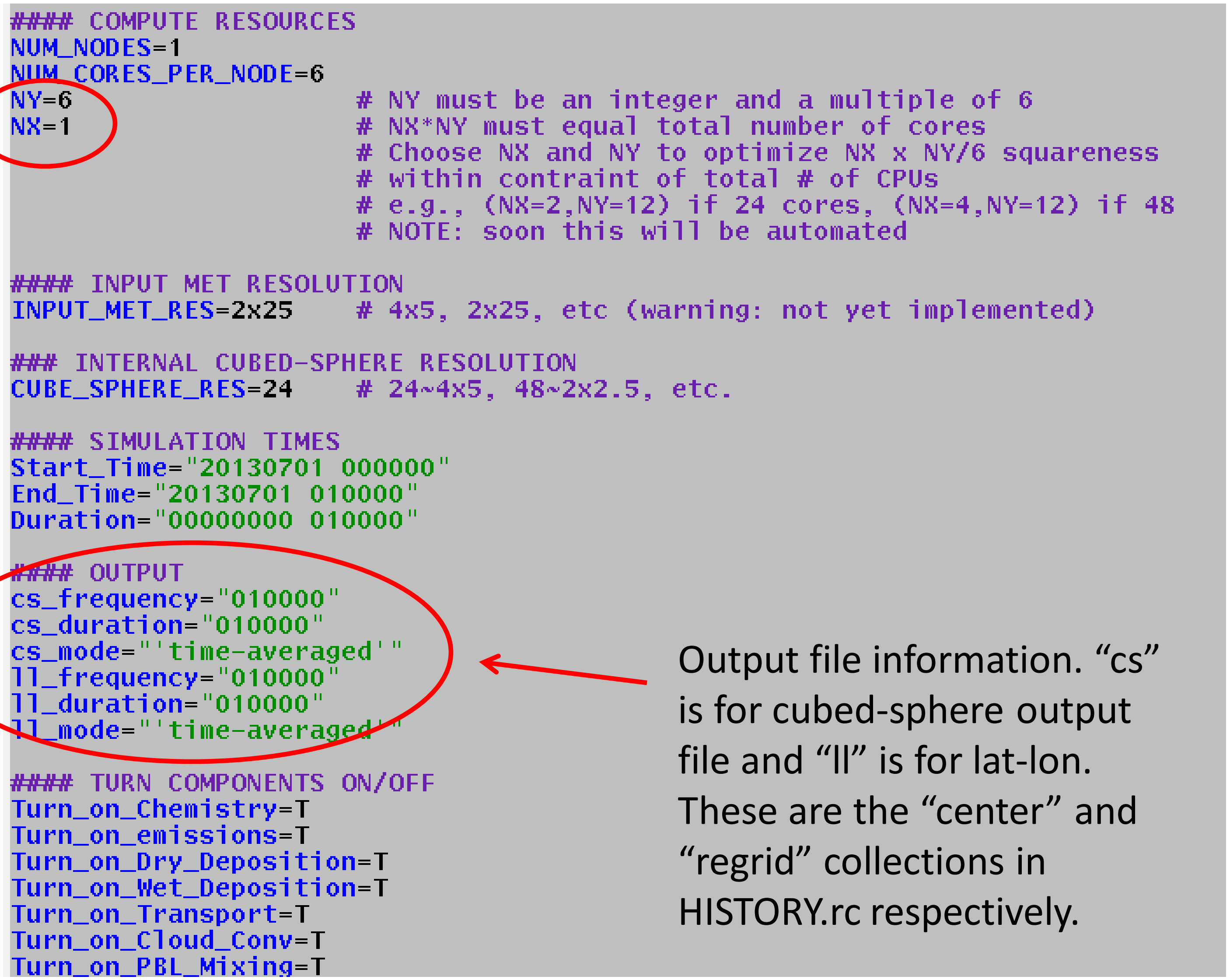
Step 6: Configure Run

- Use utility bash script **`runConfig.sh`** for select config settings

- If there is a setting you don't see in **`runConfig.sh`** (e.g. list of variables to include in output file set in **`HISTORY.rc`**) then you need to manually change it in the appropriate config file.

- Things to note about using **`runConfig.sh`**
  - Overwrites **`input.geos`** and **`*.rc`** files
  - Sample run scripts execute **`runConfig.sh`** prior to executing **`geos`**
  - Run scripts send summary of **`runConfig.sh`** settings to **`runConfig.log`**
  - **`HEMCO_Config.rc`** settings are not currently in **`runConfig.sh`**
  - Currently in development and design may change in the future!

runConfig.sh: Default Settings Part 1

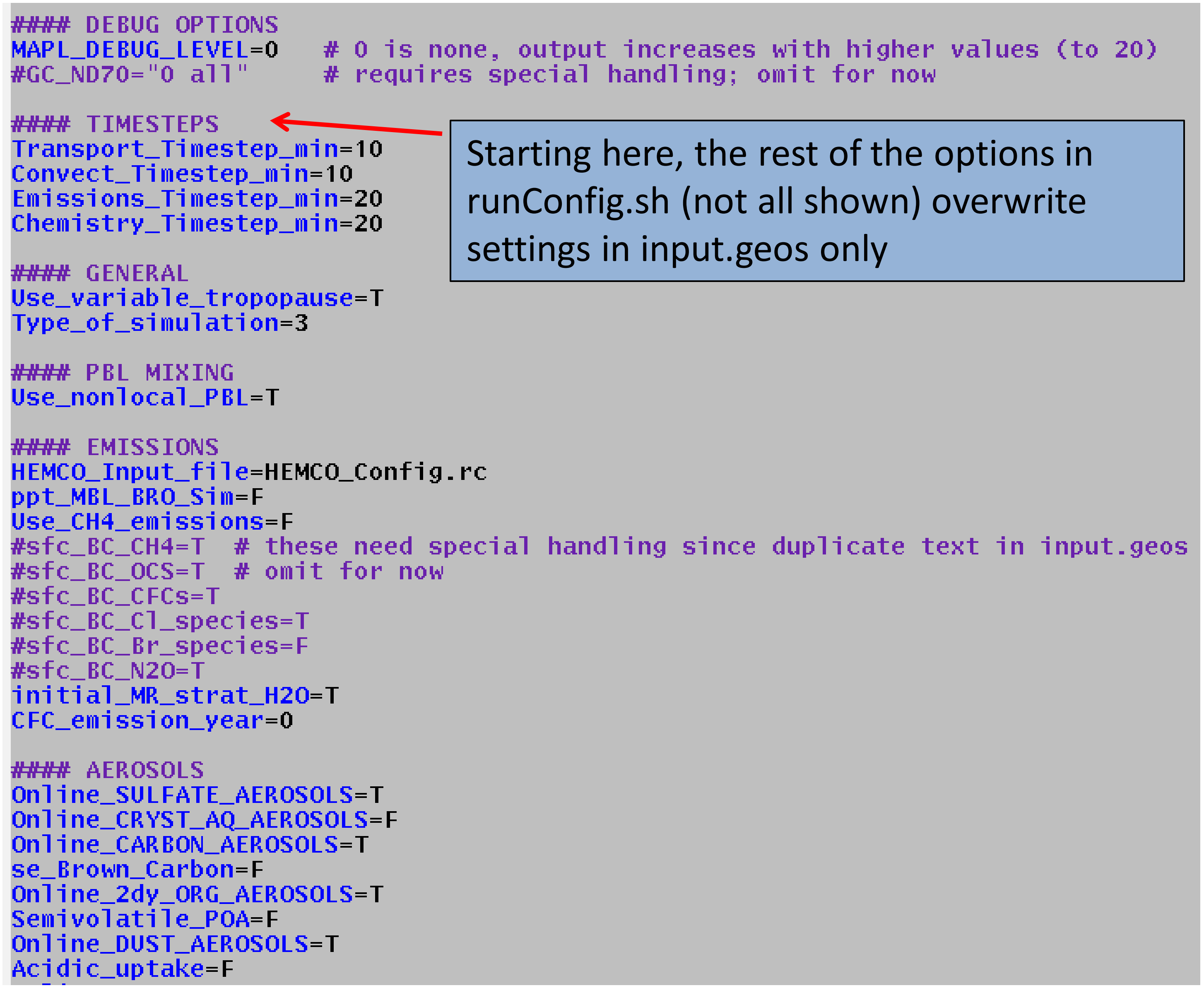


```
#### COMPUTE RESOURCES
NUM_NODES=1
NUM_CORES_PER_NODE=6
NY=6                           # NY must be an integer and a multiple of 6
NX=1                           # NX*NY must equal total number of cores
                               # Choose NX and NY to optimize NX x NY/6 squareness
                               # within contraint of total # of CPUs
                               # e.g., (NX=2,NY=12) if 24 cores, (NX=4,NY=12) if 48
                               # NOTE: soon this will be automated

#### INPUT MET RESOLUTION
INPUT_MET_RES=2x25             # 4x5, 2x25, etc (warning: not yet implemented)

### INTERNAL CUBED-SPHERE RESOLUTION
CUBE_SPHERE_RES=24             # 24~4x5, 48~2x2.5, etc.

#### SIMULATION TIMES
Start_Time="20130701 000000"
End_Time="20130701 010000"
Duration="00000000 010000"

#### OUTPUT
cs_frequency="010000"
cs_duration="010000"
cs_mode="'time-averaged'"
ll_frequency="010000"
ll_duration="010000"
ll_mode="'time-averaged'"

#### TURN COMPONENTS ON/OFF
Turn_on_Chemistry=T
Turn_on_emissions=T
Turn_on_Dry_Deposition=T
Turn_on_Wet_Deposition=T
Turn_on_Transport=T
Turn_on_Cloud_Conv=T
Turn_on_PBL_Mixing=T
```

See Useful Tip #2 at start of slides

Output file information. "cs" is for cubed-sphere output file and "ll" is for lat-lon. These are the "center" and "regrid" collections in HISTORY.rc respectively.

runConfig.sh: Default Settings Part 2

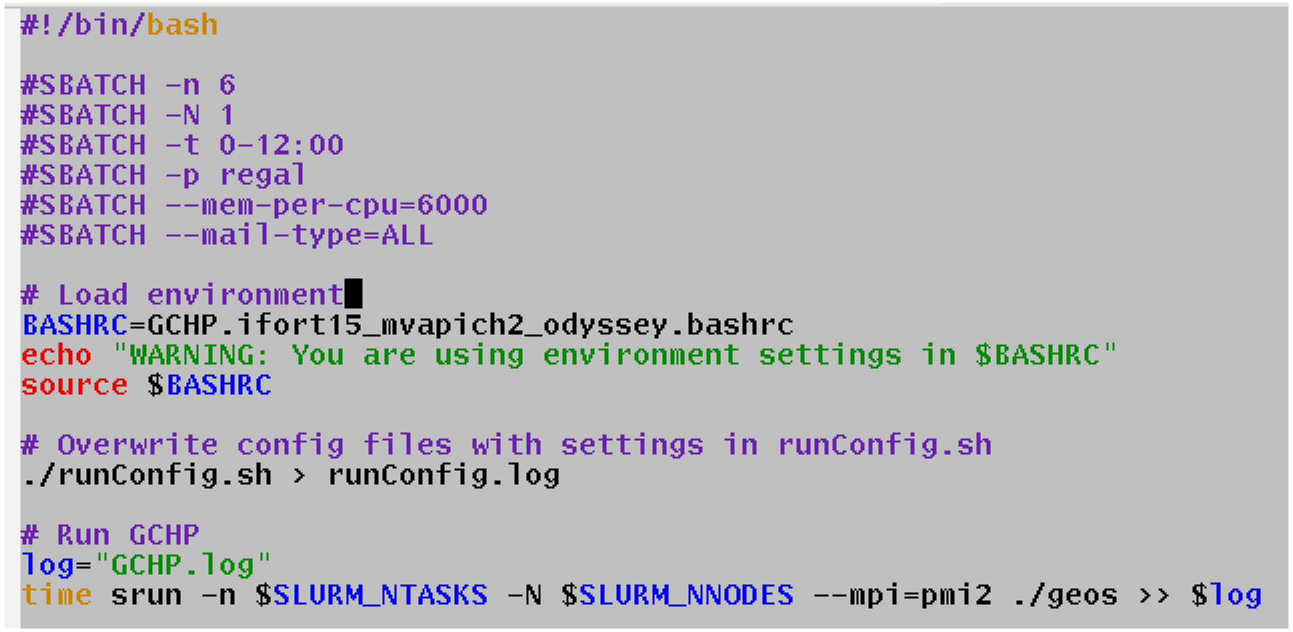


```
#### DEBUG OPTIONS
MAPL_DEBUG_LEVEL=0   # 0 is none, output increases with higher values (to 20)
#GC_ND70="0 all"     # requires special handling; omit for now

#### TIMESTEPS
Transport_Timestep_min=10
Convect_Timestep_min=10
Emissions_Timestep_min=20
Chemistry_Timestep_min=20

#### GENERAL
Use_variable_tropopause=T
Type_of_simulation=3

#### PBL MIXING
Use_nonlocal_PBL=T

#### EMISSIONS
HEMCO_Input_file=HEMCO_Config.rc
ppt_MBL_BRO_Sim=F
Use_CH4_emissions=F
#sfc_BC_CH4=T  # these need special handling since duplicate text in input.geos
#sfc_BC_OCS=T  # omit for now
#sfc_BC_CFCs=T
#sfc_BC_Cl_species=T
#sfc_BC_Br_species=F
#sfc_BC_N2O=T
initial_MR_strat_H2O=T
CFC_emission_year=0

#### AEROSOLS
Online_SULFATE_AEROSOLS=T
Online_CRYST_AQ_AEROSOLS=F
Online_CARBON_AEROSOLS=T
se_Brown_Carbon=F
Online_2dy_ORG_AEROSOLS=T
Semivolatile_POA=F
Online_DUST_AEROSOLS=T
Acidic_uptake=F
```

Starting here, the rest of the options in runConfig.sh (not all shown) overwrite settings in input.geos only

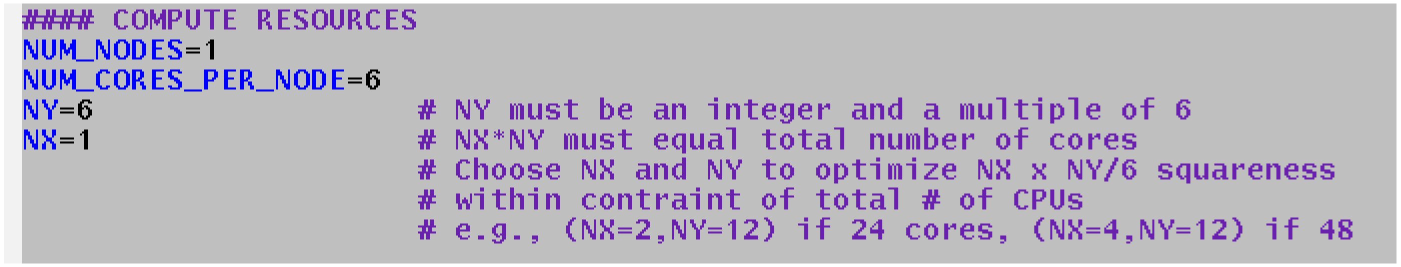
Step 7: Run GCHP (single node)

- Submit GCHP to slurm using a run script
- The most basic test is 6 cores on 1 node:



```
#!/bin/bash

#SBATCH -n 6
#SBATCH -N 1
#SBATCH -t 0-12:00
#SBATCH -p regal
#SBATCH --mem-per-cpu=6000
#SBATCH --mail-type=ALL

# Load environment
BASHRC=GCHP.ifort15_mvapich2_odyssey.bashrc
echo "WARNING: You are using environment settings in $BASHRC"
source $BASHRC

# Overwrite config files with settings in runConfig.sh
./runConfig.sh > runConfig.log

# Run GCHP
log="GCHP.log"
time srun -n $SLURM_NTASKS -N $SLURM_NNODES --mpi=pmi2 ./geos >> $log
```

- This requires the same compute resources set in **`runConfig.rc`**



```
#### COMPUTE RESOURCES
NUM_NODES=1
NUM_CORES_PER_NODE=6
NY=6                  # NY must be an integer and a multiple of 6
NX=1                  # NX*NY must equal total number of cores
                      # Choose NX and NY to optimize NX x NY/6 squareness
                      # within contraint of total # of CPUs
                      # e.g., (NX=2,NY=12) if 24 cores, (NX=4,NY=12) if 48
```

Step 8: Analyze Output

- All GCHP output is in netCDF-4 format (hurray!)
- Three outputs:
  - Restart file
    - Stored in top-level of run directory
    - Filename: **`-gcchem_internal_checkpoint_c24.nc`** (configured in **`GCHP.rc`**)
    - Cubed-sphere grid
  - **`OutputDir/GCHP.regrid.YYYYMMDD.nc4`**
    - "regrid" collection configured in **`HISTORY.rc`**
    - Analogous to ND45 in GCC (species concentration diagnostic on lat/lon grid)
    - Not conservatively regridded from cubed-sphere and so we do not recommend using this data
  - **`OutputDir/GCHP.center.YYYYMMDD.nc4`**
    - "center" collection configured in **`HISTORY.rc`**
    - On the cubed-sphere grid at the run resolution and thus superior to "regrid"
    - Can be regridded from cubed-sphere to lat/lon using either of the following tools:
      - CSGrid Matlab package (https://bitbucket.org/gcst/csgrid)
      - GCPy Python package (https://bitbucket.org/gcst/gcpy)

Step 9: Rerunning

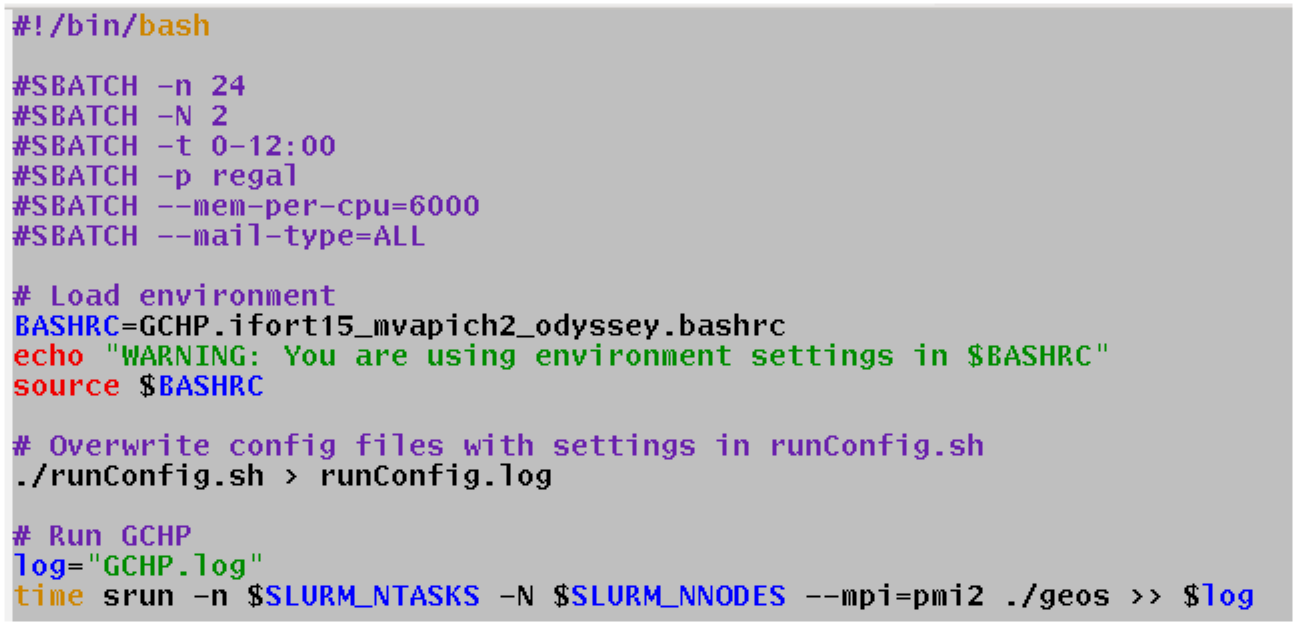
- You can reuse your GCHP run directory but MUST do the following prior to rerunning to avoid a seg fault: **`make cleanup_output`**

- Experiment with different run settings in **`runConfig.sh`**

- If changing # of cores and/or # of nodes:
  - Remember to update **`runConfig.sh`** as well as your run script
  - Choose NX and NY such that NX by NY/6 is roughly square
  - See next slide for an example

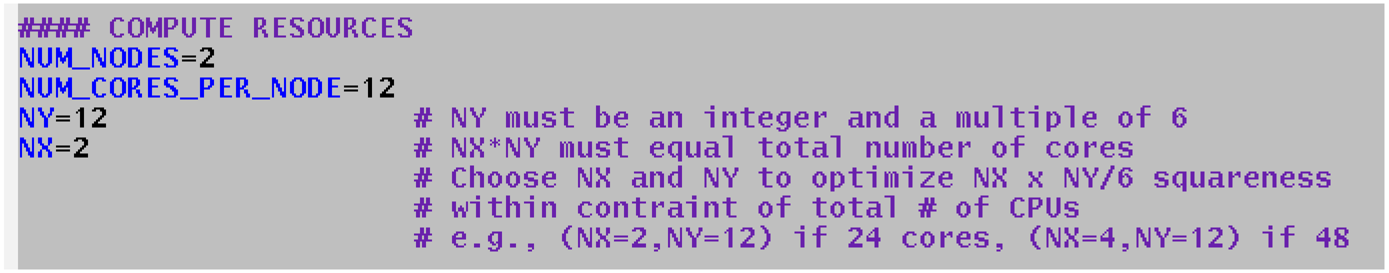
Example: GCHP with Multiple Nodes

- Run script:



```
#!/bin/bash

#SBATCH -n 24
#SBATCH -N 2
#SBATCH -t 0-12:00
#SBATCH -p regal
#SBATCH --mem-per-cpu=6000
#SBATCH --mail-type=ALL

# Load environment
BASHRC=GCHP.ifort15_mvapich2_odyssey.bashrc
echo "WARNING: You are using environment settings in $BASHRC"
source $BASHRC

# Overwrite config files with settings in runConfig.sh
./runConfig.sh > runConfig.log

# Run GCHP
log="GCHP.log"
time srun -n $SLURM_NTASKS -N $SLURM_NNODES --mpi=pmi2 ./geos >> $log
```

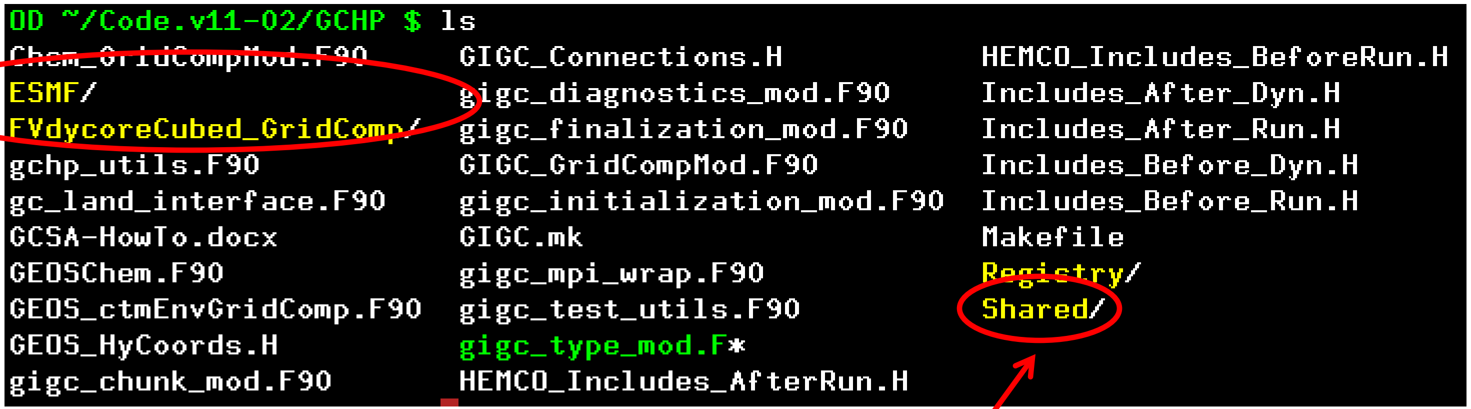
- runConfig.sh:



```
#### COMPUTE RESOURCES
NUM_NODES=2
NUM_CORES_PER_NODE=12
NY=12                 # NY must be an integer and a multiple of 6
NX=2                  # NX*NY must equal total number of cores
                      # Choose NX and NY to optimize NX x NY/6 squareness
                      # within contraint of total # of CPUs
                      # e.g., (NX=2,NY=12) if 24 cores, (NX=4,NY=12) if 48
```

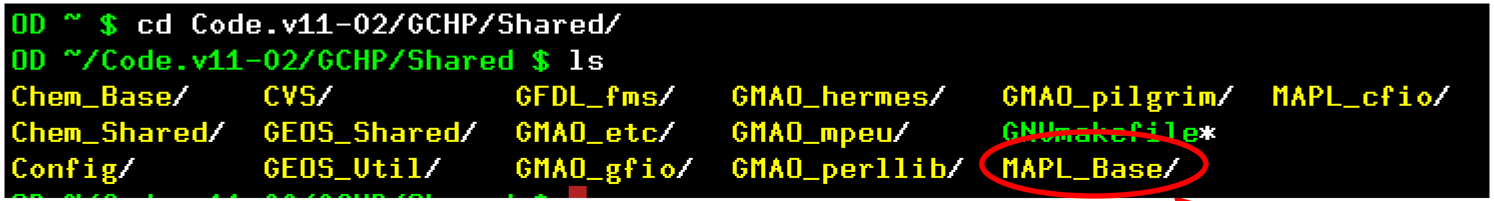
GCHP Source Code: ESMF, MAPL, FVdycore

ESMF and transport directories: these are compiled once and then you shouldn't need to touch them



```
OD ~/Code.v11-02/GCHP $ ls
Chem_GridCompMod.F90        GIGC_Connections.H          HEMCO_Includes_BeforeRun.H
ESMF/                       gigc_diagnostics_mod.F90    Includes_After_Dyn.H
FVdycoreCubed_GridComp/     gigc_finalization_mod.F90   Includes_After_Run.H
gchp_utils.F90              GIGC_GridCompMod.F90        Includes_Before_Dyn.H
gc_land_interface.F90       gigc_initialization_mod.F90 Includes_Before_Run.H
GCSA-HowTo.docx             GIGC.mk                     Makefile
GEOSChem.F90                gigc_mpi_wrap.F90           Registry/
GEOS_ctmEnvGridComp.F90     gigc_test_utils.F90         Shared/
GEOS_HyCoords.H             gigc_type_mod.F*
gigc_chunk_mod.F90          HEMCO_Includes_AfterRun.H
```

This is also compiled once. Most run directory issue errors will point you here.



```
OD ~ $ cd Code.v11-02/GCHP/Shared/
OD ~/Code.v11-02/GCHP/Shared $ ls
Chem_Base/    CVS/          GFDL_fms/    GMAO_hermes/   GMAO_pilgrim/  MAPL_cfio/
Chem_Shared/  GEOS_Shared/  GMAO_etc/    GMAO_mpeu/     GNUmakefile*
Config/       GEOS_Util/    GMAO_gfio/   GMAO_perllib/  MAPL_Base/
```

Especially here.

GCHP Source Code: MAPL

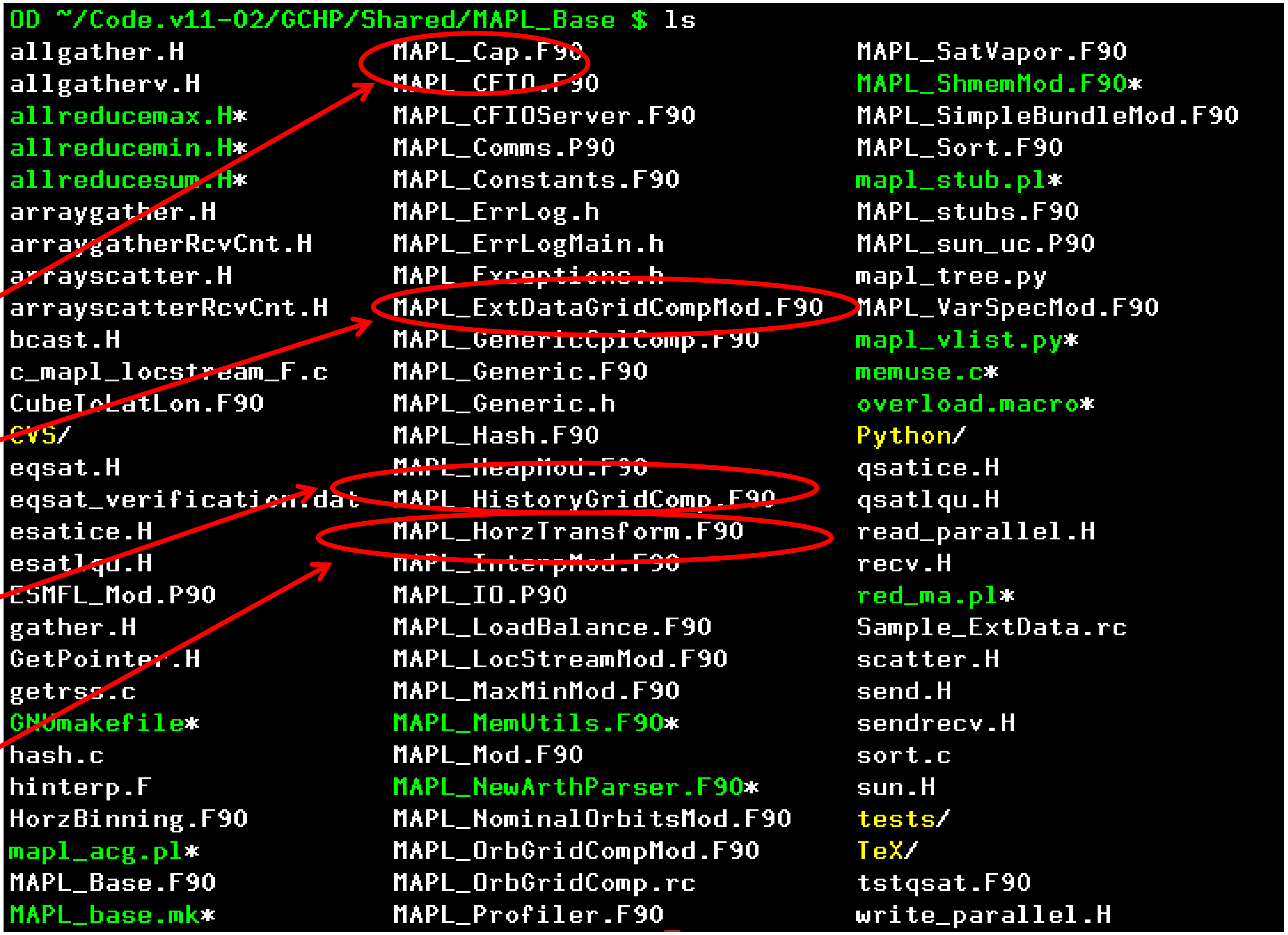
Error messages may lead you here…

Resource setup or time issues

Input data issues

Output data issues

Tile file issues (lat-lon <-> CS)

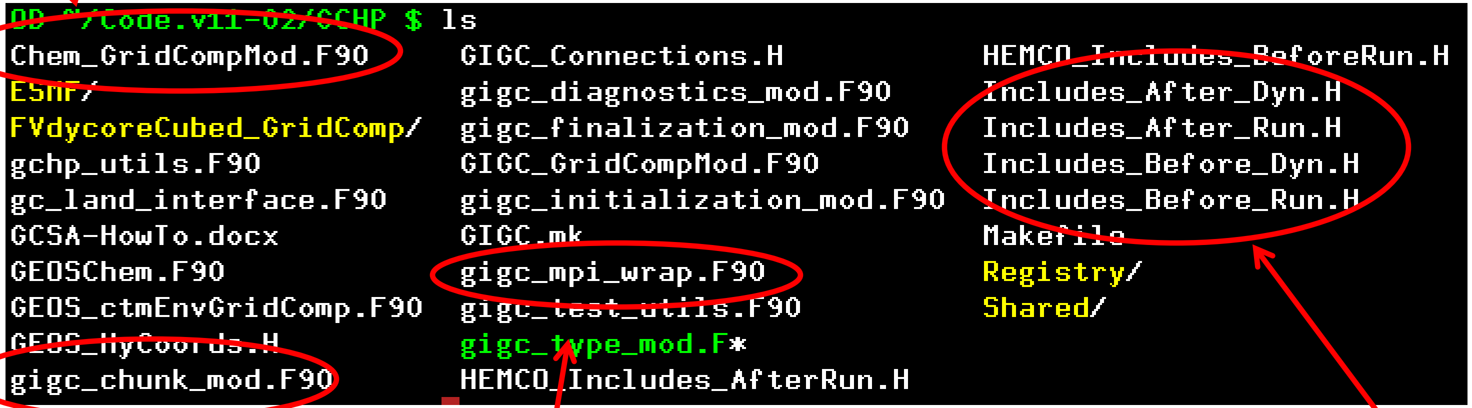


```
OD ~/Code.v11-02/GCHP/Shared/MAPL_Base $ ls
allgather.H                MAPL_Cap.F90                   MAPL_SatVapor.F90
allgatherv.H               MAPL_CFIO.F90                  MAPL_ShmemMod.F90*
allreducemax.H*            MAPL_CFIOServer.F90            MAPL_SimpleBundleMod.F90
allreducemin.H*            MAPL_Comms.P90                 MAPL_Sort.F90
allreducesum.H*            MAPL_Constants.F90             mapl_stub.pl*
arraygather.H              MAPL_ErrLog.h                  MAPL_stubs.F90
arraygatherRcvCnt.H        MAPL_ErrLogMain.h              MAPL_sun_uc.P90
arrayscatter.H             MAPL_Exceptions.h              mapl_tree.py
arrayscatterRcvCnt.H       MAPL_ExtDataGridCompMod.F90    MAPL_VarSpecMod.F90
bcast.H                    MAPL_GenericCplComp.F90        mapl_vlist.py*
c_mapl_locstream_F.c       MAPL_Generic.F90               memuse.c*
CubeToLatLon.F90           MAPL_Generic.h                 overload.macro*
CVS/                       MAPL_Hash.F90                  Python/
eqsat.H                    MAPL_HeapMod.F90               qsatice.H
eqsat_verification.dat     MAPL_HistoryGridComp.F90       qsatlqu.H
esatice.H                  MAPL_HorzTransform.F90         read_parallel.H
esatlqu.H                  MAPL_InterpMod.F90             recv.H
ESMFL_Mod.P90              MAPL_IO.P90                    red_ma.pl*
gather.H                   MAPL_LoadBalance.F90           Sample_ExtData.rc
GetPointer.H               MAPL_LocStreamMod.F90          scatter.H
getrss.c                   MAPL_MaxMinMod.F90             send.H
GNUmakefile*               MAPL_MemUtils.F90*             sendrecv.H
hash.c                     MAPL_Mod.F90                   sort.c
hinterp.F                  MAPL_NewArthParser.F90*        sun.H
HorzBinning.F90            MAPL_NominalOrbitsMod.F90      tests/
mapl_acg.pl*               MAPL_OrbGridCompMod.F90        TeX/
MAPL_Base.F90              MAPL_OrbGridComp.rc            tstqsat.F90
MAPL_base.mk*              MAPL_Profiler.F90              write_parallel.H
```

Review your run directory setup before trying to change MAPL code!

GCHP Source Code: High-level GCHP

GCHP equivalent of main.F in that it is where the actions are executed



```
OD ~/Code.v11-02/GCHP $ ls
Chem_GridCompMod.F90       GIGC_Connections.H          HEMCO_Includes_BeforeRun.H
ESMF/                      gigc_diagnostics_mod.F90    Includes_After_Dyn.H
FVdycoreCubed_GridComp/    gigc_finalization_mod.F90   Includes_After_Run.H
gchp_utils.F90             GIGC_GridCompMod.F90        Includes_Before_Dyn.H
gc_land_interface.F90      gigc_initialization_mod.F90 Includes_Before_Run.H
GCSA-HowTo.docx            GIGC.mk                     Makefile
GEOSChem.F90               gigc_mpi_wrap.F90           Registry/
GEOS_ctmEnvGridComp.F90    gigc_test_utils.F90         Shared/
GEOS_HyCoords.H            gigc_type_mod.F*
gigc_chunk_mod.F90         HEMCO_Includes_AfterRun.H
```

Module for init, run, and finalize methods called in Chem_GridCompMod.F90 (looks similar to main.F)

Where Input_Opt variables are broadcast as constants to all cores

Defines what import and internal states (full cubed_sphere arrays) are assigned to GEOS-Chem derived type objects to be processed per core (e.g. State_Met).

Resources

- GCHP Links:
  - Main Wiki Page
  - Online Tutorial
  - v11-02: new features, benchmarks, open and resolved issues
  - Working Group and Users
  - Timing Tests
- Other Useful Links:
  - Interactive construction of a cubed-sphere grid
  - FORTRAN tool for regridding between lat-lon and cubed-sphere
  - GMAO MAPL User's Guide (info may be outdated)
  - GEOS-5 wiki page for ExtData (info may be outdated)