<< Previous Chapter - Home - Next Chapter >>

5. CMAQ Installation and System Requirements

This section describes how to set up and install CMAQ on a Linux system. The installation instructions in this section guide the user through obtaining the CMAQ source code and installing it on your system. Brief instructions for running the CMAQ benchmark case and benchmarking the model are also addressed. Here, the term “benchmarking” refers to the process of verifying that a model has installed correctly on a new computer. CMAQ is distributed with a reference dataset that can be used to benchmark the CMAQ installation; in the distribution, output data from CMAQ are bundled with the input data (including emissions and meteorology) that can be used to reproduce the reference results.

After benchmarking has been successfully completed, the CMAQ system can be configured for other simulations. The same steps that are required to build the model for the benchmark case apply to building it for new simulations. Configuring CMAQ for new applications is covered in Chapter 10.

System Recommendations

All of the CMAQ programs are written in Fortran and are optimized for use on computers running a version of the Linux operating system (OS). Most personal computers (PCs) running a Linux OS are sufficiently powerful to handle basic CMAQ applications. However, to use CMAQ in a production environment where multiple iterations of the model will be executed for different spatial domains and/or emissions control strategies, either a cluster of multiprocessor PCs on a high-end network or an expandable rack-mounted Linux server is recommended.

CMAQ is distributed and supported for executing on Linux operating systems with the Intel Fortran, Portland Group Fortran (PGF), or Gnu Fortran compilers. CMAQ can be ported to most computers running Linux. Documented configurations include the SGI Altix, Red Hat Enterprise, Fedora, Ubuntu, Mandrake, MacOSX, and Suse operating systems. In addition to the Intel and PGF compilers, CMAQ has been built with Sun and Absoft compilers. Information about these ports and up-to-date hardware recommendations are available through the CMAS Center web site.

Hardware

The minimum hardware requirements for running the CMAQ benchmark case are:

- PC with a single 1.0 GHz processor with a Linux operating system OS
- 1 GB RAM
- 10 GB hard drive storage (Note: the benchmark simulation requires 3 GB of free storage capacity).

Recommendations on production-level hardware configurations for CMAQ are available on the CMAS Center Hardware Blog.

Software

CMAQ requires all of the programs listed in Table 5-1. This list includes the programs distributed with CMAQ. Note that CMAQv5.0 and greater requires I/O API version 3.1. Newer version of CMAQ will not compile with earlier versions of the I/O API library. Table 5-2 lists additional utility software that is not required for running CMAQ, but is useful for model diagnostics and evaluation.

Table 5-1. Software required for running CMAQ

Software	Description	Source
CMAQ Programs		
Bldmake	Executable builder for source code compilation	Contained in the standard CMA available at http://www.cmascent notes and documentation av http://www.cmaq-mode
JPROC	Photolysis rate preprocessor	"
ICON	Initial conditions preprocessor	"
BCON	Boundary conditions preprocessor	"
MCIP	Meteorology-Chemistry Interface Processor	"
CCTM	CMAQ Chemistry-Transport Model	"
	Chemical mechanism compiler for	

CHEMMECH	modifying or adding reactions to the CMAQ chemistry	"
CREATE_EBI	EBI Chemical solver source code generator	"
Compilers		
IFORT	Intel Fortran 90 compiler	http://www.intel.cor
PGF90	Portland Group Fortran 90 compiler	http://www.pgroup.cc
GFORT	Gnu Fortran compiler	http://gcc.gnu.org/fort
GCC	Gnu C compiler	http://gcc.gnu.org/
Code Libraries		
OpenMPI	Library for the message passing interface; used for multiprocessor CMAQ simulations	https://www.open-mpi
MPICH	Library for the message passing interface; used for multiprocessor CMAQ simulations	http://www.mcs.anl.gov/research/p

netCDF	Network Common Data Form library for controlling CMAQ file formats	http://www.unidata.ucar.edu/sof
I/O API	Input/Output Application Programming Interface for controlling internal and external communications	https://www.cmascenter.o
LAPACK	Linear algebra packages for use with the bidirectional mercury module	http://www.netlib.org/la
BLAS	Basic Linear Algebra Subprograms	http://netlib.org/blas
For versions prior to CMAQ-5.0.2		
CVS	Concurrent Versions System for managing the distributed archive of the CMAQ source code	http://ximbiot.com/cvs/cvshome/ package management s

Table 5-2. Optional support software for CMAQ

Software	Description	Source
Evaluation and visualization tools		
VERDI	Visualization Environment for Rich Data Interpretation for graphical analysis of netCDF gridded data	http://www.verdi-tool.org
PAVE	Package for Analysis and Visualization of Environmental data for graphical analysis of netCDF gridded data	http://www.cmascenter.org
IDV	Integrated Data Viewer for 3-D graphical analysis of netCDF gridded data	http://www.unidata.ucar.edu/softwa
I/O API Tools	Postprocessing tools for manipulating data in the I/O API/netCDF format	https://www.cmascenter.org/ioapi/
netCDF Tools	Postprocessing tools for manipulating	http://my.unidata.ucar.edu/content/

[illegible]	data in the netCDF format	[illegible]
Source code diagnostics		
GDB	Gnu Fortran debugger	https://www.sourceware.org/gdb/
PGDBG	Portland Group Fortran debugger	http://www.pgroup.com/
PGPROF	Portland Group Fortran code profiler	http://www.pgroup.com/
IDB	Intel Fortran debugger	https://software.intel.com/en-us/arti

Installing and Compiling CMAQ Source Code

Several steps are required to prepare your Linux system for compiling and running CMAQ. The general sequence for installing CMAQ, including the required support software and libraries is listed here.

1. Check for Fortran and C compilers on your Linux system. Install if they are not available.
2. Install Git (or CVS for older versions of CMAQ).
3. Download the source and install the I/O API and netCDF libraries. Follow the instructions in the documentation for each library on how to build them for your Linux system. Note: It is highly recommended that you use the same compiler for these libraries as you will use to build the CMAQ programs.

4. Install a Message Passing Interface (MPI) library on your Linux system.
5. Download the CMAQ source code and scripts from either the EPA GitHub Repository or the CMAS Center (http://www.cmascenter.org). After registering to download CMAQ on the CMAS Center Software Clearinghouse, users are redirected to a page that contains links to download Linux tar files of the CMAQ code, scripts, and benchmark data along with various documents describing the installation and execution processes. *Note that GitHub only provides access to source codes and scripts. Benchmark input and output data may only be downloaded from the CMAS Center.*

Distribution contents

The CMAQ distribution includes the following components:

- MODELS - CMAQ source code and libraries
- SCRIPTS - C-shell scripts to build and execute the CMAQ models
- INPUT DATA – datasets necessary to run the tutorial/benchmark case.
- REF DATA – reference output data to compare with datasets produced by the tutorial on a Linux workstation

Starting with CMAQv5.2, the structure of the CMAQ distribution includes:

- CCTM - Chemistry Transport Model source code and scripts
- PREP - Input pre-processor (e.g., ICON, BCON, MCIP) source code and scripts
- UTIL - Utility software (e.g., BLDMAKE, CHEMMECH, NML) source code and scripts
- POST - Post-processing and analysis software (e.g., COMBINE,

HR2DAY, BLDOVERLAY) source code and scripts

Notes on the CMAQ directory structure

The CMAQ installation includes a dataset for benchmarking the modeling system. Unpacking the various tar files of the distribution in a `CMAQ_HOME` (formerly M3HOME prior to CMAQv5.2) directory installs the CMAQ source code, scripts, and benchmark data files in a directory structure recognized by the default run and build scripts. The `CMAQ_HOME` directory is an arbitrary base location of the CMAQ installation on your Linux system for a specific application. It's up to the user to decide where to install CMAQ and to assign this location to the `CMAQ_HOME` environment variable in the CMAQ build scripts.

Under `CMAQ_HOME`, the `data` directory contains the input and output data for the model, and the `lib` directory contains the compiled binary library files required to build the CMAQ executables. The CMAQ scripts use the following environment variables to alias the locations of these directories:

`CMAQ_LIB = $CMAQ_HOME/lib` (M3LIB before CMAQv5.2)
`CMAQ_DATA = $CMAQ_HOME/data` (M3DATA before CMAQv5.2)

The CMAQ scripts require users to select only the location of the `CMAQ_HOME` directory; the other CMAQ directories are referenced relative to `CMAQ_HOME`. While this directory structure is convenient for the benchmark case and most CMAQ applications, other configurations are possible. Detailed instructions for installing and compiling CMAQ are contained in the next section.

Configuring your system for compiling CMAQ

Compiler flag consistency between the Fortran and C compilers used to build netCDF and I/O API is critical for building library files compatible with CMAQ. Table 5-3 lists the suggested compilation options for building netCDF and I/O API libraries that are compatible with CMAQ. Refer to the documentation for these libraries for additional information on installation and compiling.

Table 5-3. NetCDF and I/O API compilation options for CMAQ

Library Type	Intel Fortran	PGI Fortran	
netCDF	`CC = icc` `CPPFLAGS = -DNDEBUG –DpgiFortran` `CFLAGS = -g -O` `FC = ifort` `F77 = ifort` `FFLAGS = -O2 -mp -recursive` `CXX = icpc`	`CC = gcc` `CPPFLAGS = -DNDEBUG –DpgiFortran` `CFLAGS = -O` `FC = pgf90` `FFLAGS = -O -w` `CXX = g++`	`CC` `CPP` `DNDE` `DgF` `CFL` `FC` `FFL` `CXX`
I/O API 32-bit	BIN = Linux2_x86ifort	BIN = Linux2_x86pg_pgcc_nomp	N/A
I/O API 64-bit	BIN = Linux2_x86_64ifort	BIN = Linux2_x86_64pg_pgcc_nomp	BIN Linu

config.cmaq

Consistency of configuration variables is critical for building CMAQ itself, not just its libraries. Accordingly CMAQ includes the configuration script `config.cmaq` to help enforce consistent environment settings for CMAQ and its associated libraries Table 5-4 lists the `config.cmaq` variables defined for the build process and suggests values to which to set those variables.

Note that for multiprocessor applications it is recommended that the

Fortran MPI wrapper script `mpif90` be specified for the Fortran compiler (myFC). Using this script, instead of a direct call to the Fortran compiler, will ensure that the full suite of MPI components (libraries and include files) for the compiler are included in the parallel build.

Table 5-4. config.cmaq configuration variables

Variable Name	Suggested Value
`CMAQ_HOME`	The central CMAQ installation directory. For example, if you installed the CMAQ source code in the directory `/home/user/CMAQ` set CMAQ_HOME with `export CMAQ_HOME=/home/user/CMAQ` for bash or `setenv CMAQ_HOME /home/user/CMAQ` for csh; note that this variable is M3HOME prior to CMAQv5.2
`CMAQ_DATA`	Automatically set by config.cmaq; note that this variable is M3DATA prior to CMAQv5.2
`CMAQ_LIB`	Automatically set by config.cmaq; note that this variable is M3LIB prior to CMAQv5.2
`M3MODEL`	Automatically set by config.cmaq; deprecated in CMAQv5.2
`compiler`	Set the Fortran compiler type that you will use to compile CMAQ; choices are intel, pgi, or gcc
`IOAPI`	Location of the I/O API library installation on your Linux system
`NETCDF`	Location of the netCDF installation on your Linux system
`MPI`	Location of the Message Passing Interface installation on your Linux system
`netcdf_lib`	Name of the netCDF library on your system; set to "-lnetcdf" for versions < 4.2.0, "-lnetcdff -lnetcdf" for version 4.2.0 and later

`ioapi_lib`	Name of the I/O API libraryar on your system; set to "-lioapi"
`pnetcdf_lib`	Name of the parallel netCDF library on your system; set to "-lpnetcdf"
`mpi_lib`	Name of the MPI library on your system; set to "-lmpich" for MVAPICH, "-lmpi" for OpenMPI
`myFC`	Set to match the `FC` (Fortran compiler) you used to compile netCDF
`myCC`	Set to match the `CC` (C compiler) you used to compile netCDF
`myFFLAGS`	Fixed-format Fortran compiler optimization flags for your Linux system; suggested values for CMAQ are in the distributed script
`myCFLAGS`	C compiler optimization flags for your Linux system; suggested values for CMAQ are in the distributed script
`myFRFLAGS`	Free form-format Fortran compiler optimization flags for your Linux system; suggested values for CMAQ are in the distributed script
`MPI_INC`	Set to the path to your MPI library INCLUDE files, e.g. `$M3LIB/mpich/include`
`extra_lib`	Set to other libraries required for compiling on your Linux system; users will likely need to change this setting in the distributed script for portability to their system.
`EXEC_ID`	build tag, should be automatically set by config.cmaq

Installing CMAQ on your system

Use the following steps to install CMAQ (with examples using a C-shell environment, a Red Hat Linux system, and the Portland Group Fortran compiler) on a Linux system. CMAQ is not distributed with scripts for installing on Windows or MacOSX.

Obtain CMAQ source codes

CMAQ source code can be installed either using git or from tarballs downloaded from the CMAS Center. Both options are described here.

Git Installation

In the directory where you would like to install CMAQ, issue the following command to clone the official EPA GitHub repository for CMAQv5.2:

```
git clone -b 5.2 https://github.com/USEPA/CMAQ
```

Tarball Installation

Tarballs of the CMAQ source code are available from both the public GitHub repository and the CMAS Center Software Clearinghouse. In addition to the source code, reference input/output data for testing the installation of the software are available only from the CMAS Center; *data are not available through GitHub*. You must register/login to access the source codes and data from the CMAS Center.

In the directory where you would like to install CMAQ, unzip, and untar the model distribution file:

```
tar xvzf CMAQv5.2.tar.gz
```

Both of these installation options will produce the following subdirectories on your Linux system:

```
CMAQv5.2/CCTM
CMAQv5.2/PREP
CMAQv5.2/POST
CMAQv5.2/UTIL
```

Set up the central CMAQ configuration script

Edit the file `CMAQv5.2/config.cmaq` to configure the CMAQ installation for the local computing architecture and compilers.

The first step in editing `CMAQv5.2/config.cmaq` is to set the environment variable `CMAQ_HOME` to point the installation directory location of CMAQ on your Linux system.

Under the “architecture & compiler specific settings” section of the script set the compiler and libraries that you will use to build the CMAQ executables. There are three example compiler configurations built into the config.cmaq script: (1) Intel Fortran (intel), (2) Portland Group Fortran (pgi), and (3) Gnu Fortran (gcc). Set the script variable `compiler` for one of the supported compilers.

Set the names of the I/O API and netCDF libraries using the `ioapi_lib` and `netcdf_lib` script variables. You may also choose to set the pnetcdf_lib if you are using parallel netCDF. Note that for version 4.2 of the netCDF and later, you need to provide both the C and Fortran versions of the netCDF library, in this order: `-lnetcdff -lnetcdf`

Set the name of the MPI library using the `mpi` script variable. For MVAPICH use `-lmpich`; for openMPI use `-lmpi`. If you are using another MPI library, use the variable `mpi` to set the name of the library.

Save and exit from the config.cmaq file and use the source command to invoke the settings in the file:

```
source CMAQv5.2/config.cmaq
```

When you source the config.cmaq script the CMAQ directories, including all required libraries will be installed in the CMAQ working directory. If you encounter errors about libraries not being found, check the settings of the config.cmaq script variables IOAPI, NETCDF, or MPI to ensure that they are correctly point to the locations of these libraries on your Linux system.

Install the CMAQ input reference/benchmark data

After you have downloaded the CMAQ reference data from the CMAS Center Software Clearinghouse, navigate to the `$CMAQ_HOME` directory, unzip and untar the `CMAQv5.2.DATA.tar.gz` file:

```
cd $CMAQ_HOME
tar xvzf CMAQv5.2.DATA.tar.gz
```

This will produce the following subdirectories:

```
CMAQv5.2/data/
bcon/
bidi/
cctm/
crop/
dust/
emis/
icon/
lightning/
mcip/
ocean/
raw/
phot/
raw/
```

Install the CMAQ libraries

When you sourced the config.cmaq script above, the directory `CMAQ_LIB` was automatically created along with symbolic links to the I/O API, netCDF, and MPI libraries on your Linux system. The library and include files that were installed in this directory are based on the locations you specified in the config.cmaq script. The locations of the libraries in the CMAQ working directory will be based on the compiler that you selected for your CMAQ build. For example, if you are running on an x86_64 architecture with the Portland Group Fortran compiler, the config.cmaq script will set `CMAQ_LIB` to `$CMAQ_HOME/lib/x86_64/pgi`.

Compiling CMAQ

For all CMAQ programs (other than MCIP), the program Bldmake is used to compile the source code into executables. The first step in the compilation of CMAQ is to compile Bldmake. Bldmake will then be used to compile the CMAQ programs. *Note that the compiler paths and flags are all set in the config.cmaq script and then passed along to the build scripts*. None of the CMAQ build scripts contain compiler settings. Instead, the build scripts for each program reference the config.cmaq script using the Linux command “source”. See Table 5-4 for a description of the compilation flags in the config.cmaq script.

Before proceeding confirm that you have run the command `source config.cmaq`

All of the CMAQ programs other than CCTM are run in single-processor mode. CCTM may be run either in single-processor (serial) mode or in parallel on multiple processors. Program-specific compilation instructions are provided below. These compilation instructions are for building executables for simulating the test data sets distributed with CMAQ. Additional information about the configuration options for the various CMAQ programs is provided in Chapter 4 and Chapter 7.

Compile Bldmake

```
cd $CMAQ_HOME/UTIL/bldmake/src
make
```

Compile the CMAQ programs

Create the model executables for ICON, BCON, MCIP, and CCTM.

ICON and BCON can be configured for different chemical mechanisms and for different kinds of input data. The configuration options for ICON and BCON are discussed in detail in Chapter 7.

Use the following commands to compile ICON and BCON:

```
cd $CMAQ_HOME/PREP/icon
bldit.icon |& tee build.icon.log
```

```
cd $CMAQ_HOME/PREP/bcon
bldit.bcon |& tee build.bcon.log
```

Like the program Bldmake, MCIP is compiled using a Fortran Makefile.

Use the following commands to compile MCIP:

```
cd $CMAQ_HOME/PREP/mcip/src
source ../../../config.cmaq
make |& tee make.mcip.log
```

The CCTM has multiple configuration options that can be changed to optimize model performance for different applications. In addition to selecting the chemical mechanism to model gas-phase chemistry, the user can also select from several different science modules. The science configuration options for CCTM are discussed in detail in Chapter 4 and Chapter 7. The distribution CCTM build script is configured to create a multiprocessor executable for the installation test simulation. For multiprocessor applications, CMAQ uses the message passing interface (MPI) to manage communication between processors in a clustered multiprocessor computing environment. The location of the MPI include and library files on your Linux system are specified in the config.cmaq script.

For single-processor (serial) systems, configure the CCTM build script to create a single-processor executable by commenting out the line that activates the variable “ParOpt” of the CCTM build script. Use the following commands to compile CCTM:

```
cd $CMAQ_HOME/CCTM/scripts
bldit.cctm |& tee build.cctm.log
```

Although not used for the installation test simulation, the programs JPROC and PROCAN can also be compiled using Bldmake. The programs CHEMMECH and CALMAP are also not needed for the test simulation but can be compiled using Makefiles.

Running the CMAQ Installation Test Simulation

After successfully compiling the various CMAQ programs, use the distributed run scripts to generate the CCTM input files and then to run CCTM for the CMAQ benchmark case. CCTM must be run last in the simulation sequence; MCIP must be run first. Note however that CMAQ-ready meteorology data are distributed with the CMAQ test case, which means that MCIP does not actually need to be run to test the model installation. With the exception of MCIP, there are no dependencies among the other CMAQ programs, so they can be run in any order to create input data for CCTM.

To run the test simulation for the various CMAQ programs, change directories to the location of each program and execute the run script.

Run ICON to produce initial conditions:

```
cd $CMAQ_HOME/PREP/icon
./run.icon |& tee run.icon.log
```

Run BCON to produce boundary conditions:

```
cd $CMAQ_HOME/PREP/bcon
./run.bcon |& tee run.bcon.log
```

Check the ICON and BCON log file to ensure that the programs completed successfully.

The CCTM is configured by default to run in multiprocessor mode. This mode requires run time settings that specify the number of processors to allocate to the simulation and the location of the MPI initialization command (mpirun) on your system. Set the number of processors to use for the simulation by setting the NPROCS environment variable in the run.cctm script. You must also set the domain decomposition configuration by setting the variable NPCOL_NPROW. The number of processors (NPROCS) must be equal to the product of the two values selected for NPCOL_NPROW. For example, if you have a system with six processors available to run CMAQ, set NPROCS to 6 and NPCOL_NPROW equal to “3 2”.

For an MPI configuration with 6 processors,

```
setenv NPROCS 6
setenv NPCOL_NPROW “3 2”
```

Most clustered multiprocessor systems require a command to start the MPI run-time environment. The default CCTM run script uses the *mpirun* command. Consult your system administrator to find out how to invoke MPI when running multiprocessor applications. For single-processor computing, set NPROCS to 1 and NPCOL_NPROW to “1 1"

For single-processor computing,

```
setenv NPROCS 1
setenv NPCOL_NPROW to “1 1"
```

After configuring the MPI settings for your Linux system, run the CCTM:

```
cd $CMAQ_HOME/CCTM/scripts
./run.cctm |& tee cctm.log
```

Benchmarking

Benchmarking is the process of confirming that the model source code compiles and executes correctly on a new computer system. CMAQ should be benchmarked on a computing system before the model is used for research or regulatory applications on that system. The purpose of benchmarking is to ensure that there are no inconsistencies introduced into the model solution from local system parameters, such as compilers, processors, or operating systems. While differences are expected in the CMAQ results produced by different operating systems, hardware, and compilers, these differences should be small and within the numerical error of the model. Input and output reference data are packaged with CMAQ to use for benchmarking the model on new systems. After running the test case packaged with the source code, compare the results against the reference data provided in the CMAQ distribution.

CMAQ benchmark parameters

The CMAQ benchmark test case is a single day simulation for July 1, 2011 on a 72 column x 100 row x 35 layer 12-km resolution domain over California (CALNEX12 domain). The CCTM configuration parameters for the benchmark test case include the following:

- Multiprocessor simulation
- Horizontal advection: Yamo
- Vertical advection: WRF
- Horizontal diffusion: Multiscale
- Vertical diffusion: ACM2
- Deposition: M3Dry
- Chemistry solver: EBI
- Aerosol module: AERO6
- Cloud module: ACM
- Mechanism: cb05e51_ae6_aq
- Lightning NOx emissions calculated with hourly NLDN strike data
- Dynamic vertical diffusivity
- In-line deposition velocities
- Surface HONO interaction
- In-line biogenic emissions
- In-line plume rise
- In-line windblown dust emissions
- Bi-directional ammonia and mercury fluxes

The system configuration parameters used to generate the benchmark reference data include the following:

- Hardware: Dell C6100 server, 2.93 GHz Intel processor, 12M L3 cache (Model X5670), 48 GM memory
- Operating System: RHEL 5.6
- Compiler: Intel 15.0
- MPI: MVAPICH 2.1.7
- 12 processors

CMAQ Benchmark Results

After completing the CMAQ benchmark case, the CCTM output files can be compared with the reference datasets provided in the CMAQ distribution. The reference data for CMAQ are available from the CMAS Center Software Clearinghouse. The reference data may be compared to the results from your simulation using tile plots of differences, grid-cell statistics (minimum/maximum differences), and domain-wide statistics. See Chapter 12 for a list of analysis tools that are available for comparing two model simulations.

Domain-wide differences between the reference data and your simulation results for each model species and each simulation time step (hourly) of less than 1% indicate a successful benchmarking of the software on a Linux system. While larger differences may indicate a problem with the installation, they may also point to a configuration discrepancy in the benchmark simulation. Review the compiler optimization flags and the CCTM configuration that you used to resolve differences with the reference data.

Support for CMAQ is available from the CMAS Center (see Chapter 13).

<< Previous Chapter - Home - Next Chapter >>