Models-3/Community Multiscale Air Quality Model (CMAQ) User's Guide to Alternative Modules: Model of Aerosol Dynamics, Reaction, Ionization, and Dissolution (MADRID), Mercury (Hg), and Advanced Plume Treatment (APT)

Prepared by

Betty Pun, Prakash Karamchandani, Krish Vijayaraghavan,
Shu-Yun Chen and Christian Seigneur
Atmospheric & Environmental Research, Inc.
2682 Bishop Drive, Suite 120
San Ramon, CA 94583

Prepared for

Dr. Naresh Kumar, Dr. Eladio Knipping and Dr. Leonard Levin
EPRI
3412 Hillview Avenue
Palo Alto, CA 94304

Document Number CP194-05-1
January 2005

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

The U.S. Environmental Protection Agency (EPA) has developed and released the Models-3/Community Multiscale Air Quality Model (CMAQ) (Byun and Ching, 1999). Models-3/CMAQ is a three-dimensional grid-based air quality model that can be applied to simulate ozone (O_3) and other photochemical oxidants, particulate matter (PM), and the deposition of pollutants such as acids (i.e., sulfate, nitrate), nitrogen species, and toxic air pollutants. Under contract with EPRI and the California Air Resources Board (ARB), Atmospheric & Environmental Research, Inc. (AER), in collaboration with Professor John Seinfeld, California Institute of Technology (Caltech), incorporated the Model of Aerosol Dynamics, Reaction, Ionization, and Dissolution (MADRID) into Models-3/CMAQ (Pun et al., 2001; Zhang et al., 2002; Pun et al., 2003; Zhang et al., 2004). Since MADRID's original release in 2002, new versions of Models-3/CMAQ have been released by the Community Modeling and Analysis System (CMAS) Center. AER has upgraded MADRID to be compatible with the October 2004 release of Models-3/CMAQ (version 4.4) and incorporated state-of-the-art science modules for simulating mercury (Hg); these Hg modules are consistent with those used by AER in other continental and global three-dimensional grid-based models (Seigneur et al., 2004). AER has also upgraded CMAQ to incorporate the Advanced Plume Treatment (APT) for treating plumes within the grid system (Karamchandani et al., 2002). Models-3/CMAQ is supported by EPA and the user is referred to EPA documentation to install and set up Models-3/CMAQ. This user's guide focuses on the selection and application of the new modules incorporated into Models-3/CMAQ, including MADRID, Hg, and APT.

Two versions of MADRID are distributed with CMAQ-MADRID 2004. Both modules use a sectional representation of the particle size distribution. The formation of new particles is treated by calculating the relative rates of condensation and new particle formation given a minimum particle diameter threshold for the particle size distribution. Coagulation is neglected because much of the particle mass resides in the accumulation and coarse sections, where the evolution of the particle mass size distribution tends to be governed by gas-to-particle conversion. Gas-to-particle conversion is treated for both inorganic and organic species. ISORROPIA (Nenes et al, 1999) is used to simulate the

thermodynamic equilibrium of inorganic species. For organic species, MADRID 1 uses an empirical approach based on the results of smog chamber experiments conducted at Caltech. An abridged version of MADRID 1, referred to as MADRID 1a, has been developed (Pun et al., 2004). MADRID 1a streamlines the representation of secondary organic aerosols (SOA), using 20 SOA species instead of the original 38, to economize on computational resources. MADRID 2 uses a mechanistic approach with ten surrogate organic compounds. MADRID 1 is coupled to the Carbon-Bond Mechanism Version IV (CBM-IV) and the RADM2-CI4 (RADM2 chemistry with the 4-product isoprene chemistry; the original RADM2 mechanism and the version with only 1 isoprene product are not as frequently used and not supported in CMAQ-MADRID) for gas-phase chemistry. MADRID 1a is also coupled to the SAPRC99 mechanism in addition to the CBM-IV and RADM-CI4 mechanisms. MADRID 2 is coupled to the Caltech Atmospheric Chemistry Mechanism (CACM) (Griffin et al., 2002). The transfer of chemical species from the gas phase to the particles can be simulated using either a full equilibrium assumption or hybrid algorithms that approximate the kinetic process but are more computationally efficient. Growth of particles by condensation (and shrinkage by volatilization) is simulated with the moving-center method of Jacobson (Jacobson and Turco, 1995; Jacobson, 1997), which minimizes numerical diffusion and allows both particle mass and number concentrations to be tracked accurately. The PM number concentrations are explicitly treated by accounting for their changes due to various atmospheric processes (e.g., emission, diffusion, advection, deposition, and new particle formation). Table 1-1 summarizes the various components of MADRID. The formulation of MADRID is described in detail by Zhang et al. (2004) and an EPRI report titled “CMAQ-MADRID Technical Documentation” by Zhang et al. (2002).

The Carnegie-Mellon University (CMU) (Strader et al., 1998) cloud module, which was released with MADRID 2002, is not supported at this time.

CMAQ-MADRID has been upgraded with routines to simulate atmospheric mercury (Hg). The treatment of both gas-phase and aqueous phase Hg chemistry is included in this release. Details for the science of mercury treatment can be found in Seigneur et al. (2004).

APT is included as an alternative module to Plume-in-grid (PinG) models available within Models-3/CMAQ. APT is based on SCICHEM (Karamchandani et al., 2000; 2002), which is a puff model augmented with chemistry representative of in-plume conditions. Puff models represent plume transport and dispersion by means of a population of puffs that expand as atmospheric turbulence dilutes their material according to the local micrometeorological characteristics. Wind shear and puff overlap are treated and puff models provide more realistic representations of plume dynamics than can be obtained using traditional plume cross-section models.

Table 1-1. Processes simulated in MADRID. (Transport processes are those of CMAQ and are not listed here.)

Process	Module (Options are indicated with numbers)	Comments
Gas-phase chemistry	1. CBM-IV 2. RADM2-CI4 3. SAPRC99 4. CACM	CBM-IV, RADM2-CI4 and SAPRC99 modified to account for additional VOC for SOA formation and heterogeneous reactions
Gas-particle thermodynamic equilibrium for inorganic species	ISORROPIA version compatible with October 2004 CMAQ release (sulfate, nitrate, ammonium, sodium, chloride, water)	In MADRID 2, ISORROPIA modified to account for organic anions and particulate water associated with dissolved organic aerosols.
Gas-particle equilibrium for organic species	1. Empirical partition coefficients for absorption into an organic phase of 4 SOA from anthropogenic precursors and 34 SOA from biogenic precursors (MADRID 1); abridged version (MADRID 1a) simplifies biogenic SOA representation to 16 species. 2. Gas/particle thermodynamic equilibrium with 10 surrogate compounds by aqueous dissolution or absorption into an organic phase (MADRID 2)	Partition coefficients from Odum et al. (1997) and Griffin et al. (1999); compatible with CBM-IV and RADM2-CI4. See Pun et al. (2004) for description of the abridged SOA treatment in MADRID 1a See Pun et al. (2002) for module description; compatible with CACM.
Particle size distribution	Sectional with at least 2 size sections 1. 2-section representation (fine and coarse particles) 2. Multi-section representation	The Stokes diameter is used to define the size section boundaries; note that the $PM_{2.5}$ and PM_{10} definitions are based on the aerodynamic diameter.
Coagulation	None	Coagulation is negligible compared to other processes under most conditions
Nucleation	1. New particle formation theory of McMurry and Friedlander (1979) 2. None	Look-up table is the default option since the full model may be too computationally demanding for 3-D simulations. Nucleation can be neglected under conditions with high PM concentrations (e.g., polluted urban environment).
Condensational growth/shrinkage by volatilization	Diffusion-limited condensation/volatilization using the moving-center algorithm	Condensational growth algorithms do not apply to the two-section option Moving-center algorithm tracks both mass and number concentrations

Table 1-1. Processes simulated in MADRID (continued). (Transport processes are those of CMAQ and are not listed here.)

Process	Module (Options are indicated with numbers)	Comments
Gas/particle mass transfer for inorganic species	1. Hybrid algorithm – CIT 2. Hybrid algorithm – CMU 3. Full equilibrium algorithm	The CIT hybrid approach is the default approach. The CMU hybrid approach simulates explicit mass transfer for particles with diameter > 2.15 μm. It is computationally more expensive and can be numerically unstable under some conditions.
Gas/particle mass transfer for organic species	1. Hybrid algorithm – CIT 2. Hybrid algorithm – CMU 3. Full equilibrium algorithm	SOA formation occurs on fine particles when full equilibrium is selected; it occurs on fine particles with the CMU hybrid algorithm.
Cloud chemistry	1. None 2. RADM 3. CMU	CMU cloud module is not operational at this time
Heterogeneous chemistry	1. None 2. Four reactions with PM and one reaction with droplets	Jacob (2000). Three sets of reaction probabilities (i.e., base, lower bound, and upper bound) recommended by Jacob (2000) may be selected. The base values are the default values.
Dry deposition	Integrated flux approach	Venkatram and Pleim (1999)
Wet deposition	In-cloud (rainout) and below-cloud (washout) scavenging of gases and particles	Effective Henry's law constants used for gases

2. USING MODELS-3 WITH ENHANCED MODULES

2.1 Installation of New Modules and Compilation of Models-3/CMAQ

2.1.1 Chemistry transport model source codes

Models-3/CMAQ provides modularity based on the operator splitting approach and the source codes are arranged in the repository based on a directory structure that follows the science processes: (1) aero (PM thermodynamics and kinetics), (2) aero_depv (PM deposition), (3) chem (gas-phase chemistry and solver), (4) ping (plume in grid), (5) cloud (cloud dynamics and chemistry), (6) hadv (horizontal advection), (7) vadv (vertical advection), (8) hdiff (horizontal diffusion), and (9) vdiff (vertical diffusion). The routines for initialization and photolytic rate calculation are included in "init" and "phot", respectively. There are also other directories for utility routines, driver routines, and codes for process analysis.

CMAQ-MADRID, CMAQ-MADRID-Hg and CMAQ-APT are implemented within the October 2004 release (version 4.4) of Models-3/CMAQ. All source codes, including a complete set of the original CMAQ routines, are provided in the accompanying tape. The new CMAQ-MADRID source code files are incorporated into the Models-3/CMAQ repository under the various existing process directories, with one exception. A new directory called "common_MADRID" has been created to store an ordinary differential equation solver that is used in the simulations with particles or clouds, or both. During installation, the source code directory provided in the tape can replace the /*/models/CCTM/src/ directory on the local computer's Models-3 repository if the user is running the October 2004 version of Models-3/CMAQ. A comparison with the original code will show that several new subdirectories have been added, as shown in Figure 2-1 and listed in Table 2-1. While the pa_MADRID module is functional in MADRID simulations, process analysis results have not been analyzed for MADRID.

Several MADRID-relevant subroutines have also been added in the CMAQ's existing subdirectories such as "aero_noop" and "cloud_noop" to handle the simulations without particles or clouds. Three subroutines (aerorate_noop.F, movebin_noop.F and

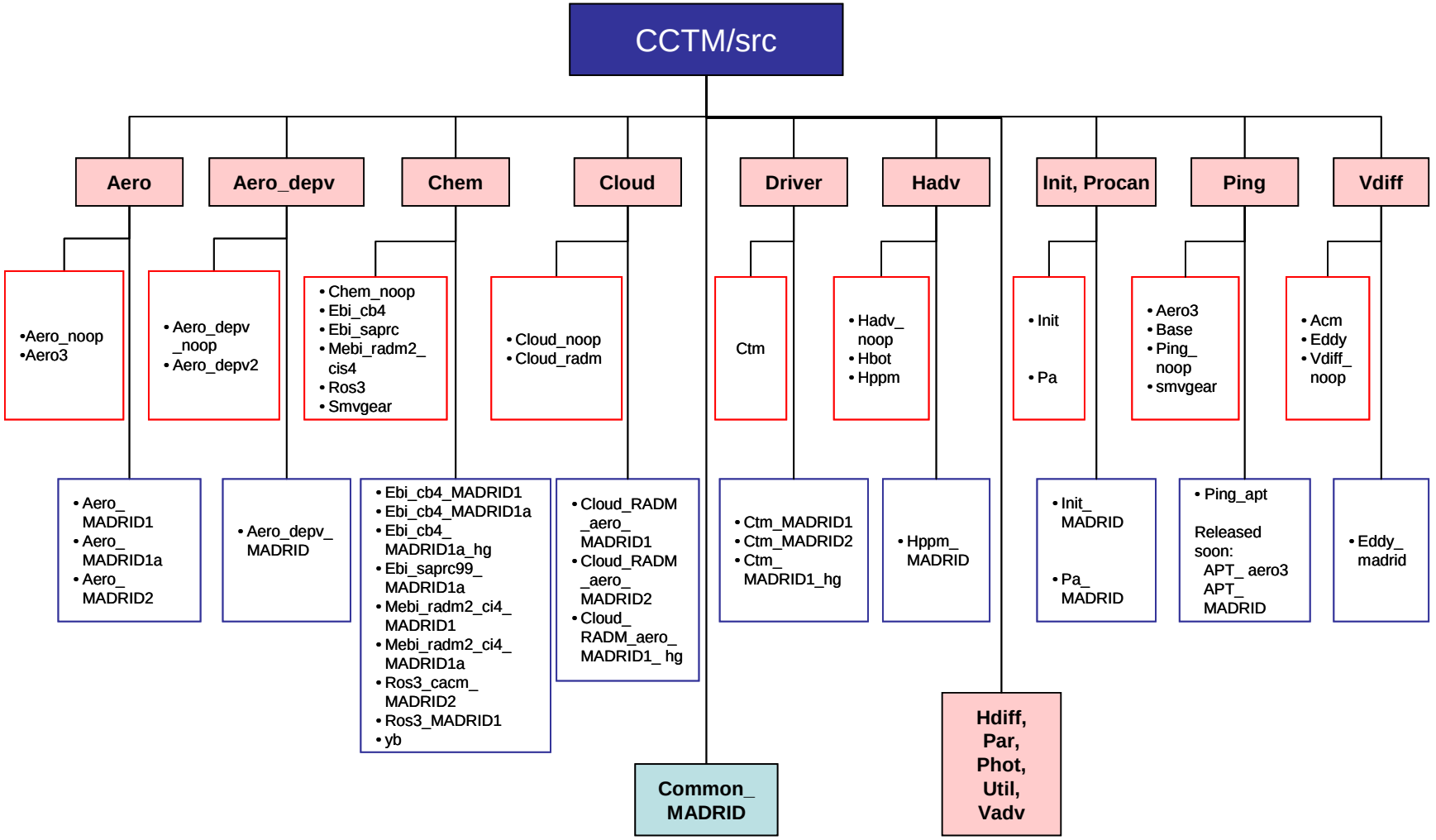

Figure 2-1. Directory structure of CMAQ-MADRID and CMAQ-APT. Blue boxes indicate MADRID and APT routines.

Table 2-1. Incorporation of new modules into the Models-3 repository.

Parent Directory	New Subdirectory	Comments
aero/	aero_MADRID1	Include files are mechanism- and size-dependent
	aero_MADRID1a	A version of MADRID 1 with fewer SOA species. Include files are mechanism- and size-dependent
	aero_MADRID2	Currently operational for 2 size sections
cloud/ *	cloud_radm_aero_MADRID1	RADM cloud chemistry for use with aero_MADRID1
	cloud_radm_aero_MADRID1_hg	Above + aqueous mercury reactions
	cloud_radm_aero_MADRID2	RADM cloud chemistry for use with aero_MADRID2
chem/	Ebi_cb4_MADRID1/ Ebi_cb4_MADRID1a	Euler backward integrative solver for CBM-IV gas-phase chemistry with SOA and heterogeneous reactions
	Ebi_cb4_MADRID1a_hg	Above + gas-phase reactions for mercury
	Ebi_saprc99_MADRID1a	Euler backward integrative solver for SAPRC99 gas-phase chemistry with SOA and heterogeneous reactions
	Mebi_radm2_ci4_Madrid1/ Mebi_radm2_ci4_Madrid1a	Modified Euler backward iterative solver for RADM-CI4 gas-phase chemistry with heterogeneous reactions
	Ros3_MADRID1	Rosenbrock solver for all gas-phase mechanisms for MADRID1 or MADRID1a with heterogeneous reactions
	Ros3_MADRID2	Rosenbrock s solver for CACM gas-phase chemistry with heterogeneous reactions
	yb	Young and Boris solver applicable with CMAQ or CMAQ-APT
hadv/	hppm_MADRID	used for all sectional simulations
init/	init_MADRID	used for all sectional simulations
vdiff/	eddy_MADRID	used for all sectional simulations
aero_depv/	aero_depv_MADRID	used for all sectional simulations
procan/	pa_MADRID	used for all sectional simulations
driver/	ctm_MADRID1	Special computing structure for driver.F and sciproc.F. New functionality to write restart files and cumulative deposition output. Used with aero_MADRID1
	ctm_MADRID1_hg	Above + directives for mercury simulations
	ctm_MADRID2	Special computing structure for driver.F and sciproc.F. New functionality to write restart files and cumulative deposition output; sciproc.F also modified to call PM module only once per hour. driver.F modified to provide additional outputs. Used with aero_MADRID2
ping/	ping_apt	Gas-phase at this time; PM APT and MADRID APT to become available in the near future

* CMU cloud modules are not operational at this time

movesetup_noop.F) are added in "aero_noop" to turn off particle growth and heterogeneous reactions on the surface of particles when particles are not simulated. Similarly, two subroutines (cldproc_lwc_noop.F and cldrate_noop.F) are added in "cloud_noop" to turn off heterogeneous reactions in cloud droplets. These "aero_noop" and "cloud_noop" modules work with both the original modal CMAQ and CMAQ-MADRID.

The new modules in CMAQ-MADRID comply with Models-3's coding standard. Therefore, these modules should be fairly transferable to future versions of Models-3/CMAQ, unless there are significant changes in Models-3/CMAQ's data structures and subroutine arguments. In that case, modifications may be necessary based on the design changes in Models-3/CMAQ.

The file "modules" in CCTM/CVSROOT/ directory is modified to reflect the addition of new modules and/or new classes in the Models-3/CMAQ repository. Since "common_MADRID" is a new class of science directory, it should be added in the repository and the new ODE solver module "ae_aq_solver_MADRID" should be added under the new CLASS called "common_MADRID". This is done by adding the following lines in the file "modules":

```
   common_MADRID_C  -s CLASS_common_MADRID  src/common_MADRID
ae_aq_solver_MADRID -s MODULE_common_MADRID src/common_MADRID/ae_aq_solver_MADRID
```

Other new "MODULE" entries are added under each "CLASS." A simple example is shown below:

```
vdiff           -s CLASS_vdiff       src/vdiff
eddy_MADRID     -s MODULE_vdiff      src/vdiff/eddy_MADRID
```

A new "modules" file is provided with the codes.

2.1.2 Include files

Include files specific to the processes are required to build the executable code. The subdirectories of include files are located in the Models-3/CMAQ include/release/ directory. Each combination of gas-phase chemistry, aerosol chemistry and dynamics, aqueous-phase chemistry modules, and particle size resolution requires an appropriate directory of include files. As a result of the incorporation of one new and three modified gas-phase mechanisms (CACM, CBM-IV, RADM2-CI4 and SAPRC99) with the MADRID aerosol modules, there are new directories of include files corresponding to the available mechanism/cloud module/PM module/particle size combinations in the Models-3/CMAQ include/release/ directory. Each of the directories of include files is labeled by the gas-phase chemistry, aqueous-phase chemistry, aerosol module, followed by the number of sections for the application. The appropriate include directory to be used with specific applications will be discussed in greater detail in Section 2.2.

In each include file directory, there are two files to describe the gas-phase mechanism and the heterogeneous reactions (i.e., RXCM.EXT and RXDT.EXT). The file called “mech.def” in each directory is the input file for the Models-3 chemistry reader used to generate RXCM.EXT and RXDT.EXT, and provides a description of all the gas-phase and heterogeneous reactions, including those responsible for the formation of SOA.

There are four module-specific include files (i.e., PARAMETER.EXT, ARRAYMAP.EXT, AQ_PARAMS.EXT and HETERPAR.EXT) to specify constants, variables, arrays, species mapping, and number of size sections that are used for specific mechanism/particle size combinations. The file “PARAMETER.EXT” is needed for all MADRID simulations regardless of the mechanism/particle size combination. The file “ARRAYMAP.EXT “ is needed for simulations with particles (i.e. gas+aerosol and gas+aerosol+cloud). The file “AQ_PARAMS.EXT” is needed for simulations with clouds (i.e., gas+cloud and gas+aerosol+cloud). The file “HETERPAR.EXT” is needed for simulations with particles, or clouds, or both. Options within CMAQ-MADRID are selected using flags in PARAMETER.EXT and HETERPAR.EXT as follows:

PARAMETER.EXT:

- IAERORATE (1 (default) to turn on heterogeneous reactions on aerosol surfaces, 0 to turn off. 0 is always used in simulations without aerosols. IAERORATE = 1 for mercury simulations)
- ICLDRATE (1 (default) to turn on cloud heterogeneous reactions, 0 to turn off. 0 is always used in simulations without clouds. ICLDRATE = 1 for mercury simulations)
- IAPART (1 (default) to select CIT hybrid algorithm for gas/particle mass transfer; 2 to select full equilibrium approach, 3 to select CMU hybrid algorithm)
- IAPARTORG (this option is effective for IAPART = 1 or 2; 0 (default) to distribute SOA in both sections in a 2-section simulation; 1 to condense SOA in fine section only)
- INUCL (1 (default) to turn on nucleation; 0 to turn off)

HETERPAR.EXT:

- NGAMMA (1 (default) to select the base values of reaction probability for heterogeneous reactions on the surface of particles; 2 to select the lower bound values, 3 to select the upper bound values)

In addition, the include directory contains 'include' files for gas-phase species (GC_*.EXT), aerosol species (AE_*.EXT), and non-reactive species (NR_*.EXT) that specify species names, process participation, output, and correspondence between species in different phases. A list of files for gas-phase species is provided in Table 2-2. Corresponding files for aerosol, non-reactive, and tracer species have similar structures and functions. For example, the list of species for each PM module can be found in AE_SPC.EXT. Note that only one file is used for the correspondence of the aerosol species: AE_A2AQ.EXT.

The include files GC_*.EXT, AE_*.EXT and NR_*.EXT are specific to each application, because (1) MADRID 1, MADRID 2, and the original Models-3 modal PM module treat different organic aerosol species; (2) MADRID treats the additional inorganic species of sodium and chloride in the particulate phase; (3) the CMU and

RADM aqueous-phase chemistry modules treat different sets of aqueous-phase equilibria and kinetic reactions among different gaseous and aqueous-phase species; and (4) different applications require the simulation of different gas-phase species. Hydrochloric acid (HCl) is included as an inert gas-phase species in all MADRID simulations. For MADRID 2 only, a non-reactive organic compound BUTAN is included. For mercury simulations, several mercury species are represented; aqueous-phase chemistry and heterogeneous reactions must be included for mercury chemistry, which require the representation of chlorine gas (Cl_2) as well as HCl. In MADRID, the mapping of gas-phase and non-reactive species to aerosol species are defined in ARRAYMAP.EXT. As a result, the include files G2AE.EXT and N2AE.EXT are not used in MADRID simulations.

Table 2-2. Gas-phase species extension files (GC_*.EXT).

Filename (* =)	Description	Comments
SPC	Species name and general characteristics	Corresponds to the gas-phase mechanism
ADV	Species that undergo advection	
DEPV	Species that undergo dry deposition and deposition velocity data	
DIFF	Species that undergo diffusion	
EMIS	Species with emission and emission factor data	
ICBC	Parameters for initial and boundary conditions	
SCAV	Species that are scavenged by rain (wet deposition)	needed for simulations with clouds
CONC	species with concentration output	
DDEP	species with dry deposition output	
WDEP	Species with wet deposition output	needed for simulations with clouds
G2AE	Correspondence between gas-phase and aerosol species	G2AE.EXT not used with CMAQ-MADRID
G2AQ	Correspondence between gas-phase and aqueous-phase species	needed for simulations with clouds

2.2 Compilation of CMAQ-MADRID

The modules within CMAQ-MADRID can be selected in the same manner as the default options provided by EPA using a script for compilation. In the code-building script, the paths of the include directories are specified, along with the selected modules. When the build-script is executed, the source codes are checked out from the repository and copied into a working source code directory and compiled there to build the executable. The appropriate libraries are linked. Include files are not copied into the directory where the code is built, and they remain in the include directory of the repository.

The SOA module in aero_MADRID2 consists of codes written in C and Fortran languages. As is the case for all other modules, the source codes are checked out of the repository, copied into the working directory, and compiled in the working directory. In the case of MADRID 2, two compilers will be invoked to generate the relevant object files.

The APT module is linked to the CMAQ executable as a library of routines, similar to other Models-3/CMAQ libraries. The ping/APT module consists of a driver routine (ping_apt.F), an include file (FILES_APT.diag.EXT) and a subdirectory 'scichem' that contains the plume model source codes. SCICHEM uses the Young and Boris solver for gas-phase chemistry within the plume; therefore, the same solver should be used to solve the chemistry within the background grid cell to ensure consistency between the plume and background chemistry. Routines specific to the Young and Boris chemistry solver are contained in the 'yb' directory under the 'chem' module in the CMAQ repository. The structure of the 'scichem' subdirectory is consistent with that of the stand-alone puff chemistry model SCICHEM, so that any developments made to SCICHEM can be easily incorporated into APT. When CMAQ is built with the APT option, the entire directory structure is copied into a subdirectory called "scichem" within the working source directory. The build script first calls a makefile to build the odepack library (libodepack.a) and then another (makefile.scilib: see Appendix B, Section 4), to build the scichem library (libsci.a) from the source codes. The APT library is

automatically linked to the other object files for CMAQ, in a manner similar to other libraries such as NetCDF and IOAPI.

Sample scripts for building CMAQ-MADRID are shown in the appendices for the examples to be discussed in Section 4. In the script shown in Appendix A-1, aero_MADRID1 is selected for PM with 8 size sections and cloud_RADM is selected for cloud processes. In the script shown in Appendix A-2, the aero_MADRID2 module (2 sections) is selected in a model that does not contain cloud processes. The build script for CMAQ-MADRID for mercury applications is shown in Appendix A-3. Appendix A-4 shows the compilation script for CMAQ with APT. Note the compilation and linking of the APT library.

Figure 2-2 illustrates the associations among different gas-phase mechanisms, PM modules, and cloud modules. The user should not combine codes from the original modal version of CMAQ and the sectional CMAQ-MADRID in one application.

The thermodynamics of inorganic species and aerosol dynamics are treated similarly in the two MADRID modules. The organic condensable compounds are specifically produced in different mechanisms. There are, therefore, compatibility requirements with the two SOA modules that treat different organic condensable species. The Odum/Griffin SOA module in aero_MADRID1 and aero_MADRID1a is used with modified versions of CBM-IV, RADM2-CI4 and SAPRC99. On the other hand, the AEC module in aero_MADRID2 is compatible with CACM. Therefore, the new PM modules are designed to be used in combination with specific gas-phase chemical mechanisms.

A modified version of the original Models-3/CMAQ cloud module (RADM) was developed to function with MADRID. The RADM cloud module interfaces with MADRID and the corresponding gas-phase mechanisms (i.e., CBM-IV, RADM2_CI4 or SPARC99 for MADRID 1 and CACM for MADRID 2). Another cloud module, the CMU cloud model, was incorporated into CMAQ-MADRID, but is not operational at this point.

The combinations of gas-phase mechanisms, aerosol modules, and cloud modules that are supported with the current release of MADRID are listed in Table 2-3. When developing a new application, the user is encouraged to compare the selected modules in the build script against those listed in Table 2-3. For each combination of gas, PM, and

aqueous modules, the appropriate include directory (also listed in Table 2-3) needs to be specified in the build script. Example build scripts (LINUX platform) are provided in the tape.

The mercury option of CMAQ-MADRID should include aqueous-phase chemistry and heterogeneous chemistry to ensure that all reduction-oxidation processes are correctly represented.

The module selection for APT applications are listed in Table 2-3. APT works only with the CBM-IV mechanism. At this time, the application of APT is limited to gas-phase simulations. APT for PM applications will be released in the near future.

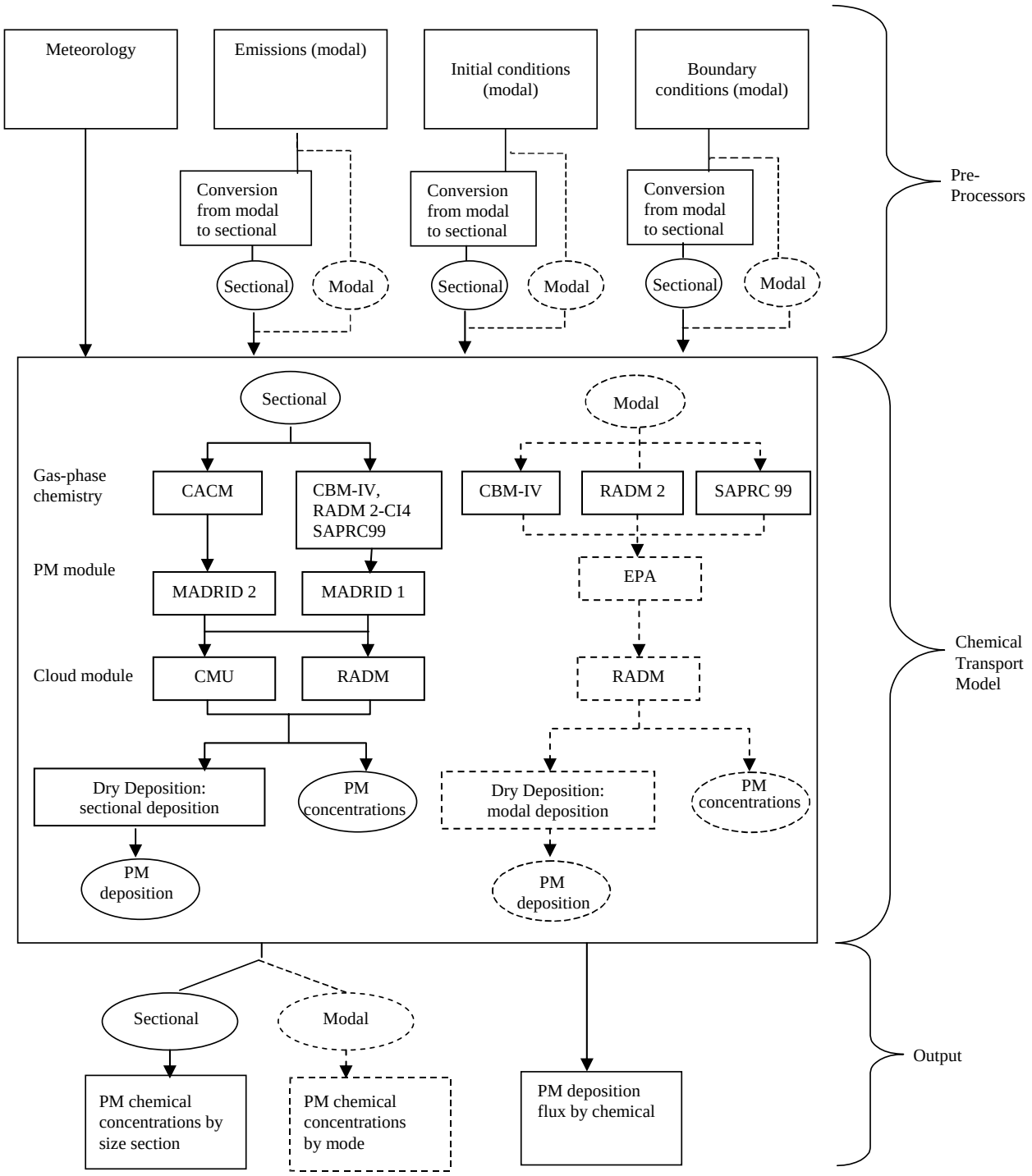

Figure 2-2. Flow chart indicating the associations of various gas-phase mechanisms, PM modules, and cloud modules. RADM2-CI4 is not represented explicitly in this figure, although it can be applied with both sectional and modal models.

Table 2-3a. Configurations of modules and mechanism available in CMAQ-MADRID: CBM-IV (2 solvers), MADRID 1 (comprehensive or abridged), RADM (cloud option) and mercury option.

Configuration[(1)]	CBM-IV[(2)] + MADRID 1 [(3)] (no cloud)	CBM-IV[(2)] + RADM + MADRID 1[(3)]	CBM-IV[(2)] + Hg + RADM + MADRID 1a
Mechanism/ Include directory	cb4_aeMADRID1_2sec cb4_aeMADRID1_8sec [(3)]	cb4_aqRADM_aeMADRID1_2sec cb4_aqRADM_aeMADRID1_8sec [(3)]	cb4_aqRADM_hg_aeMADRID1a_2sec
Module			
ModDriver	ctm_MADRID1	ctm_MADRID1	ctm_MADRID1_hg
ModPar	par_noop	par_noop	par_noop
ModInit	init_MADRID	init_MADRID	init_MADRID
ModAdjc	denrate	denrate	denrate
ModCpl	gencoor	gencoor	gencoor
ModHadv	hppm_MADRID	hppm_MADRID	hppm_MADRID
ModVadv	vppm	vppm	vppm
ModHdiff	multiscale	multiscale	multiscale
ModVdiff	eddy_MADRID	eddy_MADRID	eddy_MADRID
ModPhot	phot	phot	phot
ModPing	ping_noop	ping_noop	ping_noop
ModChem	ros3_MADARID1 or ebi_cb4_MADRID1[(3)]	ros3_MADARID1 or ebi_cb4_MADRID1[(3)]	ebi_cb4_MADRID1a_hg
ModAero	aero_MADRID1[(3)]	aero_MADRID1[(3)]	aero_MADRID1a
ModAdepv	aero_depv_MADRID	aero_depv_MADRID	aero_depv_MADRID
ModCloud	cloud_noop	cloud_RADM_aero_MADRID1	cloud_RADM_aero_MADRID1_hg
ModPa	pa_MADRID	pa_MADRID	pa_MADRID
ModUtil	util	util	util
Modaeaqsolver	ae_aq_solver_MADRID	ae_aq_solver_MADRID	ae_aq_solver_MADRID

(1) The combination of gas-phase mechanism, cloud module, and aerosol module. CBM-IV, MADRID 1, and RADM denote the CBM-IV gas-phase chemical mechanism, the MADRID 1 aerosol module, the RADM aqueous-phase chemical mechanism, respectively.

(2) The CBM-IV gas-phase mechanism includes heterogeneous reactions on surface of particles and droplets; these reactions are not simulated when aqueous-phase chemistry and/or aerosol processes are not selected. They can also be neglected by the user. Two flags (IAERORATE and ICLDRATE) are included in PARAMETER.EXT to provide options to turn on/off these heterogeneous reactions.

(3) The abridged version of MADRID 1, with 20 instead of 38 SOA species, can also be run with similar combinations of modules as MADRID 1, except that include files and ebi chemistry solvers are specific to MADRID 1a. If the Rosenbrock solver is selected, MADRID 1 and MADRID 1a use the same chem module

Table 2-3b. Configurations of modules and mechanism available in CMAQ-MADRID: RADM2-CIS4 (2 solvers), MADRID 1 (unabridged or abridged), and RADM (cloud option).

Configuration [1]	RADM2-CI4 [2]+ MADRID 1[3] (no cloud)	RADM2_CI4 [2]+ RADM + MADRID 1[3]
Mechanism/ Include directory	radm2_cs4_aeMADRID1_2sec radm2_ci4_aeMADRID1_8sec [3]	radm2_ci4_aqRADM_aeMADRID1_2sec radm2_ci4_aqRADM_aeMADRID1_8sec[3]
Module		
ModDriver	ctm_MADRID1	ctm_MADRID1
ModPar	par_noop	par_noop
ModInit	init_MADRID	init_MADRID
ModAdjc	denrate	denrate
ModCpl	gencoor	gencoor
ModHadv	hppm_MADRID	hppm_MADRID
ModVadv	vppm	vppm
ModHdiff	multiscale	multiscale
ModVdiff	eddy_MADRID	eddy_MADRID
ModPhot	Phot	Phot
ModPing	ping_noop	ping_noop
ModChem	ros3_MADRID1 or mebi_radm2_ci4_MADRID1[3]	ros3_MADRID1 or mebi_radm2_ci4_MADRID1[3]
ModAero	aero_MADRID1[3]	aero_MADRID1[3]
ModAdepv	aero_depv_MADRID	aero_depv_MADRID
ModCloud	cloud_noop	cloud_RADM_aero_MADRID1
ModPa	pa_MADRID	pa_MADRID
ModUtil	util	util
Modaeaqsolver	ae_aq_solver_MADRID	ae_aq_solver_MADRID

(1) The combination of gas-phase mechanism, cloud module, and aerosol module. RADM2-CI4, MADRID 1, and RADM denote the RADM2-CI4 gas-phase chemical mechanism, the MADRID 1 aerosol module, the RADM aqueous-phase chemical mechanism, respectively.

(2) The RADM2-CI4 gas-phase mechanisms include heterogeneous reactions on surface of particles and droplets; these reactions are not simulated when aqueous-phase chemistry and/or aerosol processes are not selected. They can also be neglected by the user. Two flags (IAERORATE and ICLDRATE) are included in PARAMETER.EXT to provide options to turn on/off these heterogeneous reactions.

(3) The abridged version of MADRID 1, with 20 instead of 38 SOA species, can also be run with similar combinations of modules as MADRID 1, except include files and ebi chemistry solvers are specific to MADRID 1a. If the Rosenbrock solver is selected, MADRID 1 and MADRID 1a use the same chem module

Table 2-3c. Configurations of modules and mechanism available in CMAQ-MADRID: SAPRC.

Configuration [(1)]	Saprc99[(2)] + MADRID 1a (no cloud)	Saprc99[(2)] + RADM + MADRID 1a
Mechanism/ Include directory	Saprc99_aeMADRID1a_2sec	Saprc99_aqRADM_aeMADRID1a_2sec
Module		
ModDriver	ctm_MADRID1a	ctm_MADRID1a
ModPar	par_noop	par_noop
ModInit	init_MADRID	init_MADRID
ModAdjc	denrate	denrate
ModCpl	gencoor	gencoor
ModHadv	hppm_MADRID	hppm_MADRID
ModVadv	vppm	vppm
ModHdiff	multiscale	multiscale
ModVdiff	eddy_MADRID	eddy_MADRID
ModPhot	phot	phot
ModPing	ping_noop	ping_noop
ModChem	ebi_saprc99_MADRID1a or ros3_MADRID1	ebi_saprc99_MADRID1a or ros3_MADRID1
ModAero	aero_MADRID1a	aero_MADRID1a
ModAdepv	aero_depv_MADRID	aero_depv_MADRID
ModCloud	cloud_noop	cloud_RADM_aero_MADRID1
ModPa	pa_MADRID	pa_MADRID
ModUtil	util	util
Modaeaqsolver	ae_aq_solver_MADRID	ae_aq_solver_MADRID

(1) The combination of gas-phase mechanism, cloud module, and aerosol module. SARPC, MADRID 1a and RADM denote the SAPRC gas-phase chemical mechanism, the MADRID 1a aerosol module and the RADM aqueous-phase chemical mechanism, respectively.

(2) The SAPRC gas-phase mechanisms include heterogeneous reactions on surface of particles and droplets; these reactions are not simulated when aqueous-phase chemistry and/or aerosol processes are not selected. They can also be neglected by the user. Two flags (IAERORATE and ICLDRATE) are included in PARAMETER.EXT to provide options to turn on/off these heterogeneous reactions.

Table 2-3d. Configurations of modules and mechanism available in CMAQ-MADRID: CACM.

Configuration [1]	CACM [2] (gas-phase only)	CACM [2]+ MADRID 2 (no cloud)	CACM[2] + RADM (no aerosol)	CACM + RADM + MADRID 2
Mechanism/ Include directory	CACM	CACM_aeMADRID2_2sec	CACM_aqCMU	CACM_aqRADM_aeMADRID2_2sec
Module				
ModDriver	ctm_MADRID2	ctm_MADRID2	ctm_MADRID2	ctm_MADRID2
ModPar	par_noop	par_noop	par_noop	par_noop
ModInit	init_MADRID	init_MADRID	init_MADRID	init_MADRID
ModAdjc	denrate	denrate	denrate	denrate
ModCpl	gencoor	gencoor	gencoor	gencoor
ModHadv	hppm_MADRID	hppm_MADRID	hppm_MADRID	hppm_MADRID
ModVadv	vppm	vppm	vppm	vppm
ModHdiff	multiscale	multiscale	multiscale	multiscale
ModVdiff	eddy_MADRID	eddy_MADRID	eddy_MADRID	eddy_MADRID
ModPhot	phot	phot	phot	phot
ModPing	ping_noop	ping_noop	ping_noop	ping_noop
ModChem	ros3_cacm_MADRID2	ros3_cacm_MADRID2	ros3_cacm_MADRID2	ros3_cacm_MADRID2
ModAero	aero_noop	aero_MADRID2	aero_noop	aero_MADRID2
ModAdepv	aero_depv_noop	aero_depv_MADRID	aero_depv_noop	aero_depv_MADRID
ModCloud	cloud_noop	cloud_noop	cloud_RADM_aero_MADRID2	cloud_RADM_aero_MADRID2
ModPa	pa_MADRID	pa_MADRID	pa_MADRID	pa_MADRID
ModUtil	util	util	util	Util
Modaeaqsolver	--	ae_aq_solver_MADRID	ae_aq_solver_MADRID	ae_aq_solver_MADRID

(1) The combination of gas-phase mechanism, cloud module, and aerosol module. CACM, MADRID 2 and RADM denote the CACM gas-phase chemical mechanism, the MADRID 2 aerosol module and the RADM aqueous-phase chemical mechanism, respectively.

(2) The CACM gas-phase mechanism includes heterogeneous reactions on surface of particles and droplets; these reactions are not simulated when aqueous-phase chemistry and/or aerosol processes are not selected. They can also be neglected by the user. Two flags (IAERORATE and ICLDRATE) are included in PARAMETER.EXT to provide options to turn on/off these heterogeneous reactions.

Table 2-3e. Configurations of modules and mechanism available in CMAQ-MADRID: APT.

Configuration*	CBM-IV + APT (gas-phase only)
Mechanism/ Include directory	cb4
Module	
ModDriver	
ModPar	par_noop
ModInit	init
ModAdjc	denrate
ModCpl	gencoor
ModHadv	hppm
ModVadv	vppm
ModHdiff	multiscale
ModVdiff	eddy
ModPhot	phot
ModPing	ping_apt
ModChem	yb
ModAero	aero_noop
ModAdepv	aero_depv_noop
ModCloud	cloud_noop
ModPa	pa
ModUtil	util
Modaeaqsolver	--

2.3 Input Files and Preprocessors

2.3.1 Sectional PM inputs

The Models-3/CMAQ inputs that need to be modified as a result of incorporating the new PM and cloud modules include emissions and initial/boundary conditions. PM inputs need to be commensurate with the sectional formulation. The modal formulation of Models-3/CMAQ October 2004 version accepts two formats for PM emissions. If the emissions of $PM_{2.5}$ and PM_{10} are supplied, PM speciation is performed internally based on some default profile for elemental carbon (EC) and organic compounds. There are different profiles available within CMAQ. Recent versions of the emission processor SMOKE also produce speciated PM output for 6 species, including sulfate (PSO4), nitrate (PNO3), EC (PEC), organic aerosols (POA), other fine PM (PMFINE), and coarse PM (PMC). The PM modules require speciated and size-specific PM emission inputs. As shown in Figure 2-1, a preprocessor, mode2sec_emis (available upon request) can be used to generate such inputs based on the $PM_{2.5}$ and PM_{10} emissions. Based on the speciation profiles selected in CMAQ, the mass of individual species is assigned to Aitken, fine, or coarse modes. The mass of the species in each section is then calculated by numerical integration. The user can also specify a PM speciation profile that is specific to the application by modifying the fraction of each species in the pre-processor code and recompiling after the change is made. Speciated PM emissions from SMOKE can be assigned to sections using the same pre-processor. PSO4 (which does not include sea salt sulfate), PNO3, PEC, POA, and PMFINE are assigned to the fine sections and PMC is assigned to the "other" species in the coarse sections. The number of particles emitted into each particle size section contributes to changes in the particle number concentrations and is, therefore, included in the aerosol emission species table (i.e., AE_EMIS.EXT), although the number of particles is not generated by the emission preprocessor and not included in the emission input file. Number emissions are internally calculated based on the total emissions of PM mass in a size section in the subroutine called "rdemis.F", assuming an initial particle size distribution for the emitted particles.

Initial and boundary conditions are also needed to drive the new aerosol modules with sectional PM representation. The original modal PM module of Models-3/CMAQ requires speciated IC/BC inputs for the Aitken, accumulation, and coarse modes. Two processors are available (upon request) to prepare sectional inputs for the initial conditions (three-dimensional) and boundary conditions (three-dimensional and time-varying), respectively, from the modal inputs. In these preprocessors, mode2sec_ic and mode2sec_bc, the mass of PM species in each mode is redistributed among the corresponding sections of the particle size distribution (see Figure 2-1). Note that the initial conditions and boundary conditions for the number of particles for each size section are not generated by the IC/BC preprocessors. They are internally calculated in subroutines “initscen.F” and “rdbcon.F” , respectively.

2.3.2 Inputs for mercury simulations

Two additional input files are needed for mercury simulations. The first is a two-dimensional NetCDF format file (saltwater_frac.ncf) containing a single variable representing the salt water fraction in each grid cell within the domain. The salt water fraction file can be created from the CMAQ meteorology file "GRIDCRO2D" as follows. The GRIDCRO2D file contains the variable DLUSE which represents the dominant land use in any grid cell. DLUSE is 7 when the cell is mostly water. The variable ‘SALTWATER_FRAC’ is set to one when DLUSE is 7 and zero otherwise. Note that grid cells within the modeling domain that contain fresh water should also be set to zero. If more detailed salt water fractions are available to the user, the variable ‘SALTWATER_FRAC’ may be set accordingly. The second input file is an ASCII file (CL_FILE) that contains the concentration of daytime chlorine, nighttime chlorine and hydrochloric acid at different levels. The format of the ASCII file is similar to the J_TABLE used in all CMAQ simulations and is shown below

```
12  LEVELS (m)
   'CL2D',1.0E-05,1.0E-05,1.0E-05,1.0E-05,1.0E-05,1.0E-05
          5.0E-06,5.0E-06,5.0E-06,5.0E-06,5.0E-06,5.0E-06  Daytime Cl2 conc
   'CL2N',1.5E-04,1.0E-04,7.5E-05,5.0E-05,5.0E-05,5.0E-05
          5.0E-06,5.0E-06,5.0E-06,5.0E-06,5.0E-06,5.0E-06  Nighttime Cl2 conc
   'HCL', 1.0E-03,1.0E-03,1.0E-03,1.0E-03,1.0E-03,1.0E-03
          1.0E-03,1.0E-03,1.0E-03,1.0E-03,1.0E-03,1.0E-03  HCl conc
```

2.3.3 Inputs for APT simulations

In addition to the standard files required by Models-3/CMAQ, three files are required for a plume-in-grid application with Models-3/CMAQ-APT. Two of these files provide information on the point sources that are being simulated by the plume model and are in NetCDF format. These files are prepared using a processor developed by MCNC as part of this study under a separate EPRI-sponsored project. The third file is the control file containing the model options, and is in ASCII format. For additional details, please refer to Section 3-1 of the CMAQ-APT user's guide provided in Appendix B.

2.4 Output Files

Models-3/CMAQ allows the user to specify the output species in extension files for instantaneous gas-phase and particulate-phase concentrations (e.g., GC_CONC.EXT, AE_CONC.EXT, NR_CONC.EXT) as well as the dry and wet deposition fluxes of particulate and soluble gaseous compounds (e.g., GC_DDEP.EXT, AE_DDEP.EXT, GC_WDEP.EXT, AE_WDEP.EXT). The output files are in NetCDF format in three dimensions for concentrations and in two dimensions for wet and dry deposition fluxes. In addition, hourly average concentrations outputs are controlled in the run script for gas-phase and particulate species.

CMAQ-MADRID offers two new output features, both of which are controlled in the run script. The first option, activated by setting the flag MAKE_RESTART to "T" in the run script, is to generate a restart file (with prefix RESTART). The restart file contains the three-dimensional fields for all simulated species at the simulation time specified in the run script (TRESTART). It is frequently desirable to break down long-term simulations into subperiods that are easier to manage in terms of file size and logistics. The purpose of the restart file is to generate a 3-D concentration file for all species at the end of a subperiod that can be used as the initial condition for the next subperiod, without having to output instantaneous concentrations of every species for every hour. The second feature is the output of cumulative deposition amounts, which is useful for acid deposition and mercury deposition simulations. This feature is activated by setting the flag MAKE_CUMULDEP to "T". The period to accumulate deposition is specified in the run script by assigning the TACCUM variable. In CMAQ, hourly deposition outputs are controlled using GC_DDEP.EXT and AE_DDEP.EXT files for dry deposition and GC_WDEP.EXT and AE_WDEP.EXT files for wet deposition. The variables specified in those extension files will automatically be included in the cumulative deposition output. Calculating the cumulative deposition reduces the disk space and post processing burden in long-term simulations. The cumulative output file names have the prefix of DDEP_ACCUM and WDEP_ACCUM, respectively, for dry and wet deposition.

3. SUBROUTINES

3.1 PM Modules

The subroutines contained in the sectional PM module are listed in Table 3-1 for MADRID 1. Key functions performed in the subroutines are listed, together with the names of the calling program(s) and the subroutine(s) used. These subroutines are housed in the PM (aero) repository of Models-3/CMAQ under the aero_MADRID1 subdirectory. Note that subroutines used in ISORROPIA are collected in aeroiso.F, isocom.F, isofwd.F, and isorev.F. Subroutines used in the CMU hybrid approach are contained in block.F, diff.F, diffund.F, dvode.F, eqpart.F, equaer.F, errprt.F, hybrid.F, madm.F, negchk.F, and step.F. These subroutines are not listed in further detail. Include files that are specific to the PM module are also listed in Table 3-1. These files (except for HETERPAR.EXT, ARRAYMAP.EXT and PARAMETER.EXT) are located in the same repository directory as the source code (akin to the original modal version of Models-3/CMAQ) and are copied into the working directory with the PM source codes before Models-3/CMAQ is compiled. Mechanism- and/or size-dependent files (e.g., ARRAYMAP.EXT, PARAMETER.EXT, HETERPAR.EXT) are located in the include directories listed in Table 2-3.

Except for the SOA module, the same PM subroutines are used for both versions of MADRID. However, some subroutines may contain module-specific lines of codes due to the differences in the representation of SOA. MADRID 2 uses the AER/EPRI/Caltech (AEC) module for SOA formation. Subroutines used in the AEC SOA module are presented in Table 3-2. These programs replace the CALTECH_SOA, CALTECH_SOA_REV, and SOASUB modules of Table 3-1. Note that decorated names that end with an underscore are used in C subroutines that are called by Fortran subroutines (e.g., aec_soa_ in C; aec_soa in the Fortran calling program) or that call Fortran subroutines (e.g., unifac_ in the calling C program; unifac in the Fortran subroutine). Note also that the *variable notation indicates that a pointer or an array is being passed as an argument in a C subroutine. The variable[] notation is also used for an array argument in C. Arrays are not explicitly noted in subroutine arguments in Fortran.

Table 3-1. Subroutines contained in MADRID 1.

Subroutine (Arguments; Return values)	Function(s)	Called By	Routine(s) Called
AERO SUBST_GRID_ID (CGRID, DPCTR, JDATE, JTIME, TSTEP)	Interfaces between the PM module and modules for other processes in Models-3/CMAQ	SCIPROC	NEXTIME, BOXAER
AEROCMU (GAS, CAERO, TEMPK, RH, PRES, Q, NG)	Interfaces between the PM module and the module for the CMU hybrid approach for gas/particle mass transfer calculation	AERODRIV	STEP, EQPART, HEQDYN
AERODRIV (TEMPE, AHE, TEMPE1, AHE1, PRES, USTAR)	Main aerosol driver	BOXAER	AEROCMU, AEROEQ, MASSFIX, NUCLRATE, WVPRE
AEROEQ (GAS, CAERO, TEMPK, RH, USTAR)	Interfaces between the PM module and the modules for the CIT hybrid and full equilibrium approaches for gas/particle mass transfer calculation	AERODRIV	AEROISO, CALTECH_SOA, AEROPARAM
aeroiso.F; isocom.F; isorev.F; isofwd.F	ISORROPIA routines	AEROEQ, EQPART, EQUAER	(various, not listed)
AEROPARAM (TEMPE, IAERO, USTAR, KN, SC, ST, VSETT)	Calculates the Knudsen, Schmidt, and Stokes numbers as a function of the input temperature and particle size.	AEROEQ,	QGAUS, FUNDIFF
AERORATE (CNBLK_AE, BLKLEN, BTEMP, BPRESS, RTDAT_AE, BLKDP)	Calculates the first-order heterogeneous loss rates of gases on particles with a specified size distribution	CALCKS	--
BISECTNUCL (A, B, E, N, XM)	Performs iterations to solve g(x) = 0 based on the Bisection method	NUCLRATE	G
block.F; diff.F; diffund.F; dvode.F, eqpart.F; equaer.F; errprt.F; hybrid.F; madm.F; negchk.F; step.F	Routines for the CMU hybrid approach	AEROCMU	(various, not listed)

Table 3-1. Subroutines contained in MADRID 1 (continued).

Subroutine (Arguments; Return Values)	Function(s)	Called by	Routine(s) Called
BOXAER (DT, CONCS, TEMPEK, TEMPEK1, PRES, QV, QV1, DENS, USTAR, COL, ROW, LVL, DPCTR)	Interfaces between Models-3/CMAQ and Aerosol routines	AERO_DRIVER	AERODRIV, INITDSECF
CALTECH_SOA (WORG, GASORG, PARTORG , CURTEMP)	Calculates total absorbing organic matter for use in Odum/Griffin SOA partition. Calls SOASUB. Checks SOA solution	AEROEQ, EQPART	SOASUB
CALTECH_SOA_REV (K, Q, CURPOC, CURTEMP)	Calculates partial pressures of organics at the surface of the particles.	STEP	--
FUNDIFF (DP)	Defines external function	QGAUS	--
G (X)	Defines external function G used in BISECTNUCL	BISECTNUCL	--
INITDSECF	Initializes the particle size structure for the CMU hybrid approach	BOXAER	--
MASSFIX (CAERO0, CAERO, AERGAS0, AERGAS)	Ensures mass conservation in the CMU hybrid approach	AERODRIV	--
MOVEBIN SUBST_GRID_ID (CGRID0, CGRID, DPCTR)	Interfaces between the sciproc.F and the growth modules for the PM	SCIPROC	MOVECENTER
MOVECENTER (CAERO, COL, ROW, LAY)	The moving-center scheme for PM growth	MOVEBIN	--
MOVESETUP SUBST_GRID_ID (CGRID, CGRID0)	Restores the old CGRID array before all growth processes	SCIPROC	CGRID_MAP

Table 3-1. Subroutines contained in MADRID 1 (continued).

Subroutine (Arguments; Return Values)	Function(s)	Called by	Routine(s) Called
NUCLRATE (GMASS, GNUM, AERGAS, TEMPK, RH)	Calculates new particle formation rates based on McMurry and Friedlander's method	AERODRIV	BISECTNUCL
QGAUS (FUNC, A, B, SS)	Program for Gaussian quadrature integration solver	AEROPARAM	--
SIZEINIT (DPCTR)	Initializes the particle size distributions that are based on the CIT or GATOR size structures	INITSCEN	--
SOASUB (MSUM, WORG, PARTORG, GASORG, CURTEMP)	Calculates equilibrium partition of individual SOA species in Odum/Griffin SOA formulation.	CALTECH_SOA	--
Include Files	**Content**	**Used by**	
ARRAYMAP.EXT	Mapping information between main and working arrays	various subroutines	
DYNAMIC.EXT, EQUAER.EXT	Common variables used in the routines for the CMU hybrid approach	various subroutines	
HETERPAR.EXT	Assigns species indices for gas-phase species that partition into heterogeneous reactions in cloud droplet and on aerosol surface	AERORATE, CLDRATE	
ISRPIA.EXT	Common variables used in ISORROPIA	ISORROPIA subroutines	
MW.EXT	Molecular weight data	various subroutines	
PARAMETER.EXT	Defines variables, arrays, and constants used in the PM module	various subroutines	
PARTITION.EXT	SOA partition coefficients based on experimental data of Odum/Griffin	CALTECH_SOA	

Table 3-2. Subroutines and include files used in the AEC SOA module.

Subroutine (Arguments; Return Values)	Function(s)	Called by	Routine(s) Called
aec_soa_ (called as aec_soa) (*tempk, *rh, *worg, *gasorg, *partorg, *lwc, *protonconc, *Organion, *DeltaLWC, *tboaflag)	Main SOA program.	AEROEQ EQPART	amain, bmain
amain (*aero, *aeros; deltaLWC)	Determines phases of particles. Specify initial guesses for PM composition. Calls Newton solver to solve aqueous PM composition. Calculates dry particle composition. Calls ZSR to calculate organic water content	aec_soa	amainabs, saturation, newt_double, TypeA, ZSR
amainabs (aabaero[])	Main program for absorption partitioning Type A compounds with no existing water	amain	newt_double, TypeAabs
bmain (gas[], aero[],)	Calculates total condensables. Makes initial guesses for particulate-phase concentrations for use in newt.	aec_soa	newt, TypeB
bpartfuncapprox (n, aero)	Calculates approximated equilibrium for hydrophobic SOA by assuming fixed absorbing medium amount and composition	TypeB	--
bpartitionfunc (n, aerof[], ff[])	Calculates the deviation from equilibrium for the absorptive SOA given set of concentrations and partition constants	fmin1 (a subroutine used by newt1)	--
bunidriver (XPASS[], GAMMA[], n)	Sets up unifac parameters for hydrophobic SOA and calls unifac subroutine	TypeB	unifac_
kpart (n, ac[], mwom, KB[])	Calculates partition coefficients for hydrophobic compounds	Type B	--
lwcorgfunc (n, dLWC[], f[])	Calculates the deviation of water activity from RH	fmin1_double (a subroutine used by newt1_double)	unidriver

Table 3-2. Subroutines and include files used in the AEC SOA module (continued).

Subroutine (Arguments; Return Values)	Function(s)	Called by	Routine(s) Called
newt newt1 newt_double newt1_double	Solves simultaneous equations using the Newton/Line Search method from Numerical Recipes (Press et al., 1997)	bmain Typeb amain ZSR	(numerous, not listed)
partfunc2 (n, aero[],gamma[], f[])	Calculates deviation from equilibrium for hydrophilic solutes	TypeA	--
partiaabs (n, aerod[], ff[])	Calculates deviation from equilibrium for hydrophilic solutes in absorption calculation	fmin1_double (which is called by TypeAabs)	--
partiabsapprox (n, aerod[]);	Calculates approximate partition of Type A compounds when LWC = 0 by assuming fixed absorbing medium amount and composition	Typeaabs	
saturation (tot, vp, *pmconc, *gasconc)	Calculates saturation partition	amain	--
TypeA (n, aero[],f[])	Calculates Henry's law activity coefficient based on Raoult's law activity coefficients from unifac routines. Calls partfunc2 to calculate deviation from equilibrium. Calculates negative ion (anion) concentration	fmin_double (a subroutine used by newt_double) or amain	unidriver, partfunc2
TypeAabs	Given initial SOA concentrations, calls subroutines to calculate activity coefficients, partition coefficients, and equilibrium SOA concentrations. Calculates difference between input and equilibrium SOA concentrations.	fmin_double (a subroutine used by newt_double) or by amainabs	newt1_double, unidriver, partiaabs, Kpart
TypeB (n, aero[], f[])	Given initial SOA concentrations, calls subroutines to calculate activity coefficients, partition coefficients, and equilibrium SOA concentrations. Calculates difference between input and equilibrium SOA concentrations.	fmin (a subroutine used by newt) or bmain	bpartfuncapprox, bpartitionfunc. bunidriver, Kpart, newt1

Table 3-2. Subroutines and include files used in the AEC SOA module (continued).

Subroutine (Arguments; Return Values)	Function(s)	Called by	Routine(s) Called
unidriver (XPASS[], GAMMA[], n)	Sets up unifac parameters for hydrophilic SOA and calls unifac subroutine	TypeA	unifac_
unifac (called using unifac_) (NMOL, NFUNC, NU, X, A, RG, QG, Z, TEMP, GAMA)	Calculates activity coefficients	bunidriver unidriver (called as unifac)	--
ZSR (*aerog; dLWC[1])	Calculates the amount of water associated with hydrophilic SOA, either by an iterative procedure based on unifac or by ZSR	amain	newt1_double, lwcorgfunc
Include Files	**Content**	**Used by**	
binsolu.h	binary solution data for aqueous particles	ZSR	
bunifacparam.h	unifac parameters for hydrophobic SOA	bunidriver	
glo.h	global variables used in AEC_SOA routines	all subroutines in AEC_SOA routines	
glodef.h	global parameters used in AEC_SOA routines	all subroutines in AEC_SOA routines	
nr.h; nr_double.h; nrutil.h	function definitions used in numerical recipes	newt, newt1, newt_double, newt1_double and other numerical recipe subroutines	
unifacparam.h	unifac parameters for hydrophilic SOA	unidriver	

The globally convergent Newton/line search method is used to solve simultaneous equations in the AEC SOA module. The subroutines newt, newt1, newt_double, and newt1_double are copies (with minor changes) of the Newton method routine used in various points in the SOA code. Routines used by newt, newt1, newt_double, and newt1_double can be found in Press et al. (1997) and are not listed in detail here.

3.2 Cloud Modules

CMAQ-MADRID is currently coupled to the RADM cloud module (cloud_RADM). The original RADM module is modified for interfacing with sectional aerosol modules in MADRID. The RADM cloud module simulates the physical and chemical processes occurring in clouds, including aerosol scavenging by cloud droplets, aqueous-phase chemistry, and removal by wet deposition. A modified RADM aqueous-phase chemical mechanism (Chang et al., 1987; Walcek and Taylor, 1986), which includes updated reaction rate constants, is used in the cloud_RADM module.

The subroutines contained in the cloud_RADM modules are listed in Table 3-3, along with brief descriptions of their functions. These subroutines are used in the simulations with gas-phase and heterogeneous chemistry and with CAMQ-MADRID with various size sections (i.e., all cloud_RADM_* directories). The names of the subroutine(s) by which each subroutine is called and those that it calls (if any) are also given in the table. The listing of the subroutines is provided in alphabetical order. The cloud_RADM module requires only one include file: AQ_PARAMS.EXT, which is required for a specific gas-phase mechanism (e.g., CBM-IV, RADM2-CI4 or SAPRC99) and a specific size resolution (e.g., 2 or multiple size sections). Those mechanism- and size-dependent AQ_PARAMS.EXT files are located in the include directories listed in Table 2-3.

Table 3-3. The subroutines contained in the cloud_RADM modules used with CMAQ-MADRID .

Subroutine (Arguments)	Function	Called by	Routine(s) called
AQCHEM (JDATE, JTIME, TEMP, PRES_PA, TAUCLD, PRCRATE, WCAVG, WTAVG, AIRM, ALFA0, ALFA3, GAS, AEROSOL, GASWDEP, AERWDEP, HPWDEP)	Computes concentration changes in cloud due to aqueous-phase chemistry, scavenging, and wet deposition.	AQMAP	
AQMAP (JDATE, JTIME, WTBAR, WCBAR, TBARC, PBARC, CTHK1, AIRM, PRATE1, TAUCLD, POLC, CEND, REMOV, REMOVAC, ALFA0, ALFA3)	Provides an interface processor between the cloud dynamics module(s) and the aqueous-phase chemistry module.	RESCLD RADMCLD	CGRID_MAP AQCHEM
CLDPROC SUBST_GRID_ID (CGRID, JDATE, JTIME, TSTEP)	Provides an interface processor between the cloud module and the modules for other processes.	SCIPROC	CGRID_MAP RESCLD RADMCLD NEXTIME
CLDPROC_LWC SUBST_GRID_ID (JDATE, JTIME, TSTEP, TOT_WCBAR)	Calculates total cloud liquid water content to be used in the calculation of the gas-cloud heterogeneous reaction rate constant.	CHEM	RESCLD_LWC RADMCLD_LWC NEXTIME
GETALPHA (MASSI, LWC, T, P, RHOAIR, ALFA0, ALFA3)	Calculates the in-cloud scavenging coefficients for aerosol number and mass.	SCAVWDEP	
HLCONST (NAME, TEMP)	Sets the Henry's law constant of species at a given temperature.	SCAVWDEP	
INDEXN (NAME1, N, NAME2, INDICES)	Searches for all occurrences of NAME1 in list NAME2.	SCAVWDEP AQMAP	
RESCLD SUBST_GRID_ID (CGRID, JDATE, JTIME, TSTEP, N_SPC_WDEP, WDEP_MAP, DEP)	Calculates cloud characteristics for grid-resolved clouds and uses them to generate cumulative and net timestep deposition, cloud top, cloud bottom, and pressure at the lifting level.	CLDPROC	CGRID_MAP NEXTIME SCAVWDEP AQ_MAP
RESCLD_LWC SUBST_GRID_ID (JDATE, JTIME, TSTEP, TOT_WCBAR)	Calculates liquid water content for resolved-scale clouds to be used in the calculation of the gas-cloud heterogeneous reaction rate constant.	CLDPROC_LWC	NEXTIME

Table 3-3. The subroutines contained in the cloud_RADM modules used with CMAQ-MADRID (continued).

Subroutine (Arguments)	Function	Called by	Routine(s) called
RADMCLD SUBST_GRID_ID (CGRID, JDATE, JTIME, TSTEP, ICLDTYPE, N_SPC_WDEP, WDEP_MAP, DEP)	Calculates cloud characteristics for subgrid convective precipitating clouds and subgrid non-precipitating clouds, and uses them to generate cumulative and net timestep deposition, cloud top, cloud bottom, and pressure at the lifting level.	CLDPROC	CGRID_MAP NEXTIME SCAVWDEP AQ_MAP
RADMCLD_LWC SUBST_GRID_ID (JDATE, JTIME, TSTEP, ICLDTYPE, TOT_WCBAR)	Calculates liquid water content for subgrid convective clouds be used in the calculation of the gas-cloud heterogeneous reaction rate constant.	CLDPROC_LWC	NEXTIME
SCAVWDEP (JDATE, JTIME, WTBAR, WCBAR, TBARC, PBARC, CTHK1, AIRM, PRATE1, TAUCLD, POLC, CEND, REMOV, REMOVAC, ALFA0, ALFA3)	Computes in-cloud scavenging and wet removal.	RESCLD RADMCLD	CGRID_MAP GETALPHA
Include Files	**Content**	**Used by**	
AQ_PARAMS.EXT	Pointers for arrays GAS and AEROSOL used in the CBM-IV, RADM2-CI4, SAPRC99 or CACM gas-phase chemistry module and parameters used in aerosol activation parameterization	AQMAP, AQCHEM	

The cloud_CMU subroutines are passive in the current version of CMAQ-MADRID. The CMU module simulates the same physical and chemical processes as RADM. These two cloud modules have the same treatments of cloud dynamics and aerosol scavenging by cloud droplets. They differ primarily in the aqueous-phase chemical mechanism used. The cloud_CMU module uses the CMU aqueous-phase chemical mechanism (Strader et al., 1998). A discussion of these routines is included in Appendix C.

3.3 Mercury Modules

There are three mercury-specific modules, placed under "driver", "chem", and "cloud" processes within the CMAQ-MADRID repository. A list of mercury-specific files and their functions is provided in Table 3-4.

3.4 APT Modules

The file structure within the APT module in the CMAQ-MADRID repository is illustrated in Figure 4-1 of Appendix B.

Table 3-4. Selected subroutines and include files used in CMAQ-MADRID 1 simulations with mercury.

Location/Subroutine (Arguments)	Function	Called by	Routine(s) called
CCTM/src/driver/ ctm_MADRID1_hg/ sciproc (CGRID, DPCTR, JDATE, JTIME, TSTEP, ASTEP)	Calls mercury chemistry driver	DRIVER	All physical and chemical subroutines, LOAD_RHOJ, ADJADV,COUPLE, DECOUPLE, HGCHEM
CCTM/src/driver/ ctm_MADRID1_hg/ hgchem (CGRID, JDATE, JTIME, TSTEP)	Driver for gas-phase chemistry; reads salt water fraction and concentrations of chlorine species	SCIPROC	HGGASCHEM
CCTM/src/chem/ ebi_cb4_MADRID1a_hg/ hggaschem (CHG0, CO3, COH, CHO2O, CHCL, CCL2, TEMP, PRES, DELT, DHG0)	Calculates gas-phase mercury reactions	HGCHEM	
CCTM/src/chem/ ebi_cb4_MADRID1a_hg/ hggasprm.ext	Lists parameters in gas-phase mercury reactions, e.g., reaction rate constants	HGGASCHEM	
CCTM/src/chem/ ebi_cb4_MADRID1a_hg/ hrprodloss	Specifies production and loss terms for the ebi solver; modified to include Hg.	HRSOLVER	
CCTM/src/chem/ ebi_cb4_MADRID1a_hg/ hrinit	Declares and assigns species to chemistry solver arrays; modified to include Hg	HRDRIVER	
CCTM/src/chem/ ebi_cb4_MADRID1a_hg/ hrdata_mod	Declares gas-phase species; modified to include Hg	HRDRIVER, HRINIT, HRSOLVER, HRPRODLOSS, HRRATES, HRG1, HRG2, HRG3, HRG4	
CCTM/src/cloud/ cloud_RADM_aero_MADRID1_hg/ hlconst (name, temp, effective, hplus)	Calculates Henry's law constant at a given temperature; added Henry's law constants for Cl and Hg species	AQCHEM, HGAQCHEM, SCAVWDEP	
CCTM/src/cloud/ cloud_RADM_aero_MADRID1_hg/ cldproc (cgrid, totwcbar, jdate, jtime, tstep)	RADM and resolved cloud process driver; modified to calculate total PM for use with Hg simulations	SCIPROC	RADMCLD, RESCLD

Table 3-4. Selected subroutines and include files used in CMAQ-MADRID 1 simulations with mercury (continued).

Location/Subroutine (Arguments)	Function	Called by	Routine(s) called
CCTM/src/cloud/ cloud_RADM_aero_MADRID1_hg/ rescld (cgrid, totwcbar, jdate, jtime, tstep)	Resolved cloud process processor; modified to include variables used in Hg chemistry	CLDPROC	SCAVWDEP, AQMAP
CCTM/src/cloud/ cloud_RADM_aero_MADRID1_hg/ AQ_MAP (JDATE, JTIME, WTBAR, WCBAR, TBARC, PBARC, CTHK1, AIRM, PRATE1, TAUCLD, POLC, CEND, REMOV, REMOVAC, ALFA0, ALFA2, ALFA3, HCLC, CL2C, PM10C)	Interface between cloud dynamics and aqueous chemistry modules; modified to include Hg and Cl variables	RADMCLD, RESCLD	AQCHEM
CCTM/src/cloud/ cloud_RADM_aero_MADRID1_hg/ RADMCLD (CGRID, JDATE, JTIME, TSTEP, ICLDTYPE, N_SPC_WDEP, WDEP_MAP, DEP, CONV_WCBAR, PM10)	Calculates cloud characteristics; modified to include Hg variables	CLDPROC	SCAVWDEP AQCHEM
CCTM/src/cloud/ cloud_RADM_aero_MADRID1_hg/ AQCHEM (JDATE, JTIME, TEMP, PRES_PA, TAUCLD, PRCRATE, WCAVE, WTAVG, AIRM, ALFA0, ALFA3, GAS, AEROSOL, GASWDEP, AERWDEP, HPWDEP, CL2C, PM10C)	Computes concentration change in clouds due to aqueous chemistry, scavenging and wet deposition; modified to include Hg variables	AQINTR	HLCONST
CCTM/src/cloud/ cloud_RADM_aero_MADRID1_hg/ HGAQCHEM (PHG00, PHG20, XL, PO3, PCL2, PSO2, CPH, OHL, HO2L, CPM, TEMPK, SO21, SO22, SO2H, O3H, HCLH, DELT, PHG0F, PHG2F, HG0L, HG2L, DHG0)	Performs Hg aqueous phase chemistry calculations	AQCHEM	HLCONST, SOLVE
CCTM/src/cloud/ cloud_RADM_aero_MADRID1_hg/ HGAQSPRM.EXT	Defines equilibrium and other constants and indices for Hg calculations	HGAQCHEM	
include/release/ FILES_CTM.Hg.EXT	Lists files opened during simulation; includes saltwater fraction and chlorine concentrations files	various	
include/release/ cb4_aqRADM_hg_aeMADRID1a_2sec	Include files for mercury simulations, see Table 2-2.	various	

4. TEST CASES

Four example simulations are provided to demonstrate the application of the CMAQ-MADRID with mercury and APT described in this document. In the tape accompanying this user's guide, emissions, meteorology, and result files are provided for these simulations. Each example is organized in a directory structure containing subdirectories for inputs (inp), outputs (out), simulation scripts (sim), and source codes (src).

The first example demonstrates MADRID 1 (with the abridged SOA representation) using 8 particle size sections. The test simulation is a 24-hour run during the August 1987 SCAQS episode. The domain covers the Los Angeles basin with 5-km grid resolution. Cloud processes are simulated using the RADM module. Note that new particle formation and the growth/formation of particles due to cloud droplets are simulated in this example. The script for building Models-3/CMAQ-MADRID with the specifications used in this first test case is shown in Appendix A-1.

The second example is an aerosol-only simulation with MADRID 2, using two size sections to represent aerosols. The domain and simulation period are identical to Example 1. As shown in the script in Appendix A-2, cloud modules and heterogeneous reactions in cloud droplets are turned off by selecting cloud_noop. New particle formation is simulated.

The third example demonstrates the use of CMAQ-MADRID for simulating mercury. The 24-hour test-case is conducted over the continental United States, and the grid resolution is 36 km. Note the selection of mercury specific modules for "driver", "chem" and "cloud", and the use of FILES_CTM.Hg.EXT in the build script (Appendix A-3). In the input directory of Example 3, there are new directories for the saltwater concentration file and the concentrations of chlorine species.

The last example involves the application of the plume-in-grid model, CMAQ-APT, to the 1995 Southern Oxidants Study. The model is used to simulate ozone and nitric acid formation due to emissions from two power plants in the Nashville/Tennessee valley area for a 12-hour period. In the build script in Appendix A-4, note the directives to build the ODE library (libodepack.a) and the APT library (libsci.a) and the linking of

the libraries to make the CMAQ executable. Additional input files are found in the input directory of Example 4, as described in Appendix B.

5. REFERENCES

Byun, D.W., and J.K.S. Ching, 1999. Science algorithms of the EPA Models-3 Community Multiscale Air Quality (CMAQ) Modeling System. EPA/600/R-99/030. U.S. Environmental Protection Agency, Research Triangle Park, NC 27711.

Chang, J.S., R.A. Brost, I.S.A. Isaksen, S. Madronich, P. Middleton, W.R. Stockwell, and C.J. Walcek, 1987. A three-dimensional Eulerian acid deposition model: Physical concepts and formulation, *J. Geophys. Res*., **92**, 14681-14700.

Griffin, R.J., D.R. Cocker III, R.C. Flagan and J.H. Seinfeld, 1999. Organic aerosol formation from the oxidation of biogenic hydrocarbons, *J. Geophys. Res.*, **104**, 3555-3567.

Griffin, R.J., D. Dabdub, J.H. Seinfeld, 2002. Secondary organic aerosols: I. Atmospheric chemical mechanism for production of molecular constituents, *J. Geophys. Res.*, in press.

Jacob, D., 2000. Heterogeneous chemistry and tropospheric ozone, *Atmos. Environ.*, **34**, 2132-2159.

Jacobson, M.Z., 1997a. Development and application of a new air pollution modeling system – II. Aerosol module structure and design, *Atmos. Environ.*, **31**, 131-144.

Jacobson, M.Z. and R.P. Turco, 1995. Simulating condensational growth, evaporation, and coagulation of aerosols using a combined moving and stationary size grid, *Aerosol Sci. Technol.*, **22**, 29-92.

Karamchandani, P., L. Santos, I. Sykes, Y. Zhang, C. Tonne, C. Seigneur, 2000. Development and evaluation of a state-of-the-science reactive plume model, *Environ. Sci. Technol*. **34,** 870-880.

Karamchandani, P., C. Seigneur, K. Vijayaraghavan and S.-Y. Wu, 2002. Development and application of a state-of-the-science plume-in-grid model, *J. Geophys. Res*., **107,** 4403-4415.

McMurry, P.H. and S.K. Friedlander, 1979. New particle formation in the presence of an aerosol, *Atmos. Environ.*, **13**, 1635-1651.

Nenes, A., C. Pilinis and S.N. Pandis, 1999. Continued development and testing of a new thermodynamic aerosol module for urban and regional air quality models, *Atmos. Environ.*, **33**, 1553-1560.

Odum, J.R., T.P.W. Jungkamp, R.J. Griffin, H.J.L. Forstner, R.C. Flagan and J.H. Seinfeld, 1997. Aromatics, reformulated gasoline, and atmospheric organic aerosol formation, *Environ. Sci. Technol.*, **31**, 1890-1897.

Press, W.H., S.A. Teulolsky, W.T. Vetterling, B.P. Flannery, 1997. Numerical Recipes in C, Second Edition, 3.5" Diskette for Windows3.1, 95, or NT. Cambridge University press, Cambridge, UK.

Pun, B.K., Y. Zhang, K. Vijayaraghavan, S.Y. Wu, C. Seigneur, J.H. Seinfeld, 2001. Development of new aerosol modules and incorporation into Models-3/CMAQ, AER Draft report to EPRI, Palo Alto, CA.

Pun, B.K., R.J. Griffin, C.Seigneur, J.H. Seinfeld, 2002. Secondary organic aerosol 2. Thermodynamic model for gas/particle partitioning of molecular constituents, *J. Geophys. Res.*, **107**, doi:10.1029/2001JD000542.

Pun, B., S.-Y. Wu, C. Seigneur, J.H. Seinfeld, R.J. Griffin and S.N. Pandis, 2003, Uncertainties in modeling secondary organic aerosols: three-dimensional modeling studies in Nashville/western Tennessee, *Environ. Sci. Technol.*, **37**, 3647-3661.

Pun, B.K.; Seigneur, C., Knipping, E, 2004. Optimizing Secondary Organic Aerosol Representation in Particulate Matter Air Quality Models. AWMA Regional and Global Perspectives on Haze: Causes, Consequences and Controversies- Visibility Specialty Conference, Asheville, NC, October, 2004.

Seigneur, C., K. Vijayaraghavan, K. Lohman, P. Karamchandani and C. Scott, 2004. Global source attribution for mercury deposition in the United States, *Environ. Sci. Technol.*, **38**, 555-569.

Strader, R., C. Gurciullo, S. Pandis, N. Kumar, F.W. Lurmann, 1998. Development of gas-phase chemistry secondary organic aerosol, and aqueous-phase chemistry modules for PM modeling, STI Final report to Coordinating Research Council, Atlanta, GA.

Venkatram, A. and J. Pleim, 1999. The electrical analogy does not apply to modeling dry deposition of particles, *Atmos. Environ.*, **33**, 3075-3076.

Walcek, C.J. and G.R. Taylor, 1986. A theoretical method for computing vertical distributions of acidity and sulfate production within cumulus clouds, *J. Atmos. Sci.*, **43**, 339-355.

Zhang, Y., B. Pun, K. Vijayaraghavan S.-Y. Wu, and C. Seigneur, 2002. CMAQ-MADRID Technical Documentation, AER Report to EPRI, Document Number CP083-02-03a, Palo Alto, CA.

Zhang, Y., Pun, B.K., Vijayaraghavan, K., Wu, S.-Y, Seigneur, C., Pandis, S.N., Jacobson, M.Z., Nenes, A. Seinfeld, J.H., 2004. Development and application of the Model of Aerosol Dynamics, Reaction, Ionization, and Dissolution (MADRID), *J. Geophys. Res.*, **109** (D01202), 2003JD003501.

APPENDIX A

A-1 Script used to compile Models-3/CMAQ with aero_MADRID1 and cloud_CMU_aero_MADRID1.

```
#! /bin/csh -f

#> RCS file, release, date & time of last delta, author, state, [and locker]
#> $Header$

#> what(1) key, module and SID; SCCS file; date and time of last delta:
#> %W% %P% %G% %U%

### setenv M3MODEL /project/work/rep            # code archive
### setenv M3LIB   /project/air5/sjr/CMAS4.4/distr/rel/lib    # libraries

setenv REPOSITORY /usr2/cp194/CMAQ-MADRID/code/oct2004
setenv M3MODEL $REPOSITORY/models    # code archive
setenv M3LIB   $REPOSITORY/lib       # libraries

set BLD_OS = Linux
 set Base = $cwd

 if ( ! -e $M3MODEL || ! -e $M3LIB ) then
    echo "   $M3MODEL or $M3LIB directory not found"
    exit 1
    endif
 echo "   Model archive path: $M3MODEL"
 echo "         library path: $M3LIB"

 set echo

#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#
#> user choices: cvs archives
 set Project = $M3MODEL/CCTM
 set GlobInc = $M3MODEL/include/release

#> user choices: base directory
#set Base = /your_dir/you/working_dir
 set Base = $cwd

 set APPL  = cb4_aqRADM_aeMD1a_8sec
 set CFG   = cfg.$APPL
 set MODEL = CCTM_$APPL

#> user choices: m3bld command
#set Opt = compile_all  # force compile, even if object files are current
#set Opt = clean_up     # remove all source files upon successful completion
#set Opt = no_compile   # do everything except compile
#set Opt = no_link      # do everything except link
#set Opt = one_step     # compile and link in one step
#set Opt = parse_only   # checks config file syntax
#set Opt = show_only    # show requested commands but doesn't execute them
 set Opt = verbose      # show requested commands as they are executed
 set MakeOpt            # builds a Makefile to make the model

#> user choices:  single or multiple processors
#set ParOpt             # multiple PE's

#> user choices: various modules
```

```
 set Revision = release       # release = latest CVS revision
#set Revision = '"STA3_2"'
#> NOTE: m3bld will try to compile with existing code; it will not retrieve
#>       new (different release) code. So if your "BLD" directory contains
#>       code from a release different than the one you have specified above,
#>       m3bld will tell you, but will merrily compile the original code.
#>       The workaround is to remove your "BLD" directory and start fresh.

#set ModDriver = ( module ctm               $Revision; )
 set ModDriver = ( module ctm_MADRID1      $Revision; )

 if ( $?ParOpt ) then
    set ModPar = ( module par              $Revision; )
    else
    set ModPar = ( module par_noop         $Revision; )
    endif

#set ModInit   = ( module init              $Revision; )
 set ModInit   = ( module init_MADRID      $Revision; )

#set ModAdjc   = ( module adjcon_noop      $Revision; )
 set ModAdjc   = ( module denrate          $Revision; )

 set ModCpl    = ( module gencoor          $Revision; )

#set ModHadv   = ( module hadv_noop        $Revision; )
#set ModHadv   = ( module hppm             $Revision; )
 set ModHadv   = ( module hppm_MADRID      $Revision; )

#set ModVadv   = ( module vadv_noop        $Revision; )
 set ModVadv   = ( module vppm             $Revision; )

#set ModHdiff  = ( module hdiff_noop       $Revision; )
 set ModHdiff  = ( module multiscale       $Revision; )

#set ModVdiff  = ( module vdiff_noop       $Revision; )
#set ModVdiff  = ( module eddy             $Revision; )
#set ModVdiff  = ( module acm              $Revision; )
 set ModVdiff  = ( module eddy_MADRID      $Revision; )

#set ModPhot   = ( module phot_noop        $Revision; )
 set ModPhot   = ( module phot             $Revision; )

 set ModPing   = ( module ping_noop        $Revision; )
#set ModPing   = ( module ping_smvgear     $Revision; )

#set ModChem   = ( module chem_noop        $Revision; )
#set ModChem   = ( module smvgear          $Revision; )
#set ModChem   = ( module ros3             $Revision; )
#set ModChem   = ( module ebi_cb4          $Revision; )
 set ModChem   = ( module ebi_cb4_MADRID1a $Revision; )
#set ModChem   = ( module ebi_saprc99      $Revision; )
#set ModChem   = ( module mebi_radm2_cis4  $Revision; )
#set ModChem   = ( module ros3_MADRID1     $Revision; )

#set ModAero   = ( module aero_noop        $Revision; )
#set ModAero   = ( module aero3            $Revision; )
 set ModAero   = ( module aero_MADRID1a    $Revision; )

#set ModAdepv  = ( module aero_depv_noop   $Revision; )
#set ModAdepv  = ( module aero_depv2       $Revision; )
 set ModAdepv  = ( module aero_depv_MADRID $Revision; )

#set ModCloud  = ( module cloud_noop       $Revision; )
```

```
#set ModCloud  = ( module cloud_radm         $Revision; )
 set ModCloud  = ( module cloud_RADM_aero_MADRID1  $Revision; )

if ($ModCloud[2] == 'cloud_noop')then
   set CVD =
else
   set CVD = -Dcld_proc
endif


#set ModPa     = ( module pa                 $Revision; )
 set ModPa     = ( module pa_MADRID          $Revision; )

 set ModUtil   = ( module util               $Revision; )

 set Modaeaqsolver = ( module ae_aq_solver_MADRID    $Revision; )

#set ModMisc   = ( misc; \
#                  $Base/file1.F \
#                  $Base/file2.F \
#                  $Base/file3.F )

#> user choices: emissions processing in chem or vdiff (default) ...
#set Cemis

#> user choices: vertical layers and mechanism
#set Layers    = 21
 set Layers    = 15
#set Mechanism = radm2_ci4_ae3_aq
#set Mechanism = cb4_ae3_aq
#set Mechanism = saprc99_ae3_aq
 set Mechanism = cb4_aqRADM_aeMADRID1a_8sec
 set Tracer    = trac0               # default: no tracer species

#> user choices: set process analysis linkages
 set PABase    = $GlobInc
 set PAOpt     = pa_noop
#set PABase    = /project/cmaq/yoj/saprc
#set PAOpt     =

#> other user choices set below are:
#>    name of the "BLD" directory
#>    compiler/link flags
#>    library paths
#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#

 set Bld = $Base/BLD_${APPL}
#unset echo
 if ( ! -e "$Bld" ) then
    mkdir $Bld
    else
    if ( ! -d "$Bld" ) then
       echo "   *** target exists, but not a directory ***"
       exit 1
       endif
    endif
#set echo
 cd $Bld

#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#

 set STENEX = ${M3LIB}/stenex_noop/${BLD_OS}
 set MODLOC = ${Base}/MOD_DIR
 if ( ! -d "$MODLOC" ) mkdir -p $MODLOC
```

```
 if ( $?ParOpt ) then
#   set Mpich = /usr/local/mpich-1.2.4       # compiled for ch_shmem
    set Mpich = /share/linux/bin/mpich-ch_p4 # compiled for ch_p4
    set seL = se_snl
    cp -p ${STENEX}/*.mod $MODLOC
    set LIB2 = "-L${M3LIB}/pario/${BLD_OS} -lpario"
    set LIB3 =
    set LIB4 = "-L${Mpich}/lib -lmpich"
    set Str1 = (// Parallel / Include message passing definitions)
    set Str2 = (include SUBST_MPICH ${Mpich}/include/mpif.h;)
    else
    set Mpich =
    set seL = sef90_noop
    cp -p ${STENEX}/noop_*.mod $MODLOC
    set LIB2 =
    set LIB3 =
    set LIB4 =
    set Str1 =
    set Str2 =
    endif

 set LIB1 = "-L${STENEX} -l${seL}"
 set LIB5 = "-L${M3LIB}/ioapi/ioapi_22/Linux2_x86pg -lioapi"
 set LIB6 = "-L${M3LIB}/netCDF/${BLD_OS} -lnetcdf"
 set LIBS = "$LIB1 $LIB2 $LIB3 $LIB4 $LIB5 $LIB6"

 set FC = /usr/pgi/linux86/5.2/bin/pgf90
 set FP = $FC

 set FSTD       = "-Mfixed -Mextend"
 set F_FLAGS    = "${FSTD} -fast -module ${MODLOC} -I."
 set CPP_FLAGS  = ""
#set C_FLAGS    = "-v -O2 -I${Mpich}/include"
 set C_FLAGS    = "-Xc -v -g"
 set LINK_FLAGS = "-Bstatic"

#set Blder = ${M3LIB1}/build/$BLD_OS/m3bld
 set Blder = ${M3LIB}/build/$BLD_OS/m3bld

 set ICL_PAR   = $GlobInc
 set ICL_CONST = $GlobInc
 set ICL_FILES = $GlobInc
 set ICL_EMCTL = $GlobInc
#set ICL_IOAPI = ${M3LIB1}/m3io/Linux
 set ICL_IOAPI = ${M3LIB}/m3io/Linux
 set ICL_VCRD  = $Base/icl
 set ICL_MECH  = $GlobInc/$Mechanism
 set ICL_TRAC  = $GlobInc/$Tracer
 set ICL_PA    = $PABase/$PAOpt

 if ( $?Cemis ) then
    set CV = -Demis_chem
    else
    set CV =
    endif

#> NOTE: To run parallel in a Scyld Beowulf cluster, e.g., remove the
#>       "-Dcluster\" below.

 if ( $?ParOpt ) then   # split to avoid line > 256 char
    set PAR = ( -Dparallel\
                -Dcluster\
                -DINTERPB=PINTERPB\
```

```
                -DM3ERR=PM3ERR\
                -DM3EXIT=PM3EXIT\
                -DM3WARN=PM3WARN\
                -DSHUT3=PSHUT3\
                -DWRITE3=PWRITE3 )

    set Popt = SE
    else
    echo "   Not Parallel; set Serial (no-op) flags"
    set PAR = "-DINTERPB=INTERP3"
    set Popt = NOOP
    endif

 set STX1 = ( -DSUBST_MODULES=${Popt}_MODULES\
              -DSUBST_BARRIER=${Popt}_BARRIER )
 set STX2 = ( -DSUBST_GLOBAL_MAX=${Popt}_GLOBAL_MAX\
              -DSUBST_GLOBAL_MIN=${Popt}_GLOBAL_MIN\
              -DSUBST_GLOBAL_MIN_DATA=${Popt}_GLOBAL_MIN_DATA\
              -DSUBST_GLOBAL_TO_LOCAL_COORD=${Popt}_GLOBAL_TO_LOCAL_COORD\
              -DSUBST_GLOBAL_SUM=${Popt}_GLOBAL_SUM\
              -DSUBST_GLOBAL_LOGICAL=${Popt}_GLOBAL_LOGICAL\
              -DSUBST_LOOP_INDEX=${Popt}_LOOP_INDEX\
              -DSUBST_SUBGRID_INDEX=${Popt}_SUBGRID_INDEX )
 set STX3 = ( -DSUBST_HI_LO_BND_PE=${Popt}_HI_LO_BND_PE\
              -DSUBST_SUM_CHK=${Popt}_SUM_CHK\
              -DSUBST_INIT_ARRAY=${Popt}_INIT_ARRAY\
              -DSUBST_COMM=${Popt}_COMM\
              -DSUBST_MY_REGION=${Popt}_MY_REGION\
              -DSUBST_SLICE=${Popt}_SLICE\
              -DSUBST_GATHER=${Popt}_GATHER\
              -DSUBST_DATA_COPY=${Popt}_DATA_COPY\
              -DSUBST_IN_SYN=${Popt}_IN_SYN )

 setenv CVSROOT $Project

#> make the config file

 set Cfile = ${CFG}.bld
 set quote = '"'

 echo                                                              > $Cfile
 echo "model       $MODEL;"                                       >> $Cfile
 echo                                                             >> $Cfile
 echo "FPP         $FP;"                                          >> $Cfile
 echo                                                             >> $Cfile
 set text = "$quote$CPP_FLAGS $CV $CVD $PAR $STX1 $STX2 $STX3$quote;"
 echo "cpp_flags   $text"                                         >> $Cfile
 echo                                                             >> $Cfile
 echo "f_compiler  $FC;"                                          >> $Cfile
 echo                                                             >> $Cfile
 echo "f_flags     $quote$F_FLAGS$quote;"                         >> $Cfile
 echo                                                             >> $Cfile
 echo "c_flags     $quote$C_FLAGS$quote;"                         >> $Cfile
 echo                                                             >> $Cfile
 echo "link_flags  $quote$LINK_FLAGS$quote;"                      >> $Cfile
 echo                                                             >> $Cfile
 echo "libraries   $quote$LIBS$quote;"                            >> $Cfile
 echo                                                             >> $Cfile
 echo "global      $Opt;"                                         >> $Cfile
 echo                                                             >> $Cfile

 set text="// layers, mechanism and tracer:"
 echo "$text ${Layers}, ${Mechanism}, ${Tracer}"                  >> $Cfile
 echo "// project archive: ${Project}"                           >> $Cfile
```

```
echo                                                                  >> $Cfile

echo "include SUBST_PE_COMM    $ICL_PAR/PE_COMM.EXT;"                 >> $Cfile
echo "include SUBST_CONST      $ICL_CONST/CONST.EXT;"                 >> $Cfile
echo "include SUBST_FILES_ID   $ICL_FILES/FILES_CTM.EXT;"             >> $Cfile
echo "include SUBST_EMPR_VD    $ICL_EMCTL/EMISPRM.vdif.EXT;"          >> $Cfile
echo "include SUBST_EMPR_CH    $ICL_EMCTL/EMISPRM.chem.EXT;"          >> $Cfile
echo "include SUBST_IOPARMS    $ICL_IOAPI/PARMS3.EXT;"                >> $Cfile
echo "include SUBST_IOFDESC    $ICL_IOAPI/FDESC3.EXT;"                >> $Cfile
echo "include SUBST_IODECL     $ICL_IOAPI/IODECL3.EXT;"               >> $Cfile
echo "include SUBST_COORD_ID   $ICL_VCRD/COORD_63X28.EXT;"   >> $Cfile
echo "include SUBST_VGRD_ID    $ICL_VCRD/VGRD_${Layers}.EXT;"         >> $Cfile

echo "include SUBST_RXCMMN     $ICL_MECH/RXCM.EXT;"                   >> $Cfile
echo "include SUBST_RXDATA     $ICL_MECH/RXDT.EXT;"                   >> $Cfile
echo "include SUBST_ARRAYMAP   $ICL_MECH/ARRAYMAP.EXT;"               >> $Cfile
echo "include SUBST_PARAMETER  $ICL_MECH/PARAMETER.EXT;"              >> $Cfile
echo "include SUBST_AQ_PARAMS  $ICL_MECH/AQ_PARAMS.EXT;"              >> $Cfile
echo "include SUBST_HETERPAR   $ICL_MECH/HETERPAR.EXT;"               >> $Cfile
echo "include SUBST_GC_SPC     $ICL_MECH/GC_SPC.EXT;"                 >> $Cfile
echo "include SUBST_GC_EMIS    $ICL_MECH/GC_EMIS.EXT;"                >> $Cfile
echo "include SUBST_GC_ICBC    $ICL_MECH/GC_ICBC.EXT;"                >> $Cfile
echo "include SUBST_GC_DIFF    $ICL_MECH/GC_DIFF.EXT;"                >> $Cfile
echo "include SUBST_GC_DDEP    $ICL_MECH/GC_DDEP.EXT;"                >> $Cfile
echo "include SUBST_GC_DEPV    $ICL_MECH/GC_DEPV.EXT;"                >> $Cfile
echo "include SUBST_GC_ADV     $ICL_MECH/GC_ADV.EXT;"                 >> $Cfile
echo "include SUBST_GC_CONC    $ICL_MECH/GC_CONC.EXT;"                >> $Cfile
echo "include SUBST_GC_G2AE    $ICL_MECH/GC_G2AE.EXT;"                >> $Cfile
echo "include SUBST_GC_G2AQ    $ICL_MECH/GC_G2AQ.EXT;"                >> $Cfile
echo "include SUBST_GC_SCAV    $ICL_MECH/GC_SCAV.EXT;"                >> $Cfile
echo "include SUBST_GC_WDEP    $ICL_MECH/GC_WDEP.EXT;"                >> $Cfile
echo "include SUBST_AE_SPC     $ICL_MECH/AE_SPC.EXT;"                 >> $Cfile
echo "include SUBST_AE_EMIS    $ICL_MECH/AE_EMIS.EXT;"                >> $Cfile
echo "include SUBST_AE_ICBC    $ICL_MECH/AE_ICBC.EXT;"                >> $Cfile
echo "include SUBST_AE_DIFF    $ICL_MECH/AE_DIFF.EXT;"                >> $Cfile
echo "include SUBST_AE_DDEP    $ICL_MECH/AE_DDEP.EXT;"                >> $Cfile
echo "include SUBST_AE_DEPV    $ICL_MECH/AE_DEPV.EXT;"                >> $Cfile
echo "include SUBST_AE_ADV     $ICL_MECH/AE_ADV.EXT;"                 >> $Cfile
echo "include SUBST_AE_CONC    $ICL_MECH/AE_CONC.EXT;"                >> $Cfile
echo "include SUBST_AE_A2AQ    $ICL_MECH/AE_A2AQ.EXT;"                >> $Cfile
echo "include SUBST_AE_SCAV    $ICL_MECH/AE_SCAV.EXT;"                >> $Cfile
echo "include SUBST_AE_WDEP    $ICL_MECH/AE_WDEP.EXT;"                >> $Cfile
echo "include SUBST_NR_SPC     $ICL_MECH/NR_SPC.EXT;"                 >> $Cfile
echo "include SUBST_NR_EMIS    $ICL_MECH/NR_EMIS.EXT;"                >> $Cfile
echo "include SUBST_NR_ICBC    $ICL_MECH/NR_ICBC.EXT;"                >> $Cfile
echo "include SUBST_NR_DIFF    $ICL_MECH/NR_DIFF.EXT;"                >> $Cfile
echo "include SUBST_NR_DDEP    $ICL_MECH/NR_DDEP.EXT;"                >> $Cfile
echo "include SUBST_NR_DEPV    $ICL_MECH/NR_DEPV.EXT;"                >> $Cfile
echo "include SUBST_NR_ADV     $ICL_MECH/NR_ADV.EXT;"                 >> $Cfile
echo "include SUBST_NR_N2AE    $ICL_MECH/NR_N2AE.EXT;"                >> $Cfile
echo "include SUBST_NR_N2AQ    $ICL_MECH/NR_N2AQ.EXT;"                >> $Cfile
echo "include SUBST_NR_SCAV    $ICL_MECH/NR_SCAV.EXT;"                >> $Cfile
echo "include SUBST_NR_WDEP    $ICL_MECH/NR_WDEP.EXT;"                >> $Cfile
echo "include SUBST_TR_SPC     $ICL_TRAC/TR_SPC.EXT;"                 >> $Cfile
echo "include SUBST_TR_EMIS    $ICL_TRAC/TR_EMIS.EXT;"                >> $Cfile
echo "include SUBST_TR_ICBC    $ICL_TRAC/TR_ICBC.EXT;"                >> $Cfile
echo "include SUBST_TR_DIFF    $ICL_TRAC/TR_DIFF.EXT;"                >> $Cfile
echo "include SUBST_TR_DDEP    $ICL_TRAC/TR_DDEP.EXT;"                >> $Cfile
echo "include SUBST_TR_DEPV    $ICL_TRAC/TR_DEPV.EXT;"                >> $Cfile
echo "include SUBST_TR_ADV     $ICL_TRAC/TR_ADV.EXT;"                 >> $Cfile
echo "include SUBST_TR_T2AQ    $ICL_TRAC/TR_T2AQ.EXT;"                >> $Cfile
echo "include SUBST_TR_SCAV    $ICL_TRAC/TR_SCAV.EXT;"                >> $Cfile
echo "include SUBST_TR_WDEP    $ICL_TRAC/TR_WDEP.EXT;"                >> $Cfile
echo                                                                  >> $Cfile
```

```
 set text = "// Process Analysis / Integrated Reaction Rates processing"
 echo $text                                                          >> $Cfile
#echo "include SUBST_PACTL_ID    $ICL_PA/PA_CTL_no_irr.EXT;"       >> $Cfile
 echo "include SUBST_PACTL_ID    $ICL_PA/PA_CTL.EXT;"              >> $Cfile
 echo "include SUBST_PACMN_ID    $ICL_PA/PA_CMN.EXT;"              >> $Cfile
 echo "include SUBST_PADAT_ID    $ICL_PA/PA_DAT.EXT;"              >> $Cfile
 echo                                                              >> $Cfile

 echo "$Str1"                                                      >> $Cfile
 echo "$Str2"                                                      >> $Cfile
 echo                                                              >> $Cfile

 echo "$ModDriver"                                                 >> $Cfile
 echo                                                              >> $Cfile

 echo "$ModPar"                                                    >> $Cfile
 echo                                                              >> $Cfile

 echo "$ModInit"                                                   >> $Cfile
 echo                                                              >> $Cfile

 set text = "denrate and adjcon_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModAdjc"                                                   >> $Cfile
 echo                                                              >> $Cfile

 echo "$ModCpl"                                                    >> $Cfile
 echo                                                              >> $Cfile

 set text = "hbot, hppm and hadv_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModHadv"                                                   >> $Cfile
 echo                                                              >> $Cfile

 set text = "vbot, vppm and vadv_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModVadv"                                                   >> $Cfile
 echo                                                              >> $Cfile

 set text = "unif, multi_scale and hdiff_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModHdiff"                                                  >> $Cfile
 echo                                                              >> $Cfile

 set text = "eddy and vdiff_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModVdiff"                                                  >> $Cfile
 echo                                                              >> $Cfile

 set text = "phot and phot_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModPhot"                                                   >> $Cfile
 echo                                                              >> $Cfile

 set text = "ping_qssa, ping_smvgear, ping_mebi_cb4_1 and ping_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModPing"                                                   >> $Cfile
 echo                                                              >> $Cfile

 set text = "qssa, smvgear, mebi_cb4_1 and chem_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModChem"                                                   >> $Cfile
 echo                                                              >> $Cfile
```

```
 set text = "aero1, aero2 and aero_noop"
 echo "// options are" $text                                          >> $Cfile
 echo "$ModAero"                                                     >> $Cfile
 echo                                                                >> $Cfile

 set text = "aero_depv1, aero_depv2 and aero_depv_noop"
 echo "// options are" $text                                          >> $Cfile
 echo "$ModAdepv"                                                    >> $Cfile
 echo                                                                >> $Cfile

 set text = "cloud_radm and cloud_noop"
 echo "// options are" $text                                          >> $Cfile
 echo "$ModCloud"                                                    >> $Cfile
 echo                                                                >> $Cfile

 set text = "pa and pa_noop, which requires the"
 echo "// options are" $text "replacement of the three"              >> $Cfile
 set text = "// global include files with their pa_noop counterparts"
 echo $text                                                          >> $Cfile
 echo "$ModPa"                                                       >> $Cfile
 echo                                                                >> $Cfile

 echo "$ModUtil"                                                     >> $Cfile
 echo                                                                >> $Cfile

 echo "$Modaeaqsolver"                                               >> $Cfile
 echo                                                                >> $Cfile

 if ( $?ModMisc ) then
    echo "$ModMisc"                                                  >> $Cfile
    echo                                                            >> $Cfile
    endif

#> make the makefile or the model executable

 if ( $?MakeOpt ) then
    $Blder -make $Cfile   # $Cfile = ${CFG}.bld
    else
    set NoMake
    $Blder $Cfile
    endif
 if ( $status != 0 ) then
    echo "   *** failure in $Blder ***"
    exit 1
    endif
 if ( -e "$Base/${CFG}" ) then
    echo "   >>> previous ${CFG} exists, re-naming to ${CFG}.old <<<"
    unalias mv
    mv $Base/${CFG} $Base/${CFG}.old
    endif
 cp ${CFG}.bld $Base/${CFG}
 if ( ( $Opt != no_compile ) && \
      ( $Opt != no_link    ) && \
      ( $Opt != parse_only ) && \
      ( $Opt != show_only  ) && \
        $?NoMake ) then
    mv $MODEL $Base
    endif

 exit
```

A-2 Script used to compile Models-3/CMAQ with aero_MADRID2.

```
#!/bin/csh -f

setenv REPOSITORY /usr2/cp194/CMAQ-MADRID/code/oct2004
setenv M3MODEL $REPOSITORY/models    # code archive
setenv M3LIB   $REPOSITORY/lib       # libraries

set BLD_OS = Linux
set echo

#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#
 set Project = $M3MODEL/CCTM
 set GlobInc = $M3MODEL/include/release

 set Base = $cwd

 set APPL  = cacm_ros3_aeMADRID2_2sec
 set CFG   = cfg.$APPL
 set MODEL = CCTM_$APPL

#> user choices: m3bld command
#set Opt = compile_all  # force compile, even if object files are current
#set Opt = clean_up     # remove all source files upon successful completion
#set Opt = no_compile   # do everything except compile
#set Opt = no_link      # do everything except link
#set Opt = one_step     # compile and link in one step
#set Opt = parse_only   # checks config file syntax

#set Opt = show_only    # show requested commands but doesn't execute them
 set Opt = verbose      # show requested commands as they are executed
 set MakeOpt            # builds a Makefile to make the model

#> user choices:  single or multiple processors
#set ParOpt             # multiple PE's

#> user choices: various modules

 set Revision = release       # release = latest CVS revision
#> NOTE: m3bld will try to compile with existing code; it will not retrieve
#>       new (different release) code. So if your "BLD" directory contains
#>       code from a release different than the one you have specified above,
#>       m3bld will tell you, but will merrily compile the original code.
#>       The workaround is to remove your "BLD" directory and start fresh.

#set ModDriver = ( module ctm               $Revision; )
#set ModDriver = ( module ctm_MADRID1       $Revision; )
 set ModDriver = ( module ctm_MADRID2       $Revision; )

 if ( $?ParOpt ) then
    set ModPar = ( module par               $Revision; )
    else
    set ModPar = ( module par_noop          $Revision; )
    endif

#set ModInit   = ( module init              $Revision; )
 set ModInit   = ( module init_MADRID       $Revision; )

#set ModAdjc   = ( module adjcon_noop       $Revision; )
 set ModAdjc   = ( module denrate           $Revision; )

 set ModCpl    = ( module gencoor           $Revision; )

#set ModHadv   = ( module hadv_noop         $Revision; )
```

```
#set ModHadv   = ( module hbot               $Revision; )
#set ModHadv   = ( module hppm               $Revision; )
 set ModHadv   = ( module hppm_MADRID        $Revision; )

#set ModVadv   = ( module vadv_noop          $Revision; )
#set ModVadv   = ( module vbot               $Revision; )
 set ModVadv   = ( module vppm               $Revision; )

#set ModHdiff  = ( module hdiff_noop         $Revision; )
#set ModHdiff  = ( module unif               $Revision; )
 set ModHdiff  = ( module multiscale         $Revision; )

#set ModVdiff  = ( module vdiff_noop         $Revision; )
#set ModVdiff  = ( module eddy               $Revision; )
 set ModVdiff  = ( module eddy_MADRID        $Revision; )
#set ModVdiff  = ( module acm                $Revision; )

#set ModPhot   = ( module phot_noop          $Revision; )
 set ModPhot   = ( module phot               $Revision; )

 set ModPing   = ( module ping_noop          $Revision; )
#set ModPing   = ( module ping_qssa          $Revision; )
#set ModPing   = ( module ping_smvgear       $Revision; )
#set ModPing   = ( module ping_mebi_cb4      $Revision; )

#set ModChem   = ( module chem_noop          $Revision; )
#set ModChem   = ( module smvgear            $Revision; )
#set ModChem   = ( module ebi_cb4            $Revision; )
#set ModChem   = ( module ebi_cb4_MADRID1a    $Revision; )
#set ModChem   = ( module mebi_radm2_cis4    $Revision; )
#set ModChem   = ( module mebi_radm2_cis4_MADRID1    $Revision; )
#set ModChem   = ( module ros3_MADRID1       $Revision; )
 set ModChem   = ( module ros3_cacm_MADRID2      $Revision; )

#set ModAero   = ( module aero_noop          $Revision; )
#set ModAero   = ( module aero3              $Revision; )
#set ModAero   = ( module aero_MADRID1a      $Revision; )
 set ModAero   = ( module aero_MADRID2       $Revision; )
#set ModAero   = ( module aero_MADRID1       $Revision; )
#set ModAero   = ( module aero_MADRID1       $Revision; )

#set ModAdepv  = ( module aero_depv_noop     $Revision; )
#set ModAdepv  = ( module aero_depv1         $Revision; )
#set ModAdepv  = ( module aero_depv2         $Revision; )
 set ModAdepv  = ( module aero_depv_MADRID   $Revision; )

 set ModCloud  = ( module cloud_noop         $Revision; )
#set ModCloud  = ( module cloud_radm         $Revision; )
#set ModCloud  = ( module cloud_RADM_aero_MADRID1  $Revision; )
#set ModCloud  = ( module cloud_RADM_aero_MADRID2  $Revision; )


if ($ModCloud[2] == 'cloud_noop')then
   set CVD =
else
   set CVD = -Dcld_proc
endif

#set ModPa     = ( module pa                 $Revision; )
 set ModPa     = ( module pa_MADRID          $Revision; )

 set ModUtil   = ( module util               $Revision; )

 set Modaeaqsolver = ( module ae_aq_solver_MADRID    $Revision; )
```

```
#set ModMisc   = ( misc; \
#                  $Base/file1.F \
#                  $Base/file2.F \
#                  $Base/file3.F )

#> user choices: emissions processing in chem or vdiff (default) ...
#set Cemis

#> user choices: vertical layers and mechanism
#set Layers    = 21
 set Layers    = 15
#set Mechanism = radm2_ci4_aqRADM_aeMADRID1_2sec
#set Mechanism = cb4_aeMADRID1_2sec
#set Mechanism = cb4_aeMADRID1a_2sec
#set Mechanism = cb4_aqRADM_aeMADRID1_2sec
#set Mechanism = cb4_aeMADRID1a_2sec
 set Mechanism = CACM_aeMADRID2_2sec
#set Mechanism = cb4_aqRADM_aeMADRID1a_2sec
#set Mechanism = saprc99_ae3_aq
 set Tracer    = trac0               # default: no tracer species

#> user choices: set process analysis linkages
 set PABase    = $GlobInc
 set PAOpt     = pa_noop
#set PABase    = /project/cmaq/yoj/saprc
#set PAOpt     =

#> other user choices set below are:
#>    name of the "BLD" directory
#>    compiler/link flags
#>    library paths
#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#

 set Bld = $Base/BLD_${APPL}
#unset echo
 if ( ! -e "$Bld" ) then
    mkdir $Bld
    else
    if ( ! -d "$Bld" ) then
       echo "   *** target exists, but not a directory ***"
       exit 1
       endif
    endif
#set echo
 cd $Bld

#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#

 set STENEX = ${REPOSITORY}/lib/stenex_noop/${BLD_OS}
 set MODLOC = ${Base}/MOD_DIR
 if ( ! -d "$MODLOC" ) mkdir -p $MODLOC

 if ( $?ParOpt ) then
#   set Mpich = /usr/local/mpich-1.2.4       # compiled for ch_shmem
    set Mpich = /share/linux/bin/mpich-ch_p4 # compiled for ch_p4
    set seL = sef90
    cp -p ${STENEX}/se_*.mod $MODLOC
    set LIB2 = "-L${M3LIB}/pario/${BLD_OS} -lpario"
    set LIB3 = "-L${M3LIB}/dynmem/${BLD_OS} -ldynmem"
    set LIB4 = "-L${Mpich}/lib -lmpich"
    set Str1 = (// Parallel / Include message passing definitions)
    set Str2 = (include SUBST_MPICH ${Mpich}/include/mpif.h;)
    else
```

```
    set Mpich =
    set seL = sef90_noop
    cp -p ${STENEX}/noop_*.mod $MODLOC
    set LIB2 =
    set LIB3 =
    set LIB4 =
    set Str1 =
    set Str2 =
    endif

 set LIB1 = "-L${STENEX} -l${seL}"
 set LIB2 = "-L${M3LIB}/ioapi/ioapi_22/Linux2_x86pg -lioapi"
 set LIB3 = "-L${M3LIB}/netCDF/${BLD_OS} -lnetcdf"
 set LIBS = "$LIB1 $LIB2 $LIB3"

 set FC = /usr/pgi/linux86/5.2/bin/pgf90
 set FP = $FC

 set FSTD       = "-Mfixed -Mextend"
 set F_FLAGS    = "${FSTD} -fast -module ${MODLOC} -I."
 set CPP_FLAGS  = ""
 set C_FLAGS    = "-Xc -v -g"
 set LINK_FLAGS = "-Bstatic"

 set Blder = ${M3LIB}/build/$BLD_OS/m3bld

 set ICL_PAR   = $GlobInc
 set ICL_CONST = $GlobInc
 set ICL_FILES = $GlobInc
 set ICL_EMCTL = $GlobInc
 set ICL_IOAPI = ${M3LIB}/m3io/Linux
#set ICL_VCRD  = $GlobInc/vcoord
 set ICL_VCRD  = $Base/icl
 set ICL_MECH  = $GlobInc/$Mechanism
 set ICL_TRAC  = $GlobInc/$Tracer
 set ICL_PA    = $PABase/$PAOpt

 if ( $?Cemis ) then
    set CV = -Demis_chem
    else
    set CV =
    endif


#> NOTE: To run parallel in a Scyld Beowulf cluster, e.g., remove the
#>       "-Dcluster\" below.

 if ( $?ParOpt ) then   # split to avoid line > 256 char
    set PAR = ( -Dparallel\
                -Dcluster\
                -DINTERPB=PINTERPB\
                -DM3ERR=PM3ERR\
                -DM3EXIT=PM3EXIT\
                -DM3WARN=PM3WARN\
                -DSHUT3=PSHUT3\
                -DWRITE3=PWRITE3 )

    set Popt = SE
    else
    echo "   Not Parallel; set Serial (no-op) flags"
    set PAR = "-DINTERPB=INTERP3"
    set Popt = NOOP
    endif
```

```
 set STX1 = ( -DSUBST_MODULES=${Popt}_MODULES\
              -DSUBST_BARRIER=${Popt}_BARRIER )
 set STX2 = ( -DSUBST_GLOBAL_MAX=${Popt}_GLOBAL_MAX\
              -DSUBST_GLOBAL_MIN=${Popt}_GLOBAL_MIN\
              -DSUBST_GLOBAL_MIN_DATA=${Popt}_GLOBAL_MIN_DATA\
              -DSUBST_GLOBAL_TO_LOCAL_COORD=${Popt}_GLOBAL_TO_LOCAL_COORD\
              -DSUBST_GLOBAL_SUM=${Popt}_GLOBAL_SUM\
              -DSUBST_GLOBAL_LOGICAL=${Popt}_GLOBAL_LOGICAL\
              -DSUBST_LOOP_INDEX=${Popt}_LOOP_INDEX\
              -DSUBST_SUBGRID_INDEX=${Popt}_SUBGRID_INDEX )
 set STX3 = ( -DSUBST_HI_LO_BND_PE=${Popt}_HI_LO_BND_PE\
              -DSUBST_SUM_CHK=${Popt}_SUM_CHK\
              -DSUBST_INIT_ARRAY=${Popt}_INIT_ARRAY\
              -DSUBST_COMM=${Popt}_COMM\
              -DSUBST_MY_REGION=${Popt}_MY_REGION\
              -DSUBST_SLICE=${Popt}_SLICE\
              -DSUBST_GATHER=${Popt}_GATHER\
              -DSUBST_DATA_COPY=${Popt}_DATA_COPY\
              -DSUBST_IN_SYN=${Popt}_IN_SYN )

 setenv CVSROOT $Project

#> make the config file

 set Cfile = ${CFG}.bld
 set quote = '"'

 echo                                                              > $Cfile
 echo "model       $MODEL;"                                       >> $Cfile
 echo                                                             >> $Cfile
 echo "FPP         $FP;"                                          >> $Cfile
 echo                                                             >> $Cfile
 set text = "$quote$CPP_FLAGS $CV $CVD $PAR $STX1 $STX2 $STX3$quote;"
 echo "cpp_flags   $text"                                         >> $Cfile
 echo                                                             >> $Cfile
 echo "f_compiler  $FC;"                                          >> $Cfile
 echo                                                             >> $Cfile
 echo "f_flags     $quote$F_FLAGS$quote;"                         >> $Cfile
 echo                                                             >> $Cfile
 echo "c_flags     $quote$C_FLAGS$quote;"                         >> $Cfile
 echo                                                             >> $Cfile
 echo "link_flags  $quote$LINK_FLAGS$quote;"                      >> $Cfile
 echo                                                             >> $Cfile
 echo "libraries   $quote$LIBS$quote;"                            >> $Cfile
 echo                                                             >> $Cfile
 echo "global      $Opt;"                                         >> $Cfile
 echo                                                             >> $Cfile

 set text="// layers, mechanism and tracer:"
 echo "$text ${Layers}, ${Mechanism}, ${Tracer}"                  >> $Cfile
 echo "// project archive: ${Project}"                           >> $Cfile
 echo                                                             >> $Cfile

 echo "include SUBST_PE_COMM    $ICL_PAR/PE_COMM.EXT;"            >> $Cfile
 echo "include SUBST_CONST      $ICL_CONST/CONST.EXT;"            >> $Cfile
 echo "include SUBST_FILES_ID   $ICL_FILES/FILES_CTM.EXT;"        >> $Cfile
 echo "include SUBST_EMPR_VD    $ICL_EMCTL/EMISPRM.vdif.EXT;"     >> $Cfile
 echo "include SUBST_EMPR_CH    $ICL_EMCTL/EMISPRM.chem.EXT;"     >> $Cfile
 echo "include SUBST_IOPARMS    $ICL_IOAPI/PARMS3.EXT;"           >> $Cfile
 echo "include SUBST_IOFDESC    $ICL_IOAPI/FDESC3.EXT;"           >> $Cfile
 echo "include SUBST_IODECL     $ICL_IOAPI/IODECL3.EXT;"          >> $Cfile
 echo "include SUBST_COORD_ID   $ICL_VCRD/COORD_63X28.EXT;"       >> $Cfile
 echo "include SUBST_VGRD_ID    $ICL_VCRD/VGRD_15.EXT;"           >> $Cfile
```

```
 echo "include SUBST_RXCMMN     $ICL_MECH/RXCM.EXT;"              >> $Cfile
 echo "include SUBST_RXDATA     $ICL_MECH/RXDT.EXT;"              >> $Cfile
 echo "include SUBST_ARRAYMAP   $ICL_MECH/ARRAYMAP.EXT;"          >> $Cfile
 echo "include SUBST_PARAMETER  $ICL_MECH/PARAMETER.EXT;"         >> $Cfile
 echo "include SUBST_AQ_PARAMS  $ICL_MECH/AQ_PARAMS.EXT;"        >> $Cfile
 echo "include SUBST_HETERPAR   $ICL_MECH/HETERPAR.EXT;"         >> $Cfile
 echo "include SUBST_GC_SPC     $ICL_MECH/GC_SPC.EXT;"           >> $Cfile
 echo "include SUBST_GC_EMIS    $ICL_MECH/GC_EMIS.EXT;"          >> $Cfile
 echo "include SUBST_GC_ICBC    $ICL_MECH/GC_ICBC.EXT;"          >> $Cfile
 echo "include SUBST_GC_DIFF    $ICL_MECH/GC_DIFF.EXT;"          >> $Cfile
 echo "include SUBST_GC_DDEP    $ICL_MECH/GC_DDEP.EXT;"          >> $Cfile
 echo "include SUBST_GC_DEPV    $ICL_MECH/GC_DEPV.EXT;"          >> $Cfile
 echo "include SUBST_GC_ADV     $ICL_MECH/GC_ADV.EXT;"           >> $Cfile
 echo "include SUBST_GC_CONC    $ICL_MECH/GC_CONC.EXT;"          >> $Cfile
 echo "include SUBST_GC_G2AE    $ICL_MECH/GC_G2AE.EXT;"          >> $Cfile
 echo "include SUBST_GC_G2AQ    $ICL_MECH/GC_G2AQ.EXT;"          >> $Cfile
 echo "include SUBST_GC_SCAV    $ICL_MECH/GC_SCAV.EXT;"          >> $Cfile
 echo "include SUBST_GC_WDEP    $ICL_MECH/GC_WDEP.EXT;"          >> $Cfile
 echo "include SUBST_AE_SPC     $ICL_MECH/AE_SPC.EXT;"           >> $Cfile
 echo "include SUBST_AE_EMIS    $ICL_MECH/AE_EMIS.EXT;"          >> $Cfile
 echo "include SUBST_AE_ICBC    $ICL_MECH/AE_ICBC.EXT;"          >> $Cfile
 echo "include SUBST_AE_DIFF    $ICL_MECH/AE_DIFF.EXT;"          >> $Cfile
 echo "include SUBST_AE_DDEP    $ICL_MECH/AE_DDEP.EXT;"          >> $Cfile
 echo "include SUBST_AE_DEPV    $ICL_MECH/AE_DEPV.EXT;"          >> $Cfile
 echo "include SUBST_AE_ADV     $ICL_MECH/AE_ADV.EXT;"           >> $Cfile
 echo "include SUBST_AE_CONC    $ICL_MECH/AE_CONC.EXT;"          >> $Cfile
 echo "include SUBST_AE_A2AQ    $ICL_MECH/AE_A2AQ.EXT;"          >> $Cfile
 echo "include SUBST_AE_SCAV    $ICL_MECH/AE_SCAV.EXT;"          >> $Cfile
 echo "include SUBST_AE_WDEP    $ICL_MECH/AE_WDEP.EXT;"          >> $Cfile
 echo "include SUBST_NR_SPC     $ICL_MECH/NR_SPC.EXT;"           >> $Cfile
 echo "include SUBST_NR_EMIS    $ICL_MECH/NR_EMIS.EXT;"          >> $Cfile
 echo "include SUBST_NR_ICBC    $ICL_MECH/NR_ICBC.EXT;"          >> $Cfile
 echo "include SUBST_NR_DIFF    $ICL_MECH/NR_DIFF.EXT;"          >> $Cfile
 echo "include SUBST_NR_DDEP    $ICL_MECH/NR_DDEP.EXT;"          >> $Cfile
 echo "include SUBST_NR_DEPV    $ICL_MECH/NR_DEPV.EXT;"          >> $Cfile
 echo "include SUBST_NR_ADV     $ICL_MECH/NR_ADV.EXT;"           >> $Cfile
 echo "include SUBST_NR_N2AE    $ICL_MECH/NR_N2AE.EXT;"          >> $Cfile
 echo "include SUBST_NR_N2AQ    $ICL_MECH/NR_N2AQ.EXT;"          >> $Cfile
 echo "include SUBST_NR_SCAV    $ICL_MECH/NR_SCAV.EXT;"          >> $Cfile
 echo "include SUBST_NR_WDEP    $ICL_MECH/NR_WDEP.EXT;"          >> $Cfile
 echo "include SUBST_TR_SPC     $ICL_TRAC/TR_SPC.EXT;"           >> $Cfile
 echo "include SUBST_TR_EMIS    $ICL_TRAC/TR_EMIS.EXT;"          >> $Cfile
 echo "include SUBST_TR_ICBC    $ICL_TRAC/TR_ICBC.EXT;"          >> $Cfile
 echo "include SUBST_TR_DIFF    $ICL_TRAC/TR_DIFF.EXT;"          >> $Cfile
 echo "include SUBST_TR_DDEP    $ICL_TRAC/TR_DDEP.EXT;"          >> $Cfile
 echo "include SUBST_TR_DEPV    $ICL_TRAC/TR_DEPV.EXT;"          >> $Cfile
 echo "include SUBST_TR_ADV     $ICL_TRAC/TR_ADV.EXT;"           >> $Cfile
 echo "include SUBST_TR_T2AQ    $ICL_TRAC/TR_T2AQ.EXT;"          >> $Cfile
 echo "include SUBST_TR_SCAV    $ICL_TRAC/TR_SCAV.EXT;"          >> $Cfile
 echo "include SUBST_TR_WDEP    $ICL_TRAC/TR_WDEP.EXT;"          >> $Cfile
 echo                                                           >> $Cfile

 set text = "// Process Analysis / Integrated Reaction Rates processing"
 echo $text                                                     >> $Cfile
#echo "include SUBST_PACTL_ID    $ICL_PA/PA_CTL_no_irr.EXT;"   >> $Cfile
 echo "include SUBST_PACTL_ID    $ICL_PA/PA_CTL.EXT;"           >> $Cfile
 echo "include SUBST_PACMN_ID    $ICL_PA/PA_CMN.EXT;"           >> $Cfile
 echo "include SUBST_PADAT_ID    $ICL_PA/PA_DAT.EXT;"           >> $Cfile
 echo                                                           >> $Cfile

 echo "$Str1"                                                   >> $Cfile
 echo "$Str2"                                                   >> $Cfile
 echo                                                           >> $Cfile
```

```
echo "$ModDriver"                                                    >> $Cfile
echo                                                                 >> $Cfile

echo "$ModPar"                                                       >> $Cfile
echo                                                                 >> $Cfile

echo "$ModInit"                                                      >> $Cfile
echo                                                                 >> $Cfile

set text = "denrate and adjcon_noop"
echo "// options are" $text                                         >> $Cfile
echo "$ModAdjc"                                                      >> $Cfile
echo                                                                 >> $Cfile

echo "$ModCpl"                                                       >> $Cfile
echo                                                                 >> $Cfile

set text = "hbot, hppm and hadv_noop"
echo "// options are" $text                                         >> $Cfile
echo "$ModHadv"                                                      >> $Cfile
echo                                                                 >> $Cfile

set text = "vbot, vppm and vadv_noop"
echo "// options are" $text                                         >> $Cfile
echo "$ModVadv"                                                      >> $Cfile
echo                                                                 >> $Cfile

set text = "unif, multi_scale and hdiff_noop"
echo "// options are" $text                                         >> $Cfile
echo "$ModHdiff"                                                     >> $Cfile
echo                                                                 >> $Cfile

set text = "eddy and vdiff_noop"
echo "// options are" $text                                         >> $Cfile
echo "$ModVdiff"                                                     >> $Cfile
echo                                                                 >> $Cfile

set text = "phot and phot_noop"
echo "// options are" $text                                         >> $Cfile
echo "$ModPhot"                                                      >> $Cfile
echo                                                                 >> $Cfile

set text = "ping_qssa, ping_smvgear, ping_mebi_cb4_1 and ping_noop"
echo "// options are" $text                                         >> $Cfile
echo "$ModPing"                                                      >> $Cfile
echo                                                                 >> $Cfile

set text = "qssa, smvgear, mebi_cb4_1 and chem_noop"
echo "// options are" $text                                         >> $Cfile
echo "$ModChem"                                                      >> $Cfile
echo                                                                 >> $Cfile

set text = "aero1, aero2 and aero_noop"
echo "// options are" $text                                         >> $Cfile
echo "$ModAero"                                                      >> $Cfile
echo                                                                 >> $Cfile

set text = "aero_depv1, aero_depv2 and aero_depv_noop"
echo "// options are" $text                                         >> $Cfile
echo "$ModAdepv"                                                     >> $Cfile
echo                                                                 >> $Cfile

set text = "cloud_radm and cloud_noop"
echo "// options are" $text                                         >> $Cfile
```

```
 echo "$ModCloud"                                                    >> $Cfile
 echo                                                                >> $Cfile

 set text = "pa and pa_noop, which requires the"
 echo "// options are" $text "replacement of the three"            >> $Cfile
 set text = "// global include files with their pa_noop counterparts"
 echo $text                                                          >> $Cfile
 echo "$ModPa"                                                       >> $Cfile
 echo                                                                >> $Cfile

 echo "$ModUtil"                                                     >> $Cfile
 echo                                                                >> $Cfile

 echo "$Modaeaqsolver"                                               >> $Cfile
 echo                                                                >> $Cfile

 if ( $?ModMisc ) then
    echo "$ModMisc"                                                  >> $Cfile
    echo                                                            >> $Cfile
    endif

#> make the makefile or the model executable

 if ( $?MakeOpt ) then
    $Blder -make $Cfile   # $Cfile = ${CFG}.bld
    else
    set NoMake
    $Blder $Cfile
    endif
 if ( $status != 0 ) then
    echo "   *** failure in $Blder ***"
    exit 1
    endif
 if ( -e "$Base/${CFG}" ) then
    echo "   >>> previous ${CFG} exists, re-naming to ${CFG}.old <<<"
    unalias mv
    mv $Base/${CFG} $Base/${CFG}.old
    endif
 cp ${CFG}.bld $Base/${CFG}
 if ( ( $Opt != no_compile ) && \
      ( $Opt != no_link    ) && \
      ( $Opt != parse_only ) && \
      ( $Opt != show_only  ) && \
        $?NoMake ) then
    mv $MODEL $Base
    endif

 exit
```

A-3 Script used to compile Models-3/CMAQ with mercury

```
#! /bin/csh -f

#> RCS file, release, date & time of last delta, author, state, [and locker]
#> $Header$

#> what(1) key, module and SID; SCCS file; date and time of last delta:
#> %W% %P% %G% %U%

### setenv M3MODEL /project/work/rep            # code archive
### setenv M3LIB   /project/air5/sjr/CMAS4.4/distr/rel/lib    # libraries

setenv REPOSITORY /usr2/cp194/CMAQ-MADRID/code/oct2004
setenv M3MODEL $REPOSITORY/models    # code archive
setenv M3LIB   $REPOSITORY/lib       # libraries

set BLD_OS = Linux
 set Base = $cwd

 if ( ! -e $M3MODEL || ! -e $M3LIB ) then
    echo "   $M3MODEL or $M3LIB directory not found"
    exit 1
    endif
 echo "   Model archive path: $M3MODEL"
 echo "         library path: $M3LIB"

 set echo

#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#
#> user choices: cvs archives
 set Project = $M3MODEL/CCTM
 set GlobInc = $M3MODEL/include/release

#> user choices: base directory
#set Base = /your_dir/you/working_dir
 set Base = $cwd

 set APPL  = cb4_aqRADM_hg_aeMD1a_2sec
 set CFG   = cfg.$APPL
 set MODEL = CCTM_$APPL

#> user choices: m3bld command
#set Opt = compile_all  # force compile, even if object files are current
#set Opt = clean_up     # remove all source files upon successful completion
#set Opt = no_compile   # do everything except compile
#set Opt = no_link      # do everything except link
#set Opt = one_step     # compile and link in one step
#set Opt = parse_only   # checks config file syntax
#set Opt = show_only    # show requested commands but doesn't execute them
 set Opt = verbose      # show requested commands as they are executed
 set MakeOpt            # builds a Makefile to make the model

#> user choices:  single or multiple processors
#set ParOpt             # multiple PE's

#> user choices: various modules

 set Revision = release       # release = latest CVS revision
#set Revision = '"STA3_2"'
#> NOTE: m3bld will try to compile with existing code; it will not retrieve
#>       new (different release) code. So if your "BLD" directory contains
#>       code from a release different than the one you have specified above,
#>       m3bld will tell you, but will merrily compile the original code.
```

```
#>       The workaround is to remove your "BLD" directory and start fresh.

#set ModDriver = ( module ctm                $Revision; )
 set ModDriver = ( module ctm_MADRID1_hg        $Revision; )

 if ( $?ParOpt ) then
    set ModPar = ( module par                $Revision; )
    else
    set ModPar = ( module par_noop           $Revision; )
    endif

#set ModInit   = ( module init               $Revision; )
 set ModInit   = ( module init_MADRID        $Revision; )

#set ModAdjc   = ( module adjcon_noop        $Revision; )
 set ModAdjc   = ( module denrate            $Revision; )

 set ModCpl    = ( module gencoor            $Revision; )

#set ModHadv   = ( module hadv_noop          $Revision; )
#set ModHadv   = ( module hppm               $Revision; )
 set ModHadv   = ( module hppm_MADRID        $Revision; )

#set ModVadv   = ( module vadv_noop          $Revision; )
 set ModVadv   = ( module vppm               $Revision; )

#set ModHdiff  = ( module hdiff_noop         $Revision; )
 set ModHdiff  = ( module multiscale         $Revision; )

#set ModVdiff  = ( module vdiff_noop         $Revision; )
#set ModVdiff  = ( module eddy               $Revision; )
#set ModVdiff  = ( module acm                $Revision; )
 set ModVdiff  = ( module eddy_MADRID        $Revision; )

#set ModPhot   = ( module phot_noop          $Revision; )
 set ModPhot   = ( module phot               $Revision; )

 set ModPing   = ( module ping_noop          $Revision; )
#set ModPing   = ( module ping_smvgear       $Revision; )

#set ModChem   = ( module chem_noop          $Revision; )
#set ModChem   = ( module smvgear            $Revision; )
#set ModChem   = ( module ros3               $Revision; )
#set ModChem   = ( module ebi_cb4            $Revision; )
 set ModChem   = ( module ebi_cb4_MADRID1a_hg $Revision; )
#set ModChem   = ( module ebi_saprc99        $Revision; )
#set ModChem   = ( module mebi_radm2_cis4    $Revision; )
#set ModChem   = ( module ros3_MADRID1       $Revision; )

#set ModAero   = ( module aero_noop          $Revision; )
#set ModAero   = ( module aero3              $Revision; )
 set ModAero   = ( module aero_MADRID1a      $Revision; )

#set ModAdepv  = ( module aero_depv_noop     $Revision; )
#set ModAdepv  = ( module aero_depv2         $Revision; )
 set ModAdepv  = ( module aero_depv_MADRID   $Revision; )

#set ModCloud  = ( module cloud_noop         $Revision; )
#set ModCloud  = ( module cloud_radm         $Revision; )
 set ModCloud  = ( module cloud_RADM_aero_MADRID1_hg  $Revision; )

if ($ModCloud[2] == 'cloud_noop')then
   set CVD =
else
```

```
   set CVD = -Dcld_proc
endif


#set ModPa     = ( module pa                 $Revision; )
 set ModPa     = ( module pa_MADRID          $Revision; )

 set ModUtil   = ( module util               $Revision; )

 set Modaeaqsolver = ( module ae_aq_solver_MADRID    $Revision; )

#set ModMisc   = ( misc; \
#                  $Base/file1.F \
#                  $Base/file2.F \
#                  $Base/file3.F )

#> user choices: emissions processing in chem or vdiff (default) ...
#set Cemis

#> user choices: vertical layers and mechanism
#set Layers    = 21
 set Layers    = 15
#set Mechanism = radm2_ci4_ae3_aq
#set Mechanism = cb4_ae3_aq
#set Mechanism = saprc99_ae3_aq
 set Mechanism = cb4_aqRADM_hg_aeMADRID1a_2sec
 set Tracer    = trac0               # default: no tracer species

#> user choices: set process analysis linkages
 set PABase    = $GlobInc
 set PAOpt     = pa_noop
#set PABase    = /project/cmaq/yoj/saprc
#set PAOpt     =

#> other user choices set below are:
#>    name of the "BLD" directory
#>    compiler/link flags
#>    library paths
#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#

 set Bld = $Base/BLD_${APPL}
#unset echo
 if ( ! -e "$Bld" ) then
    mkdir $Bld
    else
    if ( ! -d "$Bld" ) then
       echo "   *** target exists, but not a directory ***"
       exit 1
       endif
    endif
#set echo
 cd $Bld

#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#

 set STENEX = ${M3LIB}/stenex_noop/${BLD_OS}
 set MODLOC = ${Base}/MOD_DIR
 if ( ! -d "$MODLOC" ) mkdir -p $MODLOC

 if ( $?ParOpt ) then
#   set Mpich = /usr/local/mpich-1.2.4       # compiled for ch_shmem
    set Mpich = /share/linux/bin/mpich-ch_p4 # compiled for ch_p4
    set seL = se_snl
    cp -p ${STENEX}/*.mod $MODLOC
```

```
    set LIB2 = "-L${M3LIB}/pario/${BLD_OS} -lpario"
    set LIB3 =
    set LIB4 = "-L${Mpich}/lib -lmpich"
    set Str1 = (// Parallel / Include message passing definitions)
    set Str2 = (include SUBST_MPICH ${Mpich}/include/mpif.h;)
    else
    set Mpich =
    set seL = sef90_noop
    cp -p ${STENEX}/noop_*.mod $MODLOC
    set LIB2 =
    set LIB3 =
    set LIB4 =
    set Str1 =
    set Str2 =
    endif

 set LIB1 = "-L${STENEX} -l${seL}"
 set LIB5 = "-L${M3LIB}/ioapi/ioapi_22/Linux2_x86pg -lioapi"
 set LIB6 = "-L${M3LIB}/netCDF/${BLD_OS} -lnetcdf"
 set LIBS = "$LIB1 $LIB2 $LIB3 $LIB4 $LIB5 $LIB6"

 set FC = /usr/pgi/linux86/5.2/bin/pgf90
 set FP = $FC

 set FSTD       = "-Mfixed -Mextend"
 set F_FLAGS    = "${FSTD} -fast -module ${MODLOC} -I."
 set CPP_FLAGS  = ""
#set C_FLAGS    = "-v -O2 -I${Mpich}/include"
 set C_FLAGS    = "-Xc -v -g"
 set LINK_FLAGS = "-Bstatic"

#set Blder = ${M3LIB1}/build/$BLD_OS/m3bld
 set Blder = ${M3LIB}/build/$BLD_OS/m3bld

 set ICL_PAR   = $GlobInc
 set ICL_CONST = $GlobInc
 set ICL_FILES = $GlobInc
 set ICL_EMCTL = $GlobInc
#set ICL_IOAPI = ${M3LIB1}/m3io/Linux
 set ICL_IOAPI = ${M3LIB}/m3io/Linux
 set ICL_VCRD  = $Base/icl
 set ICL_MECH  = $GlobInc/$Mechanism
 set ICL_TRAC  = $GlobInc/$Tracer
 set ICL_PA    = $PABase/$PAOpt

 if ( $?Cemis ) then
    set CV = -Demis_chem
    else
    set CV =
    endif

#> NOTE: To run parallel in a Scyld Beowulf cluster, e.g., remove the
#>       "-Dcluster\" below.

 if ( $?ParOpt ) then   # split to avoid line > 256 char
    set PAR = ( -Dparallel\
                -Dcluster\
                -DINTERPB=PINTERPB\
                -DM3ERR=PM3ERR\
                -DM3EXIT=PM3EXIT\
                -DM3WARN=PM3WARN\
                -DSHUT3=PSHUT3\
                -DWRITE3=PWRITE3 )
```

```
    set Popt = SE
    else
    echo "   Not Parallel; set Serial (no-op) flags"
    set PAR = "-DINTERPB=INTERP3"
    set Popt = NOOP
    endif

 set STX1 = ( -DSUBST_MODULES=${Popt}_MODULES\
              -DSUBST_BARRIER=${Popt}_BARRIER )
 set STX2 = ( -DSUBST_GLOBAL_MAX=${Popt}_GLOBAL_MAX\
              -DSUBST_GLOBAL_MIN=${Popt}_GLOBAL_MIN\
              -DSUBST_GLOBAL_MIN_DATA=${Popt}_GLOBAL_MIN_DATA\
              -DSUBST_GLOBAL_TO_LOCAL_COORD=${Popt}_GLOBAL_TO_LOCAL_COORD\
              -DSUBST_GLOBAL_SUM=${Popt}_GLOBAL_SUM\
              -DSUBST_GLOBAL_LOGICAL=${Popt}_GLOBAL_LOGICAL\
              -DSUBST_LOOP_INDEX=${Popt}_LOOP_INDEX\
              -DSUBST_SUBGRID_INDEX=${Popt}_SUBGRID_INDEX )
 set STX3 = ( -DSUBST_HI_LO_BND_PE=${Popt}_HI_LO_BND_PE\
              -DSUBST_SUM_CHK=${Popt}_SUM_CHK\
              -DSUBST_INIT_ARRAY=${Popt}_INIT_ARRAY\
              -DSUBST_COMM=${Popt}_COMM\
              -DSUBST_MY_REGION=${Popt}_MY_REGION\
              -DSUBST_SLICE=${Popt}_SLICE\
              -DSUBST_GATHER=${Popt}_GATHER\
              -DSUBST_DATA_COPY=${Popt}_DATA_COPY\
              -DSUBST_IN_SYN=${Popt}_IN_SYN )

 setenv CVSROOT $Project

#> make the config file

 set Cfile = ${CFG}.bld
 set quote = '"'

 echo                                                          > $Cfile
 echo "model       $MODEL;"                                   >> $Cfile
 echo                                                         >> $Cfile
 echo "FPP         $FP;"                                      >> $Cfile
 echo                                                         >> $Cfile
 set text = "$quote$CPP_FLAGS $CV $CVD $PAR $STX1 $STX2 $STX3$quote;"
 echo "cpp_flags   $text"                                     >> $Cfile
 echo                                                         >> $Cfile
 echo "f_compiler  $FC;"                                      >> $Cfile
 echo                                                         >> $Cfile
 echo "f_flags     $quote$F_FLAGS$quote;"                     >> $Cfile
 echo                                                         >> $Cfile
 echo "c_flags     $quote$C_FLAGS$quote;"                     >> $Cfile
 echo                                                         >> $Cfile
 echo "link_flags  $quote$LINK_FLAGS$quote;"                  >> $Cfile
 echo                                                         >> $Cfile
 echo "libraries   $quote$LIBS$quote;"                        >> $Cfile
 echo                                                         >> $Cfile
 echo "global      $Opt;"                                     >> $Cfile
 echo                                                         >> $Cfile

 set text="// layers, mechanism and tracer:"
 echo "$text ${Layers}, ${Mechanism}, ${Tracer}"              >> $Cfile
 echo "// project archive: ${Project}"                        >> $Cfile
 echo                                                         >> $Cfile

 echo "include SUBST_PE_COMM    $ICL_PAR/PE_COMM.EXT;"        >> $Cfile
 echo "include SUBST_CONST      $ICL_CONST/CONST.EXT;"        >> $Cfile
 echo "include SUBST_FILES_ID   $ICL_FILES/FILES_CTM.Hg.EXT;"    >> $Cfile
 echo "include SUBST_EMPR_VD    $ICL_EMCTL/EMISPRM.vdif.EXT;" >> $Cfile
```

```
 echo "include SUBST_EMPR_CH    $ICL_EMCTL/EMISPRM.chem.EXT;"      >> $Cfile
 echo "include SUBST_IOPARMS    $ICL_IOAPI/PARMS3.EXT;"            >> $Cfile
 echo "include SUBST_IOFDESC    $ICL_IOAPI/FDESC3.EXT;"            >> $Cfile
 echo "include SUBST_IODECL     $ICL_IOAPI/IODECL3.EXT;"           >> $Cfile
 echo "include SUBST_COORD_ID   $ICL_VCRD/COORD.EXT;"   >> $Cfile
 echo "include SUBST_VGRD_ID    $ICL_VCRD/VGRD.EXT;"     >> $Cfile

 echo "include SUBST_RXCMMN     $ICL_MECH/RXCM.EXT;"               >> $Cfile
 echo "include SUBST_RXDATA     $ICL_MECH/RXDT.EXT;"               >> $Cfile
 echo "include SUBST_ARRAYMAP   $ICL_MECH/ARRAYMAP.EXT;"           >> $Cfile
 echo "include SUBST_PARAMETER  $ICL_MECH/PARAMETER.EXT;"          >> $Cfile
 echo "include SUBST_AQ_PARAMS  $ICL_MECH/AQ_PARAMS.EXT;"          >> $Cfile
 echo "include SUBST_HETERPAR   $ICL_MECH/HETERPAR.EXT;"           >> $Cfile
 echo "include SUBST_GC_SPC     $ICL_MECH/GC_SPC.EXT;"             >> $Cfile
 echo "include SUBST_GC_EMIS    $ICL_MECH/GC_EMIS.EXT;"            >> $Cfile
 echo "include SUBST_GC_ICBC    $ICL_MECH/GC_ICBC.EXT;"            >> $Cfile
 echo "include SUBST_GC_DIFF    $ICL_MECH/GC_DIFF.EXT;"            >> $Cfile
 echo "include SUBST_GC_DDEP    $ICL_MECH/GC_DDEP.EXT;"            >> $Cfile
 echo "include SUBST_GC_DEPV    $ICL_MECH/GC_DEPV.EXT;"            >> $Cfile
 echo "include SUBST_GC_ADV     $ICL_MECH/GC_ADV.EXT;"             >> $Cfile
 echo "include SUBST_GC_CONC    $ICL_MECH/GC_CONC.EXT;"            >> $Cfile
 echo "include SUBST_GC_G2AE    $ICL_MECH/GC_G2AE.EXT;"            >> $Cfile
 echo "include SUBST_GC_G2AQ    $ICL_MECH/GC_G2AQ.EXT;"            >> $Cfile
 echo "include SUBST_GC_SCAV    $ICL_MECH/GC_SCAV.EXT;"            >> $Cfile
 echo "include SUBST_GC_WDEP    $ICL_MECH/GC_WDEP.EXT;"            >> $Cfile
 echo "include SUBST_AE_SPC     $ICL_MECH/AE_SPC.EXT;"             >> $Cfile
 echo "include SUBST_AE_EMIS    $ICL_MECH/AE_EMIS.EXT;"            >> $Cfile
 echo "include SUBST_AE_ICBC    $ICL_MECH/AE_ICBC.EXT;"            >> $Cfile
 echo "include SUBST_AE_DIFF    $ICL_MECH/AE_DIFF.EXT;"            >> $Cfile
 echo "include SUBST_AE_DDEP    $ICL_MECH/AE_DDEP.EXT;"            >> $Cfile
 echo "include SUBST_AE_DEPV    $ICL_MECH/AE_DEPV.EXT;"            >> $Cfile
 echo "include SUBST_AE_ADV     $ICL_MECH/AE_ADV.EXT;"             >> $Cfile
 echo "include SUBST_AE_CONC    $ICL_MECH/AE_CONC.EXT;"            >> $Cfile
 echo "include SUBST_AE_A2AQ    $ICL_MECH/AE_A2AQ.EXT;"            >> $Cfile
 echo "include SUBST_AE_SCAV    $ICL_MECH/AE_SCAV.EXT;"            >> $Cfile
 echo "include SUBST_AE_WDEP    $ICL_MECH/AE_WDEP.EXT;"            >> $Cfile
 echo "include SUBST_NR_SPC     $ICL_MECH/NR_SPC.EXT;"             >> $Cfile
 echo "include SUBST_NR_EMIS    $ICL_MECH/NR_EMIS.EXT;"            >> $Cfile
 echo "include SUBST_NR_ICBC    $ICL_MECH/NR_ICBC.EXT;"            >> $Cfile
 echo "include SUBST_NR_DIFF    $ICL_MECH/NR_DIFF.EXT;"            >> $Cfile
 echo "include SUBST_NR_DDEP    $ICL_MECH/NR_DDEP.EXT;"            >> $Cfile
 echo "include SUBST_NR_DEPV    $ICL_MECH/NR_DEPV.EXT;"            >> $Cfile
 echo "include SUBST_NR_ADV     $ICL_MECH/NR_ADV.EXT;"             >> $Cfile
 echo "include SUBST_NR_N2AE    $ICL_MECH/NR_N2AE.EXT;"            >> $Cfile
 echo "include SUBST_NR_N2AQ    $ICL_MECH/NR_N2AQ.EXT;"            >> $Cfile
 echo "include SUBST_NR_SCAV    $ICL_MECH/NR_SCAV.EXT;"            >> $Cfile
 echo "include SUBST_NR_WDEP    $ICL_MECH/NR_WDEP.EXT;"            >> $Cfile
 echo "include SUBST_HGGASPARMS $Bld/HGGASPRM.EXT;"                >> $Cfile
 echo "include SUBST_HGAQSPARMS $Bld/HGAQSPRM.EXT;"                >> $Cfile
 echo "include SUBST_TR_SPC     $ICL_TRAC/TR_SPC.EXT;"             >> $Cfile
 echo "include SUBST_TR_EMIS    $ICL_TRAC/TR_EMIS.EXT;"            >> $Cfile
 echo "include SUBST_TR_ICBC    $ICL_TRAC/TR_ICBC.EXT;"            >> $Cfile
 echo "include SUBST_TR_DIFF    $ICL_TRAC/TR_DIFF.EXT;"            >> $Cfile
 echo "include SUBST_TR_DDEP    $ICL_TRAC/TR_DDEP.EXT;"            >> $Cfile
 echo "include SUBST_TR_DEPV    $ICL_TRAC/TR_DEPV.EXT;"            >> $Cfile
 echo "include SUBST_TR_ADV     $ICL_TRAC/TR_ADV.EXT;"             >> $Cfile
 echo "include SUBST_TR_T2AQ    $ICL_TRAC/TR_T2AQ.EXT;"            >> $Cfile
 echo "include SUBST_TR_SCAV    $ICL_TRAC/TR_SCAV.EXT;"            >> $Cfile
 echo "include SUBST_TR_WDEP    $ICL_TRAC/TR_WDEP.EXT;"            >> $Cfile
 echo                                                             >> $Cfile

 set text = "// Process Analysis / Integrated Reaction Rates processing"
 echo $text                                                       >> $Cfile
#echo "include SUBST_PACTL_ID    $ICL_PA/PA_CTL_no_irr.EXT;"      >> $Cfile
```

```
echo "include SUBST_PACTL_ID    $ICL_PA/PA_CTL.EXT;"              >> $Cfile
echo "include SUBST_PACMN_ID    $ICL_PA/PA_CMN.EXT;"              >> $Cfile
echo "include SUBST_PADAT_ID    $ICL_PA/PA_DAT.EXT;"              >> $Cfile
echo                                                              >> $Cfile

echo "$Str1"                                                      >> $Cfile
echo "$Str2"                                                      >> $Cfile
echo                                                              >> $Cfile

echo "$ModDriver"                                                 >> $Cfile
echo                                                              >> $Cfile

echo "$ModPar"                                                    >> $Cfile
echo                                                              >> $Cfile

echo "$ModInit"                                                   >> $Cfile
echo                                                              >> $Cfile

set text = "denrate and adjcon_noop"
echo "// options are" $text                                       >> $Cfile
echo "$ModAdjc"                                                   >> $Cfile
echo                                                              >> $Cfile

echo "$ModCpl"                                                    >> $Cfile
echo                                                              >> $Cfile

set text = "hbot, hppm and hadv_noop"
echo "// options are" $text                                       >> $Cfile
echo "$ModHadv"                                                   >> $Cfile
echo                                                              >> $Cfile

set text = "vbot, vppm and vadv_noop"
echo "// options are" $text                                       >> $Cfile
echo "$ModVadv"                                                   >> $Cfile
echo                                                              >> $Cfile

set text = "unif, multi_scale and hdiff_noop"
echo "// options are" $text                                       >> $Cfile
echo "$ModHdiff"                                                  >> $Cfile
echo                                                              >> $Cfile

set text = "eddy and vdiff_noop"
echo "// options are" $text                                       >> $Cfile
echo "$ModVdiff"                                                  >> $Cfile
echo                                                              >> $Cfile

set text = "phot and phot_noop"
echo "// options are" $text                                       >> $Cfile
echo "$ModPhot"                                                   >> $Cfile
echo                                                              >> $Cfile

set text = "ping_qssa, ping_smvgear, ping_mebi_cb4_1 and ping_noop"
echo "// options are" $text                                       >> $Cfile
echo "$ModPing"                                                   >> $Cfile
echo                                                              >> $Cfile

set text = "qssa, smvgear, mebi_cb4_1 and chem_noop"
echo "// options are" $text                                       >> $Cfile
echo "$ModChem"                                                   >> $Cfile
echo                                                              >> $Cfile

set text = "aero1, aero2 and aero_noop"
echo "// options are" $text                                       >> $Cfile
echo "$ModAero"                                                   >> $Cfile
```

```
 echo                                                                  >> $Cfile

 set text = "aero_depv1, aero_depv2 and aero_depv_noop"
 echo "// options are" $text                                          >> $Cfile
 echo "$ModAdepv"                                                    >> $Cfile
 echo                                                                  >> $Cfile

 set text = "cloud_radm and cloud_noop"
 echo "// options are" $text                                          >> $Cfile
 echo "$ModCloud"                                                    >> $Cfile
 echo                                                                  >> $Cfile

 set text = "pa and pa_noop, which requires the"
 echo "// options are" $text "replacement of the three"               >> $Cfile
 set text = "// global include files with their pa_noop counterparts"
 echo $text                                                            >> $Cfile
 echo "$ModPa"                                                       >> $Cfile
 echo                                                                  >> $Cfile

 echo "$ModUtil"                                                     >> $Cfile
 echo                                                                  >> $Cfile

 echo "$Modaeaqsolver"                                               >> $Cfile
 echo                                                                  >> $Cfile

 if ( $?ModMisc ) then
    echo "$ModMisc"                                                  >> $Cfile
    echo                                                            >> $Cfile
    endif

#> make the makefile or the model executable

 if ( $?MakeOpt ) then
    $Blder -make $Cfile   # $Cfile = ${CFG}.bld
    else
    set NoMake
    $Blder $Cfile
    endif
 if ( $status != 0 ) then
    echo "   *** failure in $Blder ***"
    exit 1
    endif
 if ( -e "$Base/${CFG}" ) then
    echo "   >>> previous ${CFG} exists, re-naming to ${CFG}.old <<<"
    unalias mv
    mv $Base/${CFG} $Base/${CFG}.old
    endif
 cp ${CFG}.bld $Base/${CFG}
 if ( ( $Opt != no_compile ) && \
      ( $Opt != no_link    ) && \
      ( $Opt != parse_only ) && \
      ( $Opt != show_only  ) && \
        $?NoMake ) then
    mv $MODEL $Base
    endif

 exit
```

A-4 Script used to compile Models-3/CMAQ with APT

```
#!/bin/csh –f
# bldit.apt: Script to build CMAQ-APT

#-------------------------------------------------------------
#    IMPORTANT
# Before using this script, the user should first set
# correct pathnames for REPOSITORY, FTN and FC
# in makefile.scilib and makefile.merge_concs
# in $REPOSITORY/models/CCTM/src/ping/ping_apt/scichem directory
# where REPOSITORY refers to the location of the cmaq-repository directory.

# The user can then proceed with this script

#-------------------------------------------------------------
#   BEGIN bldit.apt SCRIPT


# User needs to specify the pathname for
# REPOSITORY (CMAQ-Repository) and FC (Fortran compiler)

set FC = /usr/pgi/linux86/5.2/bin/pgf90
setenv REPOSITORY /usr2/cp194/cmaq-repository/oct2004

#-------------------------------------------------------------

setenv M3MODEL $REPOSITORY/models    # code archive
setenv M3LIB   $REPOSITORY/lib       # libraries

set BLD_OS = Linux
set Base = $cwd

 if ( ! -e $M3MODEL || ! -e $M3LIB ) then
    echo "   $M3MODEL or $M3LIB directory not found"
    exit 1
    endif
 echo "   Model archive path: $M3MODEL"
 echo "         library path: $M3LIB"

 set echo


#=====================================================================
# Copy and build Scichem

cp -p -r $M3MODEL/CCTM/src/ping/ping_apt/scichem .
cd scichem/odepack
make >& make.log
if (!(-e ../lib/libodepack.a)) then
   echo "libodepack.a not created. Check make.log in odepack directory"
   exit()
endif
cd ..
make -f makefile.scilib >& make.scilib.log
if (!(-e lib/libsci.a)) then
   echo "libsci.a not created. Check make.scilib.log in scichem directory"
   exit()
endif
make -f makefile.merge_concs >& make.merge_concs.log
if (!(-e bin/merge_concs)) then
   echo "merge_concs not created. Check make.merge_concs.log in scichem directory"
   exit()
```

```
endif
cd $Base

#==========================================================================

#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#
#> user choices: cvs archives
 set Project = $M3MODEL/CCTM
 set GlobInc = $M3MODEL/include/release

 set APPL  = cb4_yb_apt
 set CFG   = cfg.$APPL
 set MODEL = CCTM_$APPL

#> user choices: m3bld command
#set Opt = compile_all  # force compile, even if object files are current
#set Opt = clean_up     # remove all source files upon successful completion
#set Opt = no_compile   # do everything except compile
#set Opt = no_link      # do everything except link
#set Opt = one_step     # compile and link in one step
#set Opt = parse_only   # checks config file syntax
#set Opt = show_only    # show requested commands but doesn't execute them
 set Opt = verbose      # show requested commands as they are executed
 set MakeOpt            # builds a Makefile to make the model

#> user choices:  single or multiple processors
#set ParOpt             # multiple PE's

#> user choices: various modules

 set Revision = release       # release = latest CVS revision
#set Revision = '"STA3_2"'
#> NOTE: m3bld will try to compile with existing code; it will not retrieve
#>       new (different release) code. So if your "BLD" directory contains
#>       code from a release different than the one you have specified above,
#>       m3bld will tell you, but will merrily compile the original code.
#>       The workaround is to remove your "BLD" directory and start fresh.

 set ModDriver = ( module ctm               $Revision; )

 if ( $?ParOpt ) then
    set ModPar = ( module par               $Revision; )
    else
    set ModPar = ( module par_noop          $Revision; )
    endif

 set ModInit   = ( module init              $Revision; )

#set ModAdjc   = ( module adjcon_noop       $Revision; )
 set ModAdjc   = ( module denrate           $Revision; )

 set ModCpl    = ( module gencoor           $Revision; )

#set ModHadv   = ( module hadv_noop         $Revision; )
 set ModHadv   = ( module hppm              $Revision; )

#set ModVadv   = ( module vadv_noop         $Revision; )
 set ModVadv   = ( module vppm              $Revision; )

#set ModHdiff  = ( module hdiff_noop        $Revision; )
 set ModHdiff  = ( module multiscale        $Revision; )

#set ModVdiff  = ( module vdiff_noop        $Revision; )
#set ModVdiff  = ( module acm               $Revision; )
```

```
 set ModVdiff  = ( module eddy              $Revision; )

#set ModPhot   = ( module phot_noop         $Revision; )
 set ModPhot   = ( module phot              $Revision; )

#set ModPing   = ( module ping_noop         $Revision; )
#set ModPing   = ( module ping_smvgear      $Revision; )
 set ModPing   = ( module ping_apt          $Revision; )

#set ModChem   = ( module chem_noop         $Revision; )
#set ModChem   = ( module smvgear           $Revision; )
#set ModChem   = ( module ros3              $Revision; )
#set ModChem   = ( module ebi_cb4           $Revision; )
#set ModChem   = ( module ebi_saprc99       $Revision; )
#set ModChem   = ( module mebi_radm2_cis4   $Revision; )
 set ModChem   = ( module yb                $Revision; )

 set ModAero   = ( module aero_noop         $Revision; )
#set ModAero   = ( module aero3             $Revision; )

 set ModAdepv  = ( module aero_depv_noop    $Revision; )
#set ModAdepv  = ( module aero_depv2        $Revision; )

 set ModCloud  = ( module cloud_noop        $Revision; )
#set ModCloud  = ( module cloud_radm        $Revision; )

if ($ModCloud[2] == 'cloud_noop')then
   set CVD =
else
   set CVD = -Dcld_proc
endif


 set ModPa     = ( module pa                $Revision; )

 set ModUtil   = ( module util              $Revision; )


 set Mechanism = cb4
 set Tracer    = trac0

#> user choices: emissions processing in chem or vdiff (default) ...
#set Cemis

#> user choices: set process analysis linkages
 set PABase    = $GlobInc
 set PAOpt     = pa_noop

#> other user choices set below are:
#>    name of the "BLD" directory
#>    compiler/link flags
#>    library paths
#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#

 set Bld = $Base/BLD_${APPL}
#unset echo
 if ( ! -e "$Bld" ) then
    mkdir $Bld
    else
    if ( ! -d "$Bld" ) then
       echo "   *** target exists, but not a directory ***"
       exit 1
       endif
    endif
```

```
#set echo
 cd $Bld

#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#:#

 set STENEX = ${M3LIB}/stenex_noop/${BLD_OS}
 set MODLOC = ${Base}/MOD_DIR
 if ( ! -d "$MODLOC" ) mkdir -p $MODLOC

 if ( $?ParOpt ) then
#   set Mpich = /usr/local/mpich-1.2.4       # compiled for ch_shmem
    set Mpich = /share/linux/bin/mpich-ch_p4 # compiled for ch_p4
    set seL = se_snl
    cp -p ${STENEX}/*.mod $MODLOC
    set LIB2 = "-L${M3LIB}/pario/${BLD_OS} -lpario"
    set LIB3 =
    set LIB4 = "-L${Mpich}/lib -lmpich"
    set Str1 = (// Parallel / Include message passing definitions)
    set Str2 = (include SUBST_MPICH ${Mpich}/include/mpif.h;)
    else
    set Mpich =
    set seL = sef90_noop
    cp -p ${STENEX}/noop_*.mod $MODLOC
    set LIB2 =
    set LIB3 =
    set LIB4 =
    set Str1 =
    set Str2 =
    endif

 set LIB1 = "-L${STENEX} -l${seL}"
 set LIB5 = "-L${M3LIB}/ioapi/ioapi_22/Linux2_x86pg -lioapi"
 set LIB6 = "-L${M3LIB}/netCDF/${BLD_OS} -lnetcdf"
 set SCILIB = "-L${Base}/scichem/lib -lsci -lodepack"
 set LIBS = "$LIB1 $LIB2 $LIB3 $LIB4 $LIB5 $LIB6 $SCILIB"

 set FP = $FC

 set FSTD       = "-Mfixed -Mextend"
 set F_FLAGS    = "${FSTD} -fast -module ${MODLOC} -I."
 set CPP_FLAGS  = ""
#set C_FLAGS    = "-v -I${Mpich}/include"
 set C_FLAGS    = "-Xc -v"
 set LINK_FLAGS = "-Bstatic"

 set Blder = ${M3LIB}/build/$BLD_OS/m3bld

 set ICL_PAR   = $GlobInc
 set ICL_CONST = $GlobInc
 set ICL_FILES = $GlobInc
 set ICL_EMCTL = $GlobInc
 set ICL_IOAPI = ${M3LIB}/ioapi/ioapi_22/fixed_src
 set ICL_VCRD  = $Base/icl
 set ICL_MECH  = $GlobInc/$Mechanism
 set ICL_TRAC  = $GlobInc/$Tracer
 set ICL_PA    = $PABase/$PAOpt
 set ICL_APT   = $M3MODEL/CCTM/src/ping/ping_apt

 if ( $?Cemis ) then
    set CV = -Demis_chem
    else
    set CV =
    endif
```

```
#> NOTE: To run parallel in a Scyld Beowulf cluster, e.g., remove the
#>       "-Dcluster\" below.

 if ( $?ParOpt ) then   # split to avoid line > 256 char
    set PAR = ( -Dparallel\
                -Dcluster\
                -DINTERPB=PINTERPB\
                -DM3ERR=PM3ERR\
                -DM3EXIT=PM3EXIT\
                -DM3WARN=PM3WARN\
                -DSHUT3=PSHUT3\
                -DWRITE3=PWRITE3 )

    set Popt = SE
    else
    echo "   Not Parallel; set Serial (no-op) flags"
    set PAR = "-DINTERPB=INTERP3"
    set Popt = NOOP
    endif

 set STX1 = ( -DSUBST_MODULES=${Popt}_MODULES\
              -DSUBST_BARRIER=${Popt}_BARRIER )
 set STX2 = ( -DSUBST_GLOBAL_MAX=${Popt}_GLOBAL_MAX\
              -DSUBST_GLOBAL_MIN=${Popt}_GLOBAL_MIN\
              -DSUBST_GLOBAL_MIN_DATA=${Popt}_GLOBAL_MIN_DATA\
              -DSUBST_GLOBAL_TO_LOCAL_COORD=${Popt}_GLOBAL_TO_LOCAL_COORD\
              -DSUBST_GLOBAL_SUM=${Popt}_GLOBAL_SUM\
              -DSUBST_GLOBAL_LOGICAL=${Popt}_GLOBAL_LOGICAL\
              -DSUBST_LOOP_INDEX=${Popt}_LOOP_INDEX\
              -DSUBST_SUBGRID_INDEX=${Popt}_SUBGRID_INDEX )
 set STX3 = ( -DSUBST_HI_LO_BND_PE=${Popt}_HI_LO_BND_PE\
              -DSUBST_SUM_CHK=${Popt}_SUM_CHK\
              -DSUBST_INIT_ARRAY=${Popt}_INIT_ARRAY\
              -DSUBST_COMM=${Popt}_COMM\
              -DSUBST_MY_REGION=${Popt}_MY_REGION\
              -DSUBST_SLICE=${Popt}_SLICE\
              -DSUBST_GATHER=${Popt}_GATHER\
              -DSUBST_DATA_COPY=${Popt}_DATA_COPY\
              -DSUBST_IN_SYN=${Popt}_IN_SYN )

 setenv CVSROOT $Project

#> make the config file

 set Cfile = ${CFG}.bld
 set quote = '"'

 echo                                                              > $Cfile
 echo "model       $MODEL;"                                       >> $Cfile
 echo                                                             >> $Cfile
 echo "FPP         $FP;"                                          >> $Cfile
 echo                                                             >> $Cfile
 set text = "$quote$CPP_FLAGS $CV $CVD $PAR $STX1 $STX2 $STX3$quote;"
 echo "cpp_flags   $text"                                         >> $Cfile
 echo                                                             >> $Cfile
 echo "f_compiler  $FC;"                                          >> $Cfile
 echo                                                             >> $Cfile
 echo "f_flags     $quote$F_FLAGS$quote;"                         >> $Cfile
 echo                                                             >> $Cfile
 echo "c_flags     $quote$C_FLAGS$quote;"                         >> $Cfile
 echo                                                             >> $Cfile
 echo "link_flags  $quote$LINK_FLAGS$quote;"                      >> $Cfile
 echo                                                             >> $Cfile
 echo "libraries   $quote$LIBS$quote;"                            >> $Cfile
```

```
echo                                                                    >> $Cfile
echo "global      $Opt;"                                                >> $Cfile
echo                                                                    >> $Cfile

set text="// mechanism and tracer:"
echo "$text ${Mechanism}, ${Tracer}"                                    >> $Cfile
echo "// project archive: ${Project}"                                   >> $Cfile
echo                                                                    >> $Cfile

echo "include SUBST_PE_COMM    $ICL_PAR/PE_COMM.EXT;"                   >> $Cfile
echo "include SUBST_CONST      $ICL_CONST/CONST.EXT;"                   >> $Cfile
echo "include SUBST_FILES_ID   $ICL_FILES/FILES_CTM.EXT;"               >> $Cfile
echo "include SUBST_EMPR_VD    $ICL_EMCTL/EMISPRM.vdif.EXT;"            >> $Cfile
echo "include SUBST_EMPR_CH    $ICL_EMCTL/EMISPRM.chem.EXT;"            >> $Cfile
echo "include SUBST_IOPARMS    $ICL_IOAPI/PARMS3.EXT;"                  >> $Cfile
echo "include SUBST_IOFDESC    $ICL_IOAPI/FDESC3.EXT;"                  >> $Cfile
echo "include SUBST_IODECL     $ICL_IOAPI/IODECL3.EXT;"                 >> $Cfile
echo "include SUBST_COORD_ID   $ICL_VCRD/COORD.EXT;"                    >> $Cfile
echo "include SUBST_VGRD_ID    $ICL_VCRD/VGRD.EXT;"                     >> $Cfile

echo "include SUBST_RXCMMN     $ICL_MECH/RXCM.EXT;"                     >> $Cfile
echo "include SUBST_RXDATA     $ICL_MECH/RXDT.EXT;"                     >> $Cfile
echo "include SUBST_ARRAYMAP   $ICL_MECH/ARRAYMAP.EXT;"                 >> $Cfile
echo "include SUBST_PARAMETER  $ICL_MECH/PARAMETER.EXT;"                >> $Cfile
echo "include SUBST_AQ_PARAMS  $ICL_MECH/AQ_PARAMS.EXT;"                >> $Cfile
echo "include SUBST_HETERPAR   $ICL_MECH/HETERPAR.EXT;"                 >> $Cfile
echo "include SUBST_GC_SPC     $ICL_MECH/GC_SPC.EXT;"                   >> $Cfile
echo "include SUBST_GC_EMIS    $Base/icl/$Mechanism/GC_EMIS.EXT;"       >> $Cfile
echo "include SUBST_GC_ICBC    $ICL_MECH/GC_ICBC.EXT;"                  >> $Cfile
echo "include SUBST_GC_DIFF    $ICL_MECH/GC_DIFF.EXT;"                  >> $Cfile
echo "include SUBST_GC_DDEP    $Base/icl/$Mechanism/GC_DDEP.EXT;"       >> $Cfile
echo "include SUBST_GC_DEPV    $Base/icl/$Mechanism/GC_DEPV.EXT;"       >> $Cfile
echo "include SUBST_GC_ADV     $ICL_MECH/GC_ADV.EXT;"                   >> $Cfile
echo "include SUBST_GC_CONC    $ICL_MECH/GC_CONC.EXT;"                  >> $Cfile
echo "include SUBST_GC_G2AE    $ICL_MECH/GC_G2AE.EXT;"                  >> $Cfile
echo "include SUBST_GC_G2AQ    $ICL_MECH/GC_G2AQ.EXT;"                  >> $Cfile
echo "include SUBST_GC_SCAV    $ICL_MECH/GC_SCAV.EXT;"                  >> $Cfile
echo "include SUBST_GC_WDEP    $ICL_MECH/GC_WDEP.EXT;"                  >> $Cfile
echo "include SUBST_AE_SPC     $ICL_MECH/AE_SPC.EXT;"                   >> $Cfile
echo "include SUBST_AE_EMIS    $ICL_MECH/AE_EMIS.EXT;"                  >> $Cfile
echo "include SUBST_AE_ICBC    $ICL_MECH/AE_ICBC.EXT;"                  >> $Cfile
echo "include SUBST_AE_DIFF    $ICL_MECH/AE_DIFF.EXT;"                  >> $Cfile
echo "include SUBST_AE_DDEP    $ICL_MECH/AE_DDEP.EXT;"                  >> $Cfile
echo "include SUBST_AE_DEPV    $ICL_MECH/AE_DEPV.EXT;"                  >> $Cfile
echo "include SUBST_AE_ADV     $ICL_MECH/AE_ADV.EXT;"                   >> $Cfile
echo "include SUBST_AE_CONC    $ICL_MECH/AE_CONC.EXT;"                  >> $Cfile
echo "include SUBST_AE_A2AQ    $ICL_MECH/AE_A2AQ.EXT;"                  >> $Cfile
echo "include SUBST_AE_SCAV    $ICL_MECH/AE_SCAV.EXT;"                  >> $Cfile
echo "include SUBST_AE_WDEP    $ICL_MECH/AE_WDEP.EXT;"                  >> $Cfile
echo "include SUBST_NR_SPC     $ICL_MECH/NR_SPC.EXT;"                   >> $Cfile
echo "include SUBST_NR_EMIS    $ICL_MECH/NR_EMIS.EXT;"                  >> $Cfile
echo "include SUBST_NR_ICBC    $ICL_MECH/NR_ICBC.EXT;"                  >> $Cfile
echo "include SUBST_NR_DIFF    $ICL_MECH/NR_DIFF.EXT;"                  >> $Cfile
echo "include SUBST_NR_DDEP    $ICL_MECH/NR_DDEP.EXT;"                  >> $Cfile
echo "include SUBST_NR_DEPV    $ICL_MECH/NR_DEPV.EXT;"                  >> $Cfile
echo "include SUBST_NR_ADV     $ICL_MECH/NR_ADV.EXT;"                   >> $Cfile
echo "include SUBST_NR_N2AE    $ICL_MECH/NR_N2AE.EXT;"                  >> $Cfile
echo "include SUBST_NR_N2AQ    $ICL_MECH/NR_N2AQ.EXT;"                  >> $Cfile
echo "include SUBST_NR_SCAV    $ICL_MECH/NR_SCAV.EXT;"                  >> $Cfile
echo "include SUBST_NR_WDEP    $ICL_MECH/NR_WDEP.EXT;"                  >> $Cfile
echo "include SUBST_TR_SPC     $ICL_TRAC/TR_SPC.EXT;"                   >> $Cfile
echo "include SUBST_TR_EMIS    $ICL_TRAC/TR_EMIS.EXT;"                  >> $Cfile
echo "include SUBST_TR_ICBC    $ICL_TRAC/TR_ICBC.EXT;"                  >> $Cfile
echo "include SUBST_TR_DIFF    $ICL_TRAC/TR_DIFF.EXT;"                  >> $Cfile
```

```
 echo "include SUBST_TR_DDEP    $ICL_TRAC/TR_DDEP.EXT;"           >> $Cfile
 echo "include SUBST_TR_DEPV    $ICL_TRAC/TR_DEPV.EXT;"           >> $Cfile
 echo "include SUBST_TR_ADV     $ICL_TRAC/TR_ADV.EXT;"            >> $Cfile
 echo "include SUBST_TR_T2AQ    $ICL_TRAC/TR_T2AQ.EXT;"           >> $Cfile
 echo "include SUBST_TR_SCAV    $ICL_TRAC/TR_SCAV.EXT;"           >> $Cfile
 echo "include SUBST_TR_WDEP    $ICL_TRAC/TR_WDEP.EXT;"           >> $Cfile
 echo                                                             >> $Cfile

 set text = "// Process Analysis / Integrated Reaction Rates processing"
 echo $text                                                       >> $Cfile
#echo "include SUBST_PACTL_ID    $ICL_PA/PA_CTL_no_irr.EXT;"     >> $Cfile
 echo "include SUBST_PACTL_ID    $ICL_PA/PA_CTL.EXT;"            >> $Cfile
 echo "include SUBST_PACMN_ID    $ICL_PA/PA_CMN.EXT;"            >> $Cfile
 echo "include SUBST_PADAT_ID    $ICL_PA/PA_DAT.EXT;"            >> $Cfile
 echo                                                             >> $Cfile

 set text = "// Include file for APT"
 echo $text                                                       >> $Cfile
 echo "include SUBST_FILES_APT   $ICL_APT/FILES_APT.diag.EXT;"   >> $Cfile
 echo                                                             >> $Cfile

 echo "$Str1"                                                     >> $Cfile
 echo "$Str2"                                                     >> $Cfile
 echo                                                             >> $Cfile

 echo "$ModDriver"                                                >> $Cfile
 echo                                                             >> $Cfile

 echo "$ModPar"                                                   >> $Cfile
 echo                                                             >> $Cfile

 echo "$ModInit"                                                  >> $Cfile
 echo                                                             >> $Cfile

 set text = "denrate and adjcon_noop"
 echo "// options are" $text                                     >> $Cfile
 echo "$ModAdjc"                                                  >> $Cfile
 echo                                                             >> $Cfile

 echo "$ModCpl"                                                   >> $Cfile
 echo                                                             >> $Cfile

 set text = "hppm and hadv_noop"
 echo "// options are" $text                                     >> $Cfile
 echo "$ModHadv"                                                  >> $Cfile
 echo                                                             >> $Cfile

 set text = "vppm and vadv_noop"
 echo "// options are" $text                                     >> $Cfile
 echo "$ModVadv"                                                  >> $Cfile
 echo                                                             >> $Cfile

 set text = "unif, multi_scale and hdiff_noop"
 echo "// options are" $text                                     >> $Cfile
 echo "$ModHdiff"                                                 >> $Cfile
 echo                                                             >> $Cfile

 set text = "eddy and vdiff_noop"
 echo "// options are" $text                                     >> $Cfile
 echo "$ModVdiff"                                                 >> $Cfile
 echo                                                             >> $Cfile

 set text = "phot and phot_noop"
 echo "// options are" $text                                     >> $Cfile
```

```
 echo "$ModPhot"                                                    >> $Cfile
 echo                                                               >> $Cfile

 set text = "ping_apt, ping_qssa, ping_smvgear, ping_mebi_cb4_1 and ping_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModPing"                                                  >> $Cfile
 echo                                                               >> $Cfile

 set text = "qssa, yb, smvgear, mebi_cb4_1 and chem_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModChem"                                                  >> $Cfile
 echo                                                               >> $Cfile

 set text = "aero1, aero2 and aero_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModAero"                                                  >> $Cfile
 echo                                                               >> $Cfile

 set text = "aero_depv1, aero_depv2 and aero_depv_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModAdepv"                                                 >> $Cfile
 echo                                                               >> $Cfile

 set text = "cloud_radm and cloud_noop"
 echo "// options are" $text                                       >> $Cfile
 echo "$ModCloud"                                                 >> $Cfile
 echo                                                               >> $Cfile

 set text = "pa and pa_noop, which requires the"
 echo "// options are" $text "replacement of the three"           >> $Cfile
 set text = "// global include files with their pa_noop counterparts"
 echo $text                                                         >> $Cfile
 echo "$ModPa"                                                    >> $Cfile
 echo                                                               >> $Cfile

 echo "$ModUtil"                                                  >> $Cfile
 echo                                                               >> $Cfile

 if ( $?ModMisc ) then
    echo "$ModMisc"                                               >> $Cfile
    echo                                                          >> $Cfile
    endif

#> make the makefile or the model executable

 if ( $?MakeOpt ) then
    $Blder -make $Cfile   # $Cfile = ${CFG}.bld
    else
    set NoMake
    $Blder $Cfile
    endif
 if ( $status != 0 ) then
    echo "   *** failure in $Blder ***"
    exit 1
    endif
 if ( -e "$Base/${CFG}" ) then
    echo "   >>> previous ${CFG} exists, re-naming to ${CFG}.old <<<"
    unalias mv
    mv $Base/${CFG} $Base/${CFG}.old
    endif
 cp ${CFG}.bld $Base/${CFG}
 if ( ( $Opt != no_compile ) && \
      ( $Opt != no_link    ) && \
      ( $Opt != parse_only ) && \
```

```
     ( $Opt != show_only  ) && \
       $?NoMake ) then
   mv $MODEL $Base
   endif

exit
```

Appendix B

MODELS-3/CMAQ-APT VERSION 2.0 USER'S GUIDE

MODELS-3/CMAQ-APT VERSION 2.0 USER'S GUIDE

Prepared for

EPRI

3412 Hillview Avenue

Palo Alto, CA 94304

Prepared by

Prakash Karamchandani, Krishnakumar Vijayaraghavan, and Christian Seigneur

Atmospheric & Environmental Research, Inc.

2682 Bishop Drive, Suite 120

San Ramon, CA 94583

March 2001

Document Number CP080-01-2

ACKNOWLEDGEMENTS

Lynne Santos and Ian Sykes, ARAP/Titan Corporation

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (continued)

LIST OF FIGURES (continued)

1. INTRODUCTION

The Models-3 Community Multiscale Air Quality modeling system with Advanced Plume Treatment (Models-3/CMAQ-APT) is a new state-of-the-science plume-in-grid air quality model developed to provide a more realistic representation of the behavior of reactive plumes in the atmosphere within an Eulerian grid model framework. The development of the model has been sponsored by EPRI. This plume-in-grid model consists of a reactive plume model, SCICHEM, imbedded into a three-dimensional grid-based model, Models-3/CMAQ, referred to as the host model. These two components of Models-3/CMAQ-APT are described briefly below.

The reactive plume model used for the plume-in-grid treatment is the Second-order Closure Integrated puff model (SCIPUFF) with CHEMistry (SCICHEM). SCICHEM has been developed by Titan/ARAP and Atmospheric & Environmental Research, Inc. (AER). Plume transport and dispersion are simulated with SCIPUFF, a model that uses a second-order closure approach to solve the turbulent diffusion equations (Sykes et al., 1988, 1993; Sykes and Henn, 1995). The plume is represented by a myriad of three-dimensional puffs that are advected and dispersed according to the local micrometeorological characteristics. Each puff has a Gaussian representation of the concentrations of emitted inert species. The overall plume, however, can have any spatial distribution of these concentrations, since it consists of a multitude of puffs that are independently affected by the transport and dispersion characteristics of the atmosphere. SCIPUFF can simulate the effect of wind shear since individual puffs will evolve according to their respective locations in an inhomogenous velocity field. As puffs grow larger, they may encompass a volume that cannot be considered homogenous in terms of the meteorological variables. A puff splitting algorithm accounts for such conditions by dividing puffs that have become too large into more smaller puffs. Conversely, puffs may overlap significantly, thereby leading to an excessive computational burden. A puff merging algorithm allows individual puffs that are affected by the same (or very similar) micro-scale meteorology to be combined into a single puff. Also, the effects of buoyancy on plume rise and initial dispersion are simulated by solving the conservation equations for mass, heat, and momentum.

The chemical reactions within the puffs are simulated using a general framework that allows any chemical kinetic mechanism to be treated. The user enters the chemical reactions and their associated rate parameters, and SCICHEM sets the corresponding system of ordinary differential equations (ODE) to be solved. Chemical species concentrations in the puffs are treated as perturbations from the background concentrations. Recent improvements to SCICHEM include the incorporation of modules for aerosol thermodynamics and aqueous-phase chemistry. A detailed description of SCICHEM can be found in EPRI (2000a). Karamchandani et al. (2000a) describe the evaluation of SCICHEM using helicopter power plant plume measurements from the 1995 Southern Oxidants Study (SOS) Nashville/Middle Tennessee Ozone Study.

Models-3/CMAQ was developed by the U.S. Environmental Protection Agency (EPA) to address multiscale multi-pollutant air pollution problems (Byun and Ching, 1999). Models-3 is the computational framework and CMAQ is the air quality modeling system. Models-3/CMAQ treats the emissions, transport, dispersion, chemical transformations, gas-particle conversion, and removal processes that govern the behavior of chemical pollutants in the atmosphere. Emissions include those from area sources (e.g., industrial, residential, agricultural, mobile and biogenic emissions) and point sources (e.g., power plants and smelters). The plume rise of point source emissions is treated in a pre-processor to CMAQ. Transport processes include advection, large-scale convection and, in the presence of cumulus clouds, subgrid-scale convection. Dispersion includes both horizontal and vertical dispersion. Chemical transformations include reactions in the gas phase and reactions in the aqueous phase (i.e., in cloud droplets). The formation of secondary aerosols is treated and it includes the gas-particle partitioning of volatile chemical species. Dry deposition is simulated for gases and particles. Wet deposition is treated for precipitating clouds that are resolved by the grid system as well as for clouds that are treated at the sub-grid scale.

The first version of Models-3/CMAQ-APT was developed in 1999. In that version, SCICHEM was incorporated into the 1998 release of Models-3/CMAQ. The model was

applied to explicitly simulate the plumes of two power plants for a five-day simulation in the Nashville/western Tennessee area. The development of the model and results from its application have been described by Karamchandani et al. (2000b).

More recently, an updated version of SCICHEM has been incorporated into the August 2000 release (Version 4.0) of Models-3/CMAQ and the model has been applied to a larger domain (NARSTO-Northeast) for 30 power plant plumes. (Karamchandani et al., 2000c). We will refer to this version of Models-3/CMAQ-APT as Version 2.0. This User's Guide provides guidance on the operation of Models-3/CMAQ-APT Version 2.0. Note that while both SCICHEM and Models-3/CMAQ offer options for the treatment of aerosols, aqueous-phase chemistry and wet deposition, the version of Models-3/CMAQ-APT described in this User's Guide is applicable only for gas-phase simulations, because the interfaces between the two models for aerosols and clouds have not yet been developed and also because of inconsistencies in the treatment of aerosol processes between the two models.

Note that this User's Guide assumes that the user has installed the Models-3 framework and is familiar with the operation of Models-3/CMAQ and the input/output file requirements of Models-3/CMAQ. A User's Manual for Models-3/CMAQ (EPA, 1998) can be obtained from EPA or downloaded from the Models-3 website at http://www.epa.gov/asmdnerl/models3/doc. This User's Guide can be considered as a supplement to the Models-3/CMAQ User's Manual and describes the additional steps required to develop, conduct and analyze a Models-3/CMAQ-APT simulation.

Section 2 provides a brief overview of the interaction between the host model, CMAQ, and the embedded reactive plume model, SCICHEM. Section 3 describes the additional input files required for a Models-3/CMAQ-APT simulation and the output files that are generated in addition to those created in a typical Models-3/CMAQ simulation. Section 3 also describes the utilities that have been developed to post-process the Models-3/CMAQ-APT output files. Section 4 describes the building of the Models-3/CMAQ-APT executable code using a sample build script for the NARSTO-Northeast domain

simulation, as well as the building of the post-processing utilities. Section 5 demonstrates the application of Models-3/CMAQ-APT using the NARSTO-Northeast domain simulation as an example.

2. INTERACTION BETWEEN MODELS-3/CMAQ AND SCICHEM

Models-3/CMAQ is highly modular. This modularity is achieved by virtually eliminating data flow dependencies among the various process modules. Each process module reads and manipulates the direct access data files independently of the other modules. The only data shared by the various modules are the three-dimensional gridded trace species concentrations. The data files are manipulated using the Input/Output Applications Program Interface (I/O API) (Coats et al., 1993; Coats, 1995; Coats et al., 1999). The I/O API is built on top of UCAR's NetCDF (Rew and Davis, 1990) which provides self-describing files accessible from both FORTRAN and C, is compatible with a variety of platforms, and makes data portable across heterogeneous computing environments.

The interface between SCICHEM and the host model exploits these structural aspects of Models-3. Figure 2-1 illustrates the interface between the host model and SCICHEM. Like the other process modules in Models-3/CMAQ, all relevant information related to the emissions and the dynamical state of the atmosphere required by SCICHEM are accessed directly from the input files; only the three-dimensional concentration fields are directly shared between the host model and SCICHEM. On input to SCICHEM, these host model concentrations serve as the background (ambient) concentrations for SCICHEM calculations. On output from SCICHEM, these concentrations are updated whenever plume-to-grid transfer occurs (see Section 2.3 below) and are returned to the host model.

SCICHEM is invoked in Models-3/CMAQ-APT by a single subroutine call similar to that of any other physical or chemical process of the host model. The SCICHEM code is compiled as a library that is linked to the host model when the executable is being built (see Section 4). The “ping_apt” module has to be specified in the build script instead of the “ping_noopt” module (i.e., no plume-in-grid treatment) or the “ping_qssa” module (EPA plume-in-grid treatment using the QSSA chemistry solver) or the “ping_smvgear” module (EPA plume-in-grid treatment using the SMVGEAR chemistry solver). The details are provided in Section 4.3.

It is necessary to maintain consistency between the plume model and the host model for those variables and processes that are treated similarly in both models (e.g., chemical mechanism, numerical solution of the equations, emissions and deposition processes). Second, the plume model requires some input variables that are the same as those required by the host model. Third, there must be some algorithm for the transfer of the puffs to the 3-D grid when the puffs become sufficiently large. We discuss each of those aspects below.

2.1 Consistency between SCICHEM and Models-3/CMAQ

The processes that are common to SCICHEM and the host model, Models-3/CMAQ, include the emissions of chemical species from point sources, the gas-phase chemistry, the meteorology governing the transport and dispersion of chemical species, and the dry deposition of chemicals from the atmosphere. Here we discuss the treatments of plume rise and chemistry in the host model and the embedded plume model.

As described in Karamchandani et al. (2000c), several modifications were made to Models-3/CMAQ-APT in developing the current version of the model. Some of these modifications, such as the addition of an alternative plume rise algorithm within SCICHEM and the addition of the Young & Boris chemical solver to Models-3/CMAQ, were introduced to enhance the computational efficiency of the model. In addition to reducing the CPU requirements of the model, these modifications resulted in a consistent treatment of plume rise and chemistry in the host model and the embedded plume model.

The original SCICHEM formulation for plume rise uses a dynamic model for the buoyancy and momentum rise calculations that results in a large number of puffs that must be carried in the simulation (EPRI, 2000a). This considerably increases the computational time requirements of SCICHEM. In Models-3/CMAQ-APT Version 2.0, an alternative plume rise algorithm has been implemented in SCICHEM that is based on that used in the Sparse Matrix Operator Kernel Emissions (SMOKE) modeling system. This algorithm is also used in the Models-3/CMAQ Emission-Chemistry Interface Processor (ECIP) as well as in the processor developed by MCNC for EPRI for

converting UAM files to Models-3/CMAQ format. As described in Section 3.1.1, the user selects this option by setting the plume dynamics flag to "FALSE" in the SCICHEM control input file, i.e., the puffs are assumed to be "passive" and the dynamic buoyancy and momentum rise calculations are not performed.

SCICHEM offers the flexibility of using any gas-phase chemical kinetic mechanism. The species and reactions are specified in the SCICHEM control input file (see Section 3.1.1). The reaction rate constants are calculated using the same data and formulas used by the host model. All applications of Models-3/CMAQ-APT to date have used the CB-IV mechanism.

The numerical solution of the chemical kinetic equations is typically the most time-consuming step in an air quality simulation. As discussed in Karamchandani et al. (2000c), we identified the Young & Boris solver as offering the best compromise between computational speed and accuracy. This solver was already available in SCICHEM. For consistency and computational efficiency, this solver was implemented as an additional module in the Models-3/CMAQ framework. This solver is selected in the Models-3/CMAQ and Models-3/CMAQ-APT build scripts by specifying the "chemeq" module (see Section 4.3).

2.2 Models-3/CMAQ Input Files Used by SCICHEM

As shown in Figure 2-1, SCICHEM obtains data from 5 Models-3/CMAQ input files in addition to the plume-in-grid specific input files. Variables that are used jointly by SCICHEM and the host model include:

- Time-dependent three-dimensional meteorological variables, such as winds, temperatures, pressures, and humidities (MET_CRO_3D and MET_DOT_3D files)

- Time-dependent two-dimensional meteorological and micrometeorological variables, such as friction velocities, Monin-Obukhov lengths, cloud cover (MET_CRO_2D file)

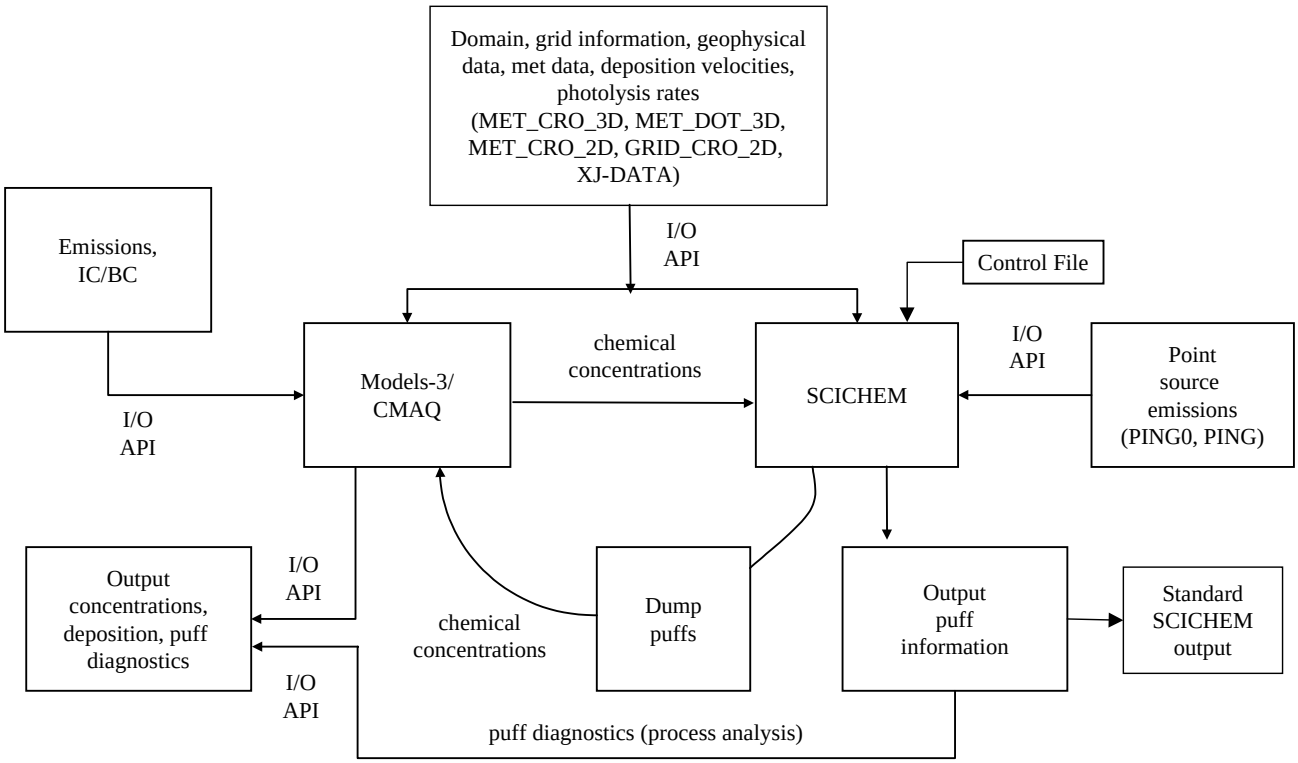

Figure 2-1. Interface between SCICHEM and Models-3/CMAQ.

- Time-dependent dry deposition velocities of the chemical species (MET_CRO_2D file)

- Time-independent two-dimensional variables, such as latitudes and longitudes, roughness lengths, map factors, and terrain elevations (GRID_CRO_2D file)

- Photolysis rates look up table (XJ_DATA file)

The XJ_DATA file is a text file while all the other files are in NetCDF format. The NetCDF files are directly read by SCICHEM using the I/O API in a manner similar to that used by the other process modules in Models-3/CMAQ.

2.3 Transfer of Information Between Models-3/CMAQ and SCICHEM

As mentioned previously, only the three-dimensional host model concentrations are directly exchanged between the host model and the embedded plume model. The SCICHEM gas-phase chemistry involves chemical species from the ambient background and chemical species emitted from the point source. The host model concentrations serve as the ambient background concentrations. The SCICHEM chemical concentrations are treated as a perturbation over the ambient background. Note that this perturbation can be positive (i.e., the concentration in the plume is greater than that in the background) or negative (i.e., the concentration in the plume is smaller than that in the background). For example, NO_x concentrations in the plume near the stack will correspond to a positive perturbation whereas O_3 concentrations will correspond to a negative perturbation. When SCICHEM is invoked from the host model, the three-dimensional host model concentrations are input to SCICHEM.

Once the plume is mature enough, it is appropriate to release the plume material into the 3-D grid system of the host model. The earlier application of Models-3/CMAQ-APT used a criterion for releasing the plume material into the grid system that was solely based on plume dimensions (Karamchandani et al., 2000b). In Models-3/CMAQ-APT Version 2.0, this plume-to-grid transfer criterion has been augmented by taking into account the maturity of plume chemistry in addition to the plume dimensions, as

described in Karamchandani et al. (2000c). When SCICHEM finishes its simulation for a given time step, the puff-to-grid transfer criteria are checked for each puff. If any puffs satisfy the criteria, the material in these puffs are transferred to the grid, i.e., the three-dimensional host model concentrations are updated and returned from SCICHEM as an output to the host model. All dumped puffs are then eliminated from subsequent SCICHEM calculations.

3. ADDITIONAL MODELS-3/CMAQ-APT INPUT/OUTPUT FILES

Besides the standard input and output files involved in a typical Models-3/CMAQ simulation, the plume-in-grid simulation requires 3 additional input files and generates several more output files. Note that many of the additional output files are identical to those created by SCICHEM in stand-alone mode and are described in detail in the SCICHEM User's Guide (EPRI, 2000b). In this section, we describe the additional input files required by Models-3/CMAQ-APT and the preparation of these input files. We also list the relevant standard SCICHEM output files and discuss how these are combined with the host model output files using utilities developed for the plume-in-grid application. In addition, we describe the new SCICHEM process analysis or diagnostic output files that are applicable to Models-3/CMAQ-APT. These files are in the NetCDF format, allowing the user to view the plume-in-grid diagnostics with PAVE.

3.1 Additional Input Files Required for Models-3/CMAQ-APT

In addition to the standard files required by Models-3/CMAQ, three files are required for a plume-in-grid application with Models-3/CMAQ-APT. Two of these files provide information on the point sources that are being simulated by the plume model and are in NetCDF format. These files are prepared using a processor developed by MCNC as part of this study under a separate EPRI-sponsored project. The third file is the control file containing the model options, and is in ASCII format. This file consists of Fortran NAMELIST statements, and is a combination of the following three input files used in stand-alone SCICHEM simulations and described in EPRI (2000b): the main SCICHEM input file, the meteorology scenario file, and the multicomponent input file. Further details on the Models-3/CMAQ-APT additional input files are provided below.

3.1.1 Models-3/CMAQ-APT control file

The Models-3/CMAQ-APT control file has the name "**ProjectName.inp**", where "ProjectName" is the name of the project. For example, for the NARSTO-Northeast application, the file name is "**narsto.inp**". This file consists of three sections. The first two sections (Namelists **OPTIONS** and **MET**) contain run control and model options for

the transport and dispersion component of SCICHEM (i.e., SCIPUFF). Note that many of the SCIPUFF control parameters described in the stand-alone SCICHEM User's Guide (EPRI, 2000b) are not applicable for the plume-in-grid version.

The third section is the "multicomponent" section for the chemistry component of SCICHEM. This section consists of one namelist (**CONTROL**) section specifying the chemistry and plume-in-grid specific parameters, a "multicomponent species" section listing the chemical species that are being carried in the model, a "balance" section listing the nitrogen species that are to be used to preserve nitrogen balance in the simulation, and a "multicomponent equation" section listing the chemical reactions that the species will undergo. Each section begins with "#Section name" (e.g., #Control, #Species, etc.). All sections are unformatted and need only be space delimited.

All namelist parameters must be contained within the '**&groupname**' and '**&end**' lines in the control file. The '**&**' on these lines must be preceded by at least one space in order for the program to recognize this block as namelist input. The '**&end**' may be replaced with a '/'. All lines contained outside of the namelist sections are considered comments and are disregarded by SCICHEM, except the multicomponent file section (see below). Some input parameters have default values; if a parameter does not have a default, the program will abort if the parameter is not included in the namelist.

The parameters for the **OPTIONS** and **MET** namelists are listed below. These parameters are also described in more detail in the stand-alone SCICHEM User's Guide (EPRI, 2000b).

Namelist: **OPTIONS** - optional parameters

t_avg - Conditional averaging time (seconds) for defining the diffusive component of turbulence (REAL*4). The default is no conditional averaging.

cmin - Minimum puff mass (REAL*4) of the tracer in ppm. The default is zero.

lsplitz - Flag for vertical puff splitting within the planetary boundary layer (LOGICAL). If 'True', puffs are not split in the vertical direction within the boundary layer. The default is 'False'.

delmin - Minimum grid size (meters) for the adaptive surface grid (REAL*4). Not used in the plume-in-grid version. The default is zero.

wwtrop - Tropospheric vertical turbulent fluctuations (m^2s^{-2}) used as the minimum value (REAL*4). The default is 0.01.

sltrop - Tropospheric vertical length scale (meters) (REAL*4). The default is 10.

epstrop - Tropospheric energy dissipation rate (m^2s^{-3}) (REAL*4). The default is 4.0×10^{-4}.

uu_calm - Minimum horizontal velocity fluctuation variance (m^2s^{-2}) (REAL*4). The default is 0.25.

sl_calm - Horizontal length scale associated with *uu_calm* (meters) (REAL*4). The default is 1000.

nzbl - Number of boundary layer vertical grid levels (INTEGER*4). The default is 11.

mgrd - Grid resolution parameter that limits the horizontal growth of a puff (INTEGER*4). The horizontal size of the puffs will be limited to $2^{mgrd} \times hres$, where *hres* is determined by the host grid. The default is 2.

grdmin - Minimum grid size (meters) for the puff grid resolution (REAL*4). The default is zero.

z_dosage - Elevation at which surface dosages are computed (REAL*4). Not used in the plume-in-grid version. The default is zero.

vres - Spacing parameter (meters) that limits the vertical growth of a puff (REAL*4). The default is 250 meters. However, for plume-in-grid applications, we recommend using higher values, since the vertical resolution of the host model in the upper layers is typically much coarser than 250 meters.

dynamic - Flag for including momentum and buoyancy dynamics (LOGICAL). If 'False' then puffs are treated as passive, and the Models-3/CMAQ ECIP algorithm is used for calculating plume rise (see Section 2.1). The default is 'True'. However, for plume-in-grid applications, we recommend using 'False' both for computational efficiency and for consistency with the host model treatment of plume rise.

Namelist: **MET** - meteorological parameters

ensm_type - Type of large-scale meteorological input, 'MODEL', 'OPER3.1' or 'NONE' (CHARACTER*80). No default.

uu_ensm - Velocity variance (m^2s^{-2}) for calculating the large-scale component of the dispersion (REAL*4). Not used in the plume-in-grid version. The default is 0.25.

sl_ensm - Length scale (meters) for calculating the meandering component of the dispersion (REAL*4). Not used in the plume-in-grid version. The default is 1.0×10^5.

sl_haz - Length scale (meters) for calculating dispersion due to wind uncertainty given as velocity variance on observational meteorology input files (REAL*4). Not used in the plume-in-grid version. The default is 1.0×10^5.

Figure 3-1 illustrates the **OPTIONS** and **MET** sections of a typical plume-in-grid control file. Not all of the namelist parameters are shown in the example, since either the default values of the other parameters are recommended, or the parameters are not relevant for a plume-in-grid application.

```
&OPTIONS
 LSPLITZ =  F,
 VRES = 500.,
 DYNAMIC = F,
 /
&MET
 ENSM_TYPE = 'OPER3.1 ',
/
```

Figure 3-1. Example **OPTIONS** and **MET** sections of the control file.

The **CONTROL** section of the multicomponent portion of the control file is described below. This is also a namelist section like the **OPTIONS** and **MET** sections. The section must begin with the line "#Control" (see Figure 3-2). In the stand-alone SCICHEM, this section consists of many parameters (EPRI, 2000b). However, not all these parameters are relevant for plume-in-grid applications. For example, this version of Models-3/CMAQ-APT does not treat gas-particle equilibrium and aqueous-phase chemistry, so the associated parameters for these processes are not required in the control file. In our description of the multicomponent control section, we do not discuss these parameters. Also, there are several parameters for which the default value is appropriate, or for which the user input is ignored since these parameters are directly related to parameters in the host model. An example of the former case is the variable "species_units" in SCICHEM which specifies the concentration units for the chemical species. In the host model these concentrations are carried in ppm and this is the default unit for "species_units" in SCICHEM. Examples of the latter case are the variables used to specify the units for the reaction rate constants. In the plume-in-grid version of SCICHEM, the rate constants are calculated using the host model parameters and algorithms (see Section 2.1). Thus, the values given to the "rate_species_units" and "rate_time_units" variables in the multicomponent control file are ignored. Another example is the "emission_units" variable used to specify the emission rate units of the emitted species. In the plume-in-grid version of SCICHEM, the emissions are directly read from the emission files described in Section 3.1.2, and are always assumed to be in moles/s, the same units used in the host model.

In addition to these differences between the plume-in-grid and stand-alone versions of SCICHEM, there are two parameters (described below) which are only relevant for plume-in-grid applications and are not described in the SCICHEM User's Guide (EPRI, 2000b).

For greater clarity, we only discuss those multicomponent control parameters that are directly relevant to the plume-in-grid version of SCICHEM.

Namelist: **CONTROL** - multicomponent parameters

solver - Specifies the numerical solver that will be used to solve the chemistry equations (CHARACTER*8). The available options are Young & Boris ("YNB") and LSODE ("LSODE"). The default is "YNB", which is the recommended solver for plume-in-grid applications for computational efficiency and also for consistency with the host model (see Section 2.1). Note that in a previous version of SCICHEM, a logical flag ("solve_ynb") was used to select the solver. This flag is now obsolete in both the stand-alone and plume-in-grid versions of SCICHEM, but the SCICHEM User's Guide (EPRI, 2000b) does not reflect this change.

rtol - Relative tolerance (REAL*4) used to solve chemical reaction equations. The default is $1x10^{-2}$.

dump_chem - Flag (LOGICAL) to specify whether a chemical maturity criteria should be used in addition to the physical criteria used to determine whether the puff contents are ready to be transferred to the host model concentrations (see Section 2.3 and Karamchandani et al., 2000c). The default is 'False'. Note that this parameter is not applicable to the stand-alone SCICHEM and is therefore not described in EPRI (2000b).

param_chem - Tolerance for the chemical maturity criteria (REAL*4). If the O_3 and O_x (O_3 + NO_2) in the plume are within this tolerance, then the chemical maturity criteria is satisfied. Used only if "dump_chem" is set to 'True'. The default value is 0.01, i.e., the O_3 and O_x have to be within 1% to satisfy the chemical maturity criteria. Like "dump_chem", this parameter is not applicable to the stand-alone SCICHEM and is therefore not described in EPRI (2000b).

staged_chem - Specifies whether the chemical mechanism in the Equation Section (see below) is staged or not (LOGICAL). Must have certain species defined in the Species Section. The default is 'False'.

voc_names(1) - Used only for staged chemistry. Names (CHARACTER*80) of volatile organic compounds that react with NO_3. No default.

phno3 - Used only with *staged_chem* set to 'True' (REAL*4). Production rate (% per hour) of HNO_3 due to the reaction of OH and NO_2 that indicates the stage should include acid formation (switch from stage 1 to stage 2). Used both day and night. The default is 0.1% per hour.

rno3 - Used only with *staged_chem* set to 'True' (REAL*4). Production rate (% per hour) of NO_3 due to the reaction of NO_2 and O_3 that indicates the stage should include acid formation (switch from stage 1 to stage 2). Used at night in conjunction with *cno3*. The default is 0.1% per hour.

cno3 - Used only with *staged_chem* set to 'True' and in conjunction with *pno3* and *rvocno2* (REAL*4). Concentration of NO_3 (ppm) above which the stage should include acid formation if *pno3* or *rvocno2* meet their criteria (switch from stage 1 to stage 2). Used only at night. The default is 10^{-8} ppm.

rho2o3 - Used only with *staged_chem* set to 'True' (REAL*4). Ratio (%) of the destruction of NO by HO_2 to the destruction of NO by O_3 above which the stage should include O_3 formation (switch from stage 2 to stage 3). Used day and night in conjunction with *co3*. The default is 0.1%.

co3 - Used only with *staged_chem* set to 'True' and in conjunction with *rho2o3* (REAL*4). Concentration (ppm) of O_3 above which the stage should include ozone formation if *rho2o3* meets its criteria (switch from stage 2 to stage 3). The default is 10^{-3} ppm.

rvocno2 - Used only with *staged_chem* set to 'True' (REAL*4). Ratio (%) of the destruction of NO_3 by VOCs to the destruction of NO_3 by NO_2 above which the stage should include O_3 formation (switch from stage 2 to stage 3). Used at night in conjunction with *cno3*. The default is 1%.

chem_split - Specifies whether the puffs should use a chemical criteria to split in order to resolve ozone "wings" (LOGICAL). Current split criteria: ozone concentrations are within 20% of the background and plume NO_x concentrations are within 5% of the background.

If chem_split is set to ‘True’, then *staged_chem* must also be set to ‘True’. The default is ‘False’.

Figure 3-2 demonstrates the **CONTROL** namelist section of a typical plume-in-grid control file. Not all of the namelist parameters are shown in the example, since either the default values of the other parameters are recommended, or the parameters are not relevant for a plume-in-grid application, as discussed earlier.

```
#Control
 &CONTROL
        rtol = 1.e-3
        solver = 'YNB'
        dump_chem = .true.
        param_chem = 0.01
/
```

Figure 3-2. Example **CONTROL** section of the control file.

The **SPECIES** section of the multicomponent portion of the control file is required and must begin with the line “#Species” (see Figure 3-3). In the stand-alone SCICHEM, the species may be entered in any order. However, in the plume-in-grid version, the ordering of the species must correspond to the ordering in the host model. Each line of the **SPECIES** section gives the species name, the species type (F, S, E or A), an unused real number, and the absolute tolerance (ppm). The species types are shown in Table 3-1.

Table 3-1. SCICHEM Species Types

F	Fast	This is relevant only when LSODE is selected as the chemistry solver (not recommended). Species concentrations change rapidly and species rate equations will be integrated using LSODE.
S	Slow	This is relevant only when LSODE is selected as the chemistry solver (not recommended). Species concentrations change slowly and will be integrated explicitly using a predictor-corrector scheme.
E	Equilibrium	Species concentrations are assumed to be at steady-state.
A	Ambient	Species concentrations do not change by the reaction.

Figure 3-3 shows the **SPECIES** section of the control file when the CB-IV mechanism is selected for Models-3/CMAQ.

```
#Species,Type,amb(NA),Tolerance
NO2         F             0.0000E+00          1.0000E-08
NO          F             0.0000E+00          1.0000E-08
O           F             0.0000E+00          1.0000E-12
O3          F             0.0000E+00          1.0000E-08
NO3         F             0.0000E+00          1.0000E-12
O1D         F             0.0000E+00          1.0000E-12
OH          F             0.0000E+00          1.0000E-12
HO2         F             0.0000E+00          1.0000E-12
N2O5        F             0.0000E+00          1.0000E-12
HNO3        F             0.0000E+00          1.0000E-08
HONO        F             0.0000E+00          1.0000E-08
PNA         F             0.0000E+00          1.0000E-08
H2O2        F             0.0000E+00          1.0000E-08
CO          F             0.0000E+00          1.0000E-08
FORM        F             0.0000E+00          1.0000E-08
ALD2        F             0.0000E+00          1.0000E-08
C2O3        F             0.0000E+00          1.0000E-12
XO2         F             0.0000E+00          1.0000E-12
PAN         F             0.0000E+00          1.0000E-08
PAR         F             0.0000E+00          1.0000E-08
XO2N        F             0.0000E+00          1.0000E-12
ROR         F             0.0000E+00          1.0000E-12
NTR         F             0.0000E+00          1.0000E-08
OLE         F             0.0000E+00          1.0000E-08
ETH         F             0.0000E+00          1.0000E-08
TOL         F             0.0000E+00          1.0000E-08
CRES        F             0.0000E+00          1.0000E-08
TO2         F             0.0000E+00          1.0000E-12
OPEN        F             0.0000E+00          1.0000E-08
CRO         F             0.0000E+00          1.0000E-12
XYL         F             0.0000E+00          1.0000E-08
MGLY        F             0.0000E+00          1.0000E-08
ISOP        F             0.0000E+00          1.0000E-08
ISPD        F             0.0000E+00          1.0000E-08
SO2         F             0.0000E+00          1.0000E-08
SULF        F             0.0000E+00          1.0000E-08
AIR         A             0.0000E+00          1.0000E-06
```

Figure 3-3. Example **SPECIES** section of the control file for the CB-IV mechanism implemented in Models-3/CMAQ.

The **BALANCE** section of the multicomponent portion of the control file is optional and must begin with the line "#Balance" (see Figure 3-4). The user may select this option to ensure that a perfect balance is maintained for the nitrogen species after the chemistry solver. Note that this option is also available in the stand-alone SCICHEM but the

SCICHEM User's Guide (EPRI, 2000b) does not describe this option. Each line of the **BALANCE** section gives the species name, followed by the number of nitrogen atoms in the species. Figure 3-4 shows the **BALANCE** section for the CB-IV mechanism in Models-3/CMAQ.

```
#Balance  : N
NO        1.0
NO2       1.0
PAN       1.0
HONO      1.0
HNO3      1.0
PNA       1.0
NO3       1.0
N2O5      2.0
NTR       1.0
```

Figure 3-4. Example **BALANCE** section of the control file for the CB-IV mechanism implemented in Models-3/CMAQ.

The **EQUATION** section of the multicomponent portion of the control file is also required and must begin with the line "#Equation" (see Figure 3-4). Note that every species that appears in the **EQUATION** section must also appear in the **SPECIES** section discussed previously. In the stand-alone SCICHEM, the equations may be entered in any order. However, in the plume-in-grid version, the ordering of the equations must correspond to the ordering in the host model.

Each line of the **EQUATION** section gives the equation number followed by the chemical reaction. Species names must be placed in square brackets (e.g., [NO]), and stoichiometric coefficients must be placed in parentheses, as in (2.0) [NO]. The maximum number of reactants is 2 and the maximum number of products is 10. The reaction is ended by a semi-colon, followed by the reaction rate coefficient type and rate constant data. However, this information is only relevant for the stand-alone version of SCICHEM, since the plume-in-grid version calculates the rate constants using the Models-3/CMAQ include files and algorithms. Thus, anything after the semi-colon is read but ignored in the plume-in-grid version. Also, since the rates of photolytic reactions are directly calculated using the Models-3/CMAQ look-up table (see Section

2.2), there is no need for a **TABLE** section in the plume-in-grid control file as compared to the stand-alone SCICHEM (EPRI, 2000b).

A reaction will be modeled as turbulent if a "t" is placed in front of the equation number. A correlation between the two reactants will be evaluated and used in determining the reaction rate. Note that if this option is used, and LSODE is selected as the chemical solver, then both reactants must be specified as "fast" in the **SPECIES** section.

Figure 3-5 shows the first 10 reactions in the **EQUATION** section of the control file. A complete sample control file is provided in the example dataset provided with this User's Guide.

```
#Equations -- from Models3
1 [NO2]  ->  [NO] +  [O]          ; 1 0 0
2 [O]    ->  [O3]                 ; 1 0 0
3 [O3]  +   [NO]   ->  [NO2]      ; 1 0 0
4 [O]   +  [NO2]   ->  [NO]       ; 1 0 0
5 [O]   +  [NO2]   ->  [NO3]      ; 1 0 0
6 [O]   +   [NO]   ->  [NO2]      ; 1 0 0
7 [O3]  +  [NO2]   ->  [NO3]      ; 1 0 0
8 [O3]           ->  [O]          ; 1 0 0
9 [O3]           ->  [O1D]        ; 1 0 0
10 [O1D]   ->  [O]                ; 1 0 0
```

Figure 3-5. Portion of the **EQUATION** section of the control file for the CB-IV mechanism implemented in Models-3/CMAQ.

3.1.2 Models-3/CMAQ-APT emission files

The emission files for a Models-3/CMAQ-APT simulation are currently prepared using processors developed by MCNC under EPRI sponsorship. These processors were developed specifically for converting UAM-format area and point source files to files used for MAQSIP-APT and are also applicable to Models-3/CMAQ-APT (with one minor modification, discussed below). Note that additional processors will have to be developed if UAM-format files are not available. This section briefly describes the steps required to prepare Models-3/CMAQ-APT emission files from UAM-format area and point source files.

File descriptions

Before describing the procedure for preparing the emission files, it is useful to review the information contained in the files. Three files are required for a plume-in-grid simulation. The first file is a three-dimensional time-dependent file in NetCDF format containing gridded emissions data for the chemical species. This file is always required by Models-3/CMAQ whether or not a plume-in-grid simulation is conducted. This file, referred to as "EMIS_1" in the Models-3 User's Guide (EPA, 1998), includes emissions from area sources as well as emissions from those point sources that are not being explicitly simulated by the embedded plume model. The SMOKE plume rise algorithm is used to allocate the emissions vertically based on stack characteristics and ambient meteorology. Note that SCICHEM does not read the "EMIS_1" file.

The remaining two files ("PING0" and "PING1") are required by SCICHEM. Both these files are also in NetCDF format. The "PING0" file is a one-dimensional file that contains time-independent point source information, such as stack location, stack height, stack diameter, and a stack identifier for the point sources that are selected for plume-in-grid processing. The "PING" file is also a one-dimensional file that contains time-dependent source information, such as emission rates, stack temperature, and stack velocity for the same point sources that are in the "PING0" file.

For the default case (i.e., no plume-in-grid), emissions from all point sources are included in the "EMIS-1" file and the "PING0" and "PING1" are not created by the MCNC processor. The latter two files are only created by the processor when the user selects at least one point source to be treated explicitly by the plume model (see below). In this case, the data for the selected point sources are included in the "PING0" and "PING" files, but not in the "EMIS-1" file.

Table 3-2 lists the information that is included in the "PING0" file for each point source. The contents of an example "PING" file are listed in Table 3-3. Note that the emitted species in the "PING" file correspond to the emitted species in the "EMIS-1" file and that species names are consistent with the chemical mechanism chosen for the host model.

For example, the species names shown in Table 3-3 correspond to the CB-IV mechanism. Also, the minor difference referred to previously between the MAQSIP-APT and Models-3/CMAQ-APT emissions processors is that the emission rates in the "PING" file for MAQSIP-APT are in moles/hr while they are in moles/s for Models-3/CMAQ-APT (for consistency with the emission units in the respective host models).

Table 3-2. Contents of PING0 File

VARIABLE NAME	DESCRIPTION	UNITS
LON	Longitude of source (this field is currently ignored in SCICHEM, since XLAM and YLAM are used to locate the source)	degrees
LAT	Latitude of source (this field is currently ignored in SCICHEM, since XLAM and YLAM are used to locate the source)	degrees
XLAM	X-coordinate of source (e.g., in Lambert Conformal Projection coordinates)	m
YLAM	Y-coordinate of source (e.g., in Lambert Conformal Projection coordinates)	m
STKHT	Stack height	m
STKDM	Stack diameter	m
LINE_NUM	Source identifier	N/A

Table 3-3. Contents of Example PING File for CB-IV Mechanism

VARIABLE NAME	DESCRIPTION	UNITS
STKTK	Exit stack temperature	K
STKVE	Exit stack velocity	m/s
NO	Stack NO emissions	moles/s
NO2	Stack NO2 emissions	moles/s

Table 3-3. Contents of Example PING File for CB-IV Mechanism (continued)

VARIABLE NAME	DESCRIPTION	UNITS
OLE	Stack OLE emissions	moles/s
PAR	Stack PAR emissions	moles/s
TOL	Stack TOL emissions	moles/s
XYL	Stack XYL emissions	moles/s
FORM	Stack FORM emissions	moles/s
ALD2	Stack ALD2 emissions	moles/s
ETH	Stack ETH emissions	moles/s
ISOP	Stack ISOP emissions	moles/s
CO	Stack CO emissions	moles/s
SO2	Stack SO2 emissions	moles/s

Preparation of emission files

The steps involved in converting UAM-ready area and point source files to Models-3/CMAQ-APT files using the processors developed by MCNC are listed below. These

processors and their associated files and scripts for a sample test case (application to NARSTO-Northeast) are included in the Models-3/CMAQ-APT distribution provided with this User's Guide.

1. Convert UAM ground-level emissions file into a two-dimensional NetCDF file for the Models-3/CMAQ domain and projection using the processor "**uamv2em2**". This processor performs any domain and projection conversions necessary (e.g., UTM to Lambert conformal projection). Note that it may be necessary to perform additional steps, such as combining UAM biogenic and anthropogenic ground-level emissions, before this step.

2. Read the UAM elevated point source emissions file with processor "**ascii2ping**" to create an ASCII plume-in-grid (PinG) index file which lists all the point sources, their location and a flag indicating whether a PinG treatment is to be applied for this source. Note that when the processor initially creates this file, the flag is "False" for all sources (i.e., no PinG treatment for any point source). The user is expected to manually edit the file to set the flag to "True" for all PinG sources (the flag value "F" should be replaced by "T").

3. Convert UAM elevated point source emissions file into a three-dimensional NetCDF file for the Models-3/CMAQ domain and projection using the processor "**uamv2pt3**". This processor performs any domain and projection conversions necessary (e.g., UTM to Lambert conformal projection) and performs plume rise calculations for all non-PinG sources (sources with the flag value "F" in the ASCII PinG index file created in Step 2) to estimate the vertical distribution of emissions for each hour. The meteorological data for the plume rise calculations are read from the Models-3/CMAQ NetCDF format meteorological files. If the PinG flag for all sources is "False", then no additional files are created. However, if the PinG flag for one or more sources is "True", then two additional files are created. These are the "PING0" and "PING" files described earlier and shown in Tables 3-2 and 3-3, respectively.

4. Combine the two-dimensional ground-level emissions NetCDF file created in Step 1 with the three-dimensional point source emissions NetCDF file created in Step 3 to

create a single three-dimensional NetCDF file ("EMIS-1", discussed previously) containing ground-level emissions and non-PinG point source emissions. This file merging is performed using the processor "**mrggrid**".

After the above steps have been performed, either 1 file or 3 files will be available for a Models-3/CMAQ simulation. For the case where all point sources are to be simulated with the host grid model, only the three-dimensional time-dependent emissions file ("EMIS-1") will be created. For the case where some point sources are to be explicitly simulated with the plume model, the two additional NetCDF point source files ("PING0" and "PING") will also be created.

As an optional step, it may be desirable to perform further processing of the "PING0" and "PING" file to combine stacks that are located in close proximity (e.g., a source with multiple stacks). If each stack is treated separately, then SCICHEM will take very small time steps because the puffs from these sources will be interacting with each other right away and cannot be treated as static puffs. This may cause the model to slow down considerably. We recommend that such sources be combined into one single source with an effective stack diameter, stack temperature, stack velocity, and stack height. The following constraints may be used to calculate the effective stack parameters:

- Conservation of flow
- Conservation of momentum
- Conservation of buoyancy

The above constraints result in three simultaneous equations for stack diameter, stack velocity and stack temperature:

Flow: $$\bar{v}\bar{D}^2 = \sum_{i=1}^{N} v_i D_i^2 \qquad (1)$$

Momentum: $$\frac{\bar{v}^2\bar{D}^2}{\bar{T}} = \sum_{i=1}^{N} \frac{v_i^2 D_i^2}{T_i} \qquad (2)$$

and

Buoyancy: $$\frac{\bar{v}\bar{D}^2\left(\bar{T}-T_A\right)}{\bar{T}}=\sum_{i=1}^{N}\frac{v_i D_i^2\left(T_i-T_A\right)}{T_i} \quad (3)$$

In the above equations, $\bar{D}$, $\bar{v}$, and $\bar{T}$ are the effective stack diameter, effective stack velocity and effective absolute stack temperature, respectively, *N* is the number of stacks that are being combined, and D_i, v_i and T_i are the stack diameter, stack velocity and absolute stack temperature for stack *I*, respectively. T_A is the absolute ambient temperature.

Once the effective stack diameter, velocity and temperature are computed, then the effective stack height can be calculated as the buoyancy-weighted average of the individual stack heights:

$$\bar{H}=\frac{\sum_{i=1}^{N}Buoy_i H_i}{\sum_{i=1}^{N}Buoy_i} \quad (4)$$

In Equation (4), the buoyancy term for each stack is calculated using Equation (3).

The emissions for the combined source are simply the sum of the emissions from the individual sources comprising the effective source.

Note that if there are significant temporal variations in the velocities and temperatures of the stacks being combined, then the resulting effective stack diameter and height will also have a temporal variation. However, these parameters are currently specified in the time-independent file ("PING0") and changes will be required to both the MCNC preprocessor and the Models-3/CMAQ-APT code to have these parameters specified in the time-dependent file ("PING"). One option is to use average stack parameters for the episode being simulated and apply the above equations to get average effective stack parameters for the combined source. Alternatively, the following approximations can be made:

$$\overline{D} = \sqrt{\sum_{i=1}^{N} D_i^2} \tag{5}$$

$$\overline{v} = \frac{\sum_{i=1}^{N} v_i D_i^2}{\overline{D}^2} \tag{6}$$

$$\overline{T} = \frac{\overline{v}^2 \overline{D}^2}{\sum_{i=1}^{N} \frac{v_i^2 D_i^2}{T_i}} \tag{7}$$

and

$$\overline{H} = \frac{\sum_{i=1}^{N} H_i}{N} \tag{8}$$

With these approximations, both $\overline{D}$ and $\overline{H}$ are time-independent stack parameters and the current processors and code will work.

3.2 Additional Output Files Created by Models-3/CMAQ-APT

In addition to the standard NetCDF files created in a typical Models-3/CMAQ gas-only simulation, a number of files are created by SCICHEM in a Models-3/CMAQ-APT simulation. Many of these files are identical to those produced by the stand-alone SCICHEM and are described in the SCICHEM User's Guide (EPRI, 2000b). The following files are always created (or updated for a restarted run, as described below) in a Models-3/CMAQ-APT simulation:

- The project file, named "**ProjectName.prj**", is a binary file containing project information, such as SCICHEM version number, turbulence parameters, puff parameters, grid information, control parameters, and multicomponent parameters. This file is required by many of the Models-3/CMAQ-APT postprocessors, described later in this User's Guide. This file is created for a new simulation. However, for a restarted run (e.g., a multi-day episode that is being simulated one day at a time), the file is required from the previous day's simulation (i.e., it is not recreated but read by SCICHEM and updated).

- The puff file, named "**ProjectName.puf**", is a binary file containing the complete puff data at time intervals (e.g., hourly) corresponding to the output time interval selected for Models-3/CMAQ together with boundary layer and continuous source information for restart purposes. Only data for active puffs (i.e., puffs that have not yet been dumped to the host model) are included in the file. The contents of this file are described in the SCICHEM User's Guide (EPRI, 2000b). Like the project file, this file is created for a new simulation, but the existing file is updated for restarted simulations. This file is required by the Models-3/CMAQ-APT postprocessor, "**merge_concs**", which adds the active puff information to the gridded host model concentrations (contained in the standard three-dimensional NetCDF concentration file created in the simulation) to create a new three-dimensional, time-dependent NetCDF file containing the combined puff and host model concentrations.

- The deposition file, named "**ProjectName.dep**", is a binary file containing the cumulative puff dry deposition data for the modeling grid at time intervals (e.g., hourly) corresponding to the output time interval selected for Models-3/CMAQ. The contents of this file are also described in the SCICHEM User's Guide (EPRI, 2000b). Like the project and puff files, this file is created for a new simulation, but the existing file is updated for restarted simulations. This file is required by the Models-3/CMAQ-APT postprocessor, "**merge_deps**", which adds the puff dry deposition data to the gridded host model dry deposition values (contained in the standard two-dimensional NetCDF deposition file created in the simulation) to create a new two-dimensional, time-dependent NetCDF file containing the combined puff and host model dry depositions.

- The tracer dump file, named "**ProjectName.dmp**", is a binary file containing the mass of tracer dumped (transferred to the host model grid) to each grid cell at every time step. Note that this file is not part of the stand-alone SCICHEM output since it is only relevant for plume-in-grid applications. Thus, this file is not described in the SCICHEM User's Guide (EPRI, 2000b). Like the other files discussed above, the dump file is created for a new simulation, but the existing file is updated for restarted simulations. The supplied postprocessor, "**read_dmp**" reads this file to produce three

NetCDF files that can be viewed with PAVE: a) a three-dimensional file containing basically the same information contained in the dump file; b) a two-dimensional file containing vertically aggregated tracer dump information for each grid column; and c) a two-dimensional file containing cumulative vertically aggregated tracer dump information for each grid column.

- The multicomponent diagnostics file, named "**ProjectName.dgn**", is a binary file containing puff diagnostic variables. The diagnostic information is reported as a cumulative sum over all puffs for each output time step. The contents of the diagnostic file are described in the SCICHEM User's Guide (EPRI, 2000b). Like the other files discussed above, the diagnostics file is created for a new simulation, but the existing file is updated for restarted simulations. However, this file is now obsolete for Models-3/CMAQ-APT, since the plume-in-grid diagnostics are now output to the optionally created NetCDF format process analysis files, described later in this section, allowing the user to view the plume-in-grid diagnostics with PAVE.

- Two ASCII files, "**ProjectName.log**" and "**ProjectName.err**" containing general information about the simulation, including error or warning messages (if any), as well as successful run completion messages. These files are standard SCICHEM files that can be viewed directly. No further processing is required for these files. Unlike the other output files discussed above, these files are created as new files in every simulation. Thus, for multi-day simulations in which one day is simulated at a time, these files must be removed or renamed prior to continuing the simulation for the following day.

In addition to the above files that are always created in a Models-3/CMAQ-APT simulation, the model can optionally create 7 NetCDF process analysis files containing the puff diagnostics. The user selects the option of creating these files by setting a flag in the run script. These files are all time-dependent one-dimensional files containing different diagnostic information for each chemical species for different processes at every output time step. Using PAVE, the temporal variation in this diagnostic information can

be viewed separately for each process, or the information for different processes can be combined to check mass balances.

Like the other NetCDF files produced in a standard CMAQ simulation, these diagnostic files are created for each simulation, regardless of whether the simulation is new or a restart (i.e., a continuation of a previous simulation). Thus, if a multi-day simulation is conducted one day at a time, then the 7 process analysis files are created separately for each day, and the files must be appropriately named in the run script. The 7 files are described below:

- The "**ACTIVE_DGN**" file, containing the total mass of each chemical species in all the active puffs (i.e., puffs that have not yet been transferred to the host model grid) at each output time step.

- The "**BDRY_DGN**" file, containing the total mass of each chemical species crossing the boundaries of the modeling domain at each output time step.

- The "**CHEM_DGN**" file, containing the total mass of each chemical species formed (or destroyed) by chemical conversion at each output time step.

- The "**DEPOS_DGN**" file, containing the total mass of each chemical species dry deposited to the surface at each output time step.

- The "**DUMP_DGN**" file, containing the total mass of each chemical species transferred to the host grid model at each output time step. This file also includes the total number of puffs transferred at each output time step.

- The "**EMIS_DGN**" file, containing the total mass of each chemical species emitted from the PinG sources at each output time step.

- The "**STATICS_DGN**" file, containing the total mass of each chemical species released from the end of the static puffs at each output time step. For the output time steps typically used in a Models-3/CMAQ simulation (1 hour), this mass is identical

to the amount emitted at each time step, so that the information contained in the "**STATICS_DGN**" file is the same as that in the "**EMIS_DGN**" file.

The number of variables in the above files are the number of chemical species carried by the host model plus a tracer species ("**TRAC"**) carried by SCICHEM. The names of the chemical species are consistent with the host model species names. In addition, the "**DUMP_DGN**" file includes an extra variable "**NDUMP**", which is the number of puffs transferred to the host model grid at each output time step.

4. BUILDING MODELS-3/CMAQ-APT AND POST-PROCESSORS

In this section, we describe the steps involved in building Models-3/CMAQ-APT and the associated post-processing utilities discussed in Section 3.2, using the NARSTO-Northeast 12km domain simulation as an example. The associated sample scripts and makefiles are included with this User's Guide and are not described in detail here, except that the portions of these scripts and makefiles that should be changed by the user, depending on the application, are highlighted.

4.1 Operating System and Compiler Considerations

This version of Models-3/CMAQ-APT is consistent with the August 2000 release of Models-3/CMAQ. EPA has primarily tested the August 2000 release on Solaris 7. However, AER has successfully applied both Models-3/CMAQ and Models-3/CMAQ-APT with the previous version of the operating system (Solaris 2.6).

Both the host grid model and the SCICHEM codes need to be compiled with Fortran 90. We have used both the Sun Workshop Compiler 5.0 (Fortran 90 2.0, December 1998 release) and the Sun Workshop Compiler 6.0 (Fortran 95 6.0, April 2000 release) to successfully compile and build Models-3/CMAQ and Models-3/CMAQ-APT. With the older compiler (Fortran 90 2.0), it is also necessary to link in the "libf77compat" library (available as part of the compiler package) to build Models-3/CMAQ and Models-3/CMAQ-APT.

4.2 File Organization

This section describes the recommended directory and file organization for a typical Models-3/CMAQ-APT build and application using the NARSTO-Northeast 12km grid simulation as an example. Note that it is not necessary for the user to strictly follow this structure, as long as the provided makefiles and run scripts are appropriately modified.

Figure 4-1 shows the directory organization for the NARSTO-Northeast 12km grid simulation. The main project directory is called "NARSTO_APT" and is located in the user login root directory. There are two subdirectories within this directory, for the

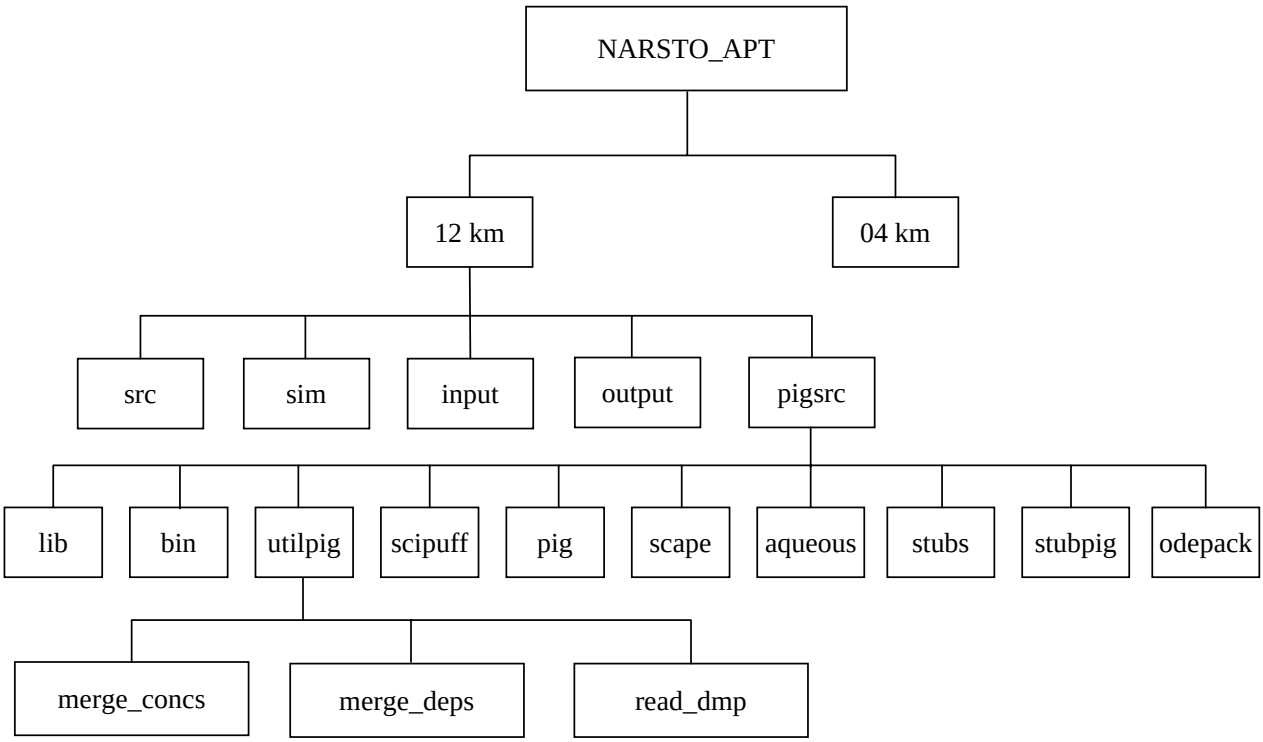

Figure 4-1. Directory structure for Models-3/CMAQ-APT for the NARSTO-Northeast application.

coarse and nested grid simulations, referred to as “12km” and “04km”, respectively. We will focus our discussion on the “12km” subdirectory since the steps involved in building Models-3/CMAQ-APT are similar for both grids.

Within the “12km” subdirectory, there are a number of subdirectories. For the time being, we will only discuss the subdirectories that are relevant to building the model and associated utilities. These subdirectories are “src” and “pigsrc”. The Models-3/CMAQ-APT executable is built in the “src” subdirectory, using the provided build script “bldit.narsto_cb4_yb_ping.12km”. This script is a modified version of the build script in the August 2000 release of Models-3/CMAQ, adapted for Models-3/CMAQ-APT (see Section 4.3).

The pigsrc subdirectory contains the following subdirectories that comprise the SCICHEM code: “scipuff”, “pig”, “scape”, “aqueous”, “stubs”, “stubpig”, and “odepack”. The SCICHEM codes in the first 6 directories are built as a single library (“libsci”), using the provided makefile (“makefile.scilib”) in directory “pigsrc”. The “odepack” directory contains the code for the LSODE solver, which is built as a separate library (“libodepack”) using the provided makefile (“makefile”) in directory “odepack”. These libraries are placed in subdirectory “lib” of “pigsrc”.

In addition, the codes for the post-processing utilities are contained in the “merge_concs”, “merge_deps”, and “read_dmp” subdirectories of the “utilpig” subdirectory of “pigsrc”. The associated makefiles for these utilities are also provided in directory “pigsrc”. Because these utilities make use of SCICHEM routines, it is necessary to build the “libsci” and “libodepack” libraries before building the utilities. The executables for the utilities are placed in the “bin” subdirectory of directory “pigsrc”.

4.2 Building Models-3/CMAQ-APT

As mentioned in the previous section, Models-3/CMAQ-APT is built in the “src” subdirectory of the project directory “12km”, using the provided build script. Note that the SCICHEM libraries, “libsci” and “libodepack”, must be built before Models-3/CMAQ-APT is built. It is possible for the user to first build the SCICHEM libraries

separately, using the provided makefiles in the "pigsrc" and "odepack" subdirectories. However, it is not necessary to do so since the Models-3/CMAQ-APT build script in the "src" directory builds the SCICHEM libraries if necessary. In this section, we discuss the sections of the build scripts and makefiles that need to be modified by Models-3/CMAQ-APT users for their applications.

Figure 4-2 shows a portion of the makefile for the "libodepack" library. The section of this makefile that may need to be modified by the user is shaded. This section defines the path for the Fortran 90 compiler. Note that additional modifications to the makefile may also be required if the user does not follow the directory configuration described in Section 4.2.

```
# This makefile will build ODEPACK on most UNIX systems.  To use it,
# name this file "makefile", and put it in the same directory as the
# ODEPACK source code, and simply type "make".
#
# Set the following path to the directory where you want the ODEPACK
# library to be installed.
#
BASEDIR     = ..
ODEPACK     = $(BASEDIR)/odepack
LIBDIR      = $(BASEDIR)/lib

#set path for f90 compiler
FC = /opt/6.0/SUNWspro/bin/f90
……….
……
```

Figure 4-2. Portion of the makefile for the LSODE library.

Figure 4-3 shows a portion of the makefile for the "libsci" library. The sections of this makefile that may need to be modified by the user are shaded. These sections define the the path for the Fortran 90 compiler, the base directory, and the locations of the Models-3 include files required by SCICHEM. Note that additional modifications may also be required if the user does not follow the directory configuration described in Section 4.2.

Note that the Models-3/CMAQ include directory, "ping_apt", shown as ICLDIR5 in Figure 4-3, is specific to Models-3/CMAQ-APT and is not part of the standard Models-3/CMAQ EPA release. This directory is provided as part of the Models-3/CMAQ-APT

```
# This makefile creates the SCICHEM library for Models-3/CMAQ

# Compiler
FTN = /opt/6.0/SUNWspro/bin/f90

# Directories

# HOME DIRECTORY
homedir  = /laplace2/cp080/NARSTO_APT/12km/pigsrc

# Library directory
LIBDIR      = $(homedir)/lib

# Models-3 include directories
M3MODELS  = /laplace1/models3/aug2000/models
Reso      = 12
Grid      = 150X129
Layers    = 13
Mechanism = cb4
Tracer    = trac0
Decomp    = 1X1

ICLDIR1 = $(M3MODELS)/include/release
ICLDIR2 = $(M3MODELS)/include/release/$(Mechanism)
ICLDIR3 = $(M3MODELS)/include/release/$(Tracer)
ICLDIR4 = $(M3MODELS)/include/release/d$(Reso)_$(Layers)
ICLDIR5 = $(M3MODELS)/include/release/ping_apt

ICLFLAGS = -I$(ICLDIR1) -I$(ICLDIR2) -I$(ICLDIR3) -I$(ICLDIR4) -
I$(ICLDIR5)

FLAGS8 = \
-DSUBST_FILES_ID=\'FILES_CTM.EXT\' \
-DSUBST_IODECL=\'IODECL3.EXT\' \
-DSUBST_IOFDESC=\'FDESC3.EXT\' \
-DSUBST_IOPARMS=\'PARMS3.EXT\' \
-DSUBST_XSTAT=\'XSTAT3.EXT\' \
-DSUBST_GC_SPC=\'GC_SPC.EXT\' \
-DSUBST_GC_DEPV=\'GC_DEPV.EXT\' \
-DSUBST_AE_SPC=\'AE_SPC.EXT\' \
-DSUBST_AE_DEPV=\'AE_DEPV.EXT\' \
-DSUBST_NR_SPC=\'NR_SPC.EXT\' \
-DSUBST_NR_DEPV=\'NR_DEPV.EXT\' \
-DSUBST_TR_SPC=\'TR_SPC.EXT\' \
-DSUBST_TR_DEPV=\'TR_DEPV.EXT\' \
-DSUBST_RXCMMN=\'RXCM.EXT\' \
-DSUBST_HGRD_ID=\'HGRD_${Grid}_${Decomp}.EXT\' \
-DSUBST_VGRD_ID=\'VGRD_${Layers}.EXT\' \
-DSUBST_FILES_APT=\'FILES_APT.diag.EXT\'
```

Figure 4-3. Portion of the makefile for the SCICHEM library.

distribution and should be installed by the user in the "release" subdirectory of the Models-3/CMAQ include directory. The "ping_apt" include directory contains a single file, "'FILES_APT.diag.EXT", required for the SCICHEM diagnostic or process analysis files described in Section 3.2.

The makefile for the SCICHEM library uses a script to compile the individual subroutines. This script ("make_obj_f90") is shown in Figure 4-4, and should be modified by the user to provide the correct pathname for the Fortran 90 compiler, as highlighted in the figure.

```
echo "  **** : f90 $2"
/opt/6.0/SUNWspro/bin/f90 $1 $2
```

Figure 4-4. The "make_obj_f90" script used by the SCICHEM library makefile.

Next, we describe the script to build Models-3/CMAQ-APT. To facilitate our discussion, we break up the script into various sections and discuss each section separately. Only those sections of the script are shown that either require user modifications or are relevant to Models-3/CMAQ-APT. The first section is shown in Figure 4-5. In this section, the path for the Fortran 90 compiler is defined, as well as environment variables to define the locations of the Models-3 directories. The shaded portions of Figure 4-5 need to be modified by the user. In addition, as shown in Figure 4-5, the user may elect to build the base Models-3/CMAQ, i.e., without plume-in-grid treatment, by activating the appropriate line for the variable "ping_run".

```
# ------- USER INPUT REQUIRED ----------------------------------------
#set path for f90 compiler and library
 set FC = /opt/6.0/SUNWspro/bin/f90

 set ping_run = 1     # use this line if this is a ping_apt run
#set ping_run =       # use this line if this is a base run

#set foll. two env. variables if new version of Models-3 is in
different dir.
setenv M3MODEL /laplace1/models3/aug2000/models
setenv M3TOOLS /laplace1/models3/aug2000/tools
# -------------------------------------------------------------------
```

Figure 4-5. Top section of the build script for Models-3/CMAQ-APT.

Figure 4-6 shows the second section of the build script. This section shows the building of the LSODE and SCICHEM libraries if the user wants to build Models-3/CMAQ with plume-in-grid treatment ("ping_run" = 1). It is not necessary for the user to make any changes to this section if the directory configuration is as described in Section 4.2.

```
# user choices: base directory
 set Base = $cwd

# -------- Build odepack and scichem libraries if this is APT run ----
if ( $ping_run ) then
 cd ../pigsrc
 if (-e lib/libodepack.a) then
    'rm' lib/libodepack.a
 endif
 cd odepack
 unset echo
 echo "Building odepack library (this could take upto 5 minutes)"
 make >& make.log
 if (-e ../lib/libodepack.a) then
    echo "odepack library successfully created"
 else
    echo "Error: odepack library not created"
    echo "Look at error messages in make.log in pigsrc/odepack"
    exit()
 endif
 cd ..
 if (-e lib/libsci.a) then
    'rm' lib/libsci.a
 endif
 echo "Building scichem library (this should take about 5-10 minutes)"
 make -f makefile.scilib >& make.scilib.log
 if (-e lib/libsci.a) then
    echo "scichem library successfully created"
 else
    echo "Error: scichem library not created"
    echo "Look at error messages in make.scilib.log in pigsrc"
    exit()
 endif

 set echo
 cd $Base
endif
# ------------------------------------------------------------------
```

Figure 4-6. Section of the build script for Models-3/CMAQ-APT showing the building of the LSODE and SCICHEM libraries.

Figure 4-7 shows the section of the build script that sets the name for the executable file depending on the value of the variable "ping_run". Because the name shown in the

figure is relevant to the NARSTO-Northeast application, the user may wish to modify the shaded portions of this section as appropriate.

```
# user choices: cvs archives
 set Project = $M3MODEL/CCTM
 set GlobInc = $M3MODEL/include/release

 if ( $ping_run ) then
    set APPL  = narsto_cb4_yb_ping_12km
 else
    set APPL  = narsto_cb4_yb_12km
 endif
 set CFG   = ${APPL}.cfg
 set MODEL = CCTM_$APPL
```

Figure 4-7. Section of the build script for Models-3/CMAQ-APT showing the project-specific name of the Models-3/CMAQ-APT executable file.

The next section of the build script pertains to the modules selected by the user. Figure 4-8 shows only the modules that are relevant to Models-3/CMAQ-APT. As discussed in previous sections, this version of Models-3/CMAQ-APT should be applied only for gas-phase simulations. Thus, the module options “aero_noop” and “aero_depv_noop” are selected for the aerosol module and the aerosol dry deposition modules, respectively. Similarly, the module option “cloud_noop” is selected for the cloud and aqueous-phase chemistry module. Also, for consistency with the chemistry solver used in SCICHEM, the Young and Boris solver should be used in the grid model. This solver is linked in by selecting the “chemeq” option for the chemistry module. Note that this module is not part of the standard Models-3/CMAQ release and is only provided with Models-3/CMAQ-APT. This module should be installed in the “chem” subdirectory of the Models-3 CCTM source archive.

Finally, the “ping” module selected is based on the value of the “ping_run” variable, discussed earlier in this section. For a Models-3/CMAQ-APT build, the module that is selected is “ping_apt”. This module is also not part of the standard Models-3/CMAQ release and is only provided with Models-3/CMAQ-APT. This module should be installed in the “ping” subdirectory of the Models-3 CCTM source archive.

```
# user choices: various modules

 set revision = release       # monocode (>="REL1_4")

 set ModDriver = ( module ctm            $revision; )
 ….
 ….

#set ModPing   = ( module ping_noop      $revision; )
#set ModPing   = ( module ping_qssa      $revision; )
#set ModPing   = ( module ping_smvgear   $revision; )
 if ( $ping_run ) then
    set ModPing   = ( module ping_apt $revision; )
 else
    set ModPing   = ( module ping_noop       $revision; )
 endif

#set ModChem   = ( module chem_noop      $revision; )
 set ModChem   = ( module chemeq         $revision; )
#set ModChem   = ( module qssa           $revision; )
#set ModChem   = ( module smvgear        $revision; )

 set ModAero   = ( module aero_noop      $revision; )
#set ModAero   = ( module aero           $revision; )

 set ModAdepv  = ( module aero_depv_noop $revision; )
#set ModAdepv  = ( module aero_depv      $revision; )

 set ModCloud  = ( module cloud_noop     $revision; )
#set ModCloud  = ( module cloud_radm     $revision; )
```

Figure 4-8. Section of the build script for Models-3/CMAQ-APT showing the selection of modules.

In the next section, shown in Figure 4-9, the user choices for the include files for the modeling domain and grid, the chemical mechanism, etc, are made. Note that these choices should be consistent with those made in the makefile for the SCICHEM library (see Figure 4-3). Depending on the application, the user may wish to make changes to the shaded portions of Figure 4-9. For example, for a horizontal resolution of 36 km, the variable “Reso” should be set to 36 (this assumes that the appropriate include file for this resolution is available in the Models-3 include directories).

```
# user choices: domain and mechanism
 set Reso      = 12
 set Grid      = 150X129
 set Layers    = 13
 set Mechanism = cb4
 set Tracer    = trac0              # default: no tracer species
 set Decomp    = 1X1
 set PABase    = $GlobInc
 set PAOpt     = pa_noop
 set APT       = ping_apt
```

Figure 4-9. Section of the build script for Models-3/CMAQ-APT showing the selection of Models-3 include directories.

Figure 4-10 shows the next section, in which the libraries that need to be linked in with the Models-3/CMAQ executable. The user need not make any changes to this section unless the file organization is different from that discussed in Section 4.2. The value of the variable "ping_run" is used to determine if the SCICHEM and LSODE libraries need to be linked. If the value of "ping_run" is 1, then these libraries need to be linked in to build Models-3/CMAQ-APT. Otherwise, these libraries are not linked, and Models-3/CMAQ is built.

```
 set LIB1 = "-L${M3TOOLS}/IOAPI/release/m3io/lib/SunOS5 -lm3iof90"
 set LIB2 = "-L${M3TOOLS}/netCDF/SunOS5 -lnetcdf"
 set LIB3 = "-L${M3TOOLS}/stenex/SunOS5 -lse_noop"
 if ( $ping_run ) then
    setenv SCIDIR  ${Base}/../pigsrc/lib
    set SCILIB = "-L${SCIDIR} -lsci -lodepack"
 else
    set SCILIB  =
 endif
 set LIBS = "$LIB1 $LIB2 $LIB3 $FCLIB $SCILIB"
```

Figure 4-10. Section of the build script for Models-3/CMAQ-APT showing the libraries that are linked in to build the executable.

Figure 4-11 shows the section of the build script in which the include directories are defined based on the selections made in Figure 4-9. The user does not need to modify this section. The main thing to note in this section is the definition of the "ping_apt" directory in the "ICL_APT" variable. This is specific to Models-3/CMAQ-APT and is not part of the standard Models-3 release.

```
 set Domain    = d${Reso}_${Layers}
 set ICL_BLOCK = $GlobInc
 set ICL_CGRID = $GlobInc
 set ICL_CONST = $GlobInc
 set ICL_FILES = $GlobInc
 set ICL_EMCTL = $GlobInc
 set ICL_IOAPI = $GlobInc
 set ICL_PAR   = $GlobInc
 set ICL_GRID  = $GlobInc/$Domain
 set ICL_MECH  = $GlobInc/$Mechanism
 set ICL_TRAC  = $GlobInc/$Tracer
 set ICL_PA    = $PABase/$PAOpt
 set ICL_APT   = $GlobInc/$APT
```

Figure 4-11. Section of the build script for Models-3/CMAQ-APT showing the definition of the Models-3/CMAQ include directories based on the selections made in Figure 4-9.

Finally, Figure 4-12 shows the section in which the substitution of include file names is made by the preprocessor before the Models-3/CMAQ-APT source code is compiled and built. There is no need for the user to modify this section. Only a portion of this section, showing the substitution of the Models-3/CMAQ-APT-specific include file, "'FILES_APT.diag.EXT" (required for the SCICHEM diagnostic or process analysis files creation), is shown in Figure 4-12 (also see Figure 4-3, where the makefile for the SCICHEM library uses this include file).

```
 include SUBST_PE_COMM     $ICL_PAR/PE_COMM.EXT;
 include SUBST_CGRID_DECL  $ICL_CGRID/CGRID_DECL.EXT;
 …….
 …….
 …….
 …….

// APT includes for diagnostics
 include SUBST_FILES_APT   $ICL_APT/FILES_APT.diag.EXT;
```

Figure 4-12. Section of the build script for Models-3/CMAQ-APT showing the preprocessor substitution of Models-3 include files, including the APT-specific include file, "'FILES_APT.diag.EXT".

4.4 Building the Models-3/CMAQ-APT Post-Processing Utilities

The three Models-3/CMAQ-APT post-processing utilities, "**merge_concs**", "**merge_deps**", and "**read_dmp**", described in Section 3.2, are built using the makefiles provided in directory "pigsrc" (see Figure 4-1). In this section, we briefly discuss these makefiles, highlighting sections that may need to be modified by users based on their specific application of Models-3/CMAQ-APT. As mentioned previously in Section 4.3, it is necessary for the SCICHEM libraries to be built before building the post-processors.

Figure 4-13 shows a portion of the makefile for the "**merge_concs**" post-processor. The user will need to modify some of the shaded sections, as appropriate. Additional modifications may also be required if the user does not follow the directory structure described in Section 4.2. The modifications involve the path to the Fortran 90 compiler, the definition of the home (or base) directory, the location of the Models-3 archive, and the selection of the Models-3 include directories. All these variables should be consistent with the SCICHEM library makefile, discussed in the previous section, and shown in Figure 4-3. In addition, it is also necessary to specify the locations of the I/O API and NetCDF libraries.

Figure 4-14 shows a portion of the makefile for the "**merge_deps**" post-processor. The sections that may need modifications are highlighted and are similar to those for the "**merge_concs**" post-processor. Figure 4-15 shows a portion of the makefile for the "**read_dmp**" post-processor. In this makefile, the only modifications that are required, assuming that the user follows the directory configuration described in Section 4.2, are to the path names for the Fortran 90 compiler, the definition of the home directory, and the locations of the I/O API and NetCDF libraries.

```
# This makefile creates the merge_concs executable
# Use this makefile only after the libsci.a has been created by
makefile.scilib

# Compiler
FTN = /opt/6.0/SUNWspro/bin/f90

# Directories

# HOME DIRECTORY
homedir  = /laplace2/cp080/NARSTO_APT/12km/pigsrc

# Library directory
LIBDIR      = $(homedir)/lib

# Models-3 include directories
M3MODELS  = /laplace1/models3/aug2000/models
Reso      = 12
Grid      = 150X129
Layers    = 13
Mechanism = cb4
Tracer    = trac0
Decomp    = 1X1

ICLDIR1 = $(M3MODELS)/include/release
ICLDIR2 = $(M3MODELS)/include/release/$(Mechanism)
ICLDIR3 = $(M3MODELS)/include/release/$(Tracer)
…….
…….
…..

# IOAPI LIBRARIES
M3TOOLS = /laplace1/models3/aug2000/tools
libIOAPI = ${M3TOOLS}/IOAPI/release/m3io/lib/SunOS5
# netCDF LIBRARIES
libnetCDF = ${M3TOOLS}/netCDF/SunOS5
…….
…….
…..
```

Figure 4-13. Portion of the makefile for the “merge_concs” post-processor.

```
# This makefile creates the merge_deps executable
# Use this makefile only after the libsci.a has been created by
makefile.scilib

# Compiler
FTN = /opt/6.0/SUNWspro/bin/f90

# Directories

# HOME DIRECTORY
homedir  = /laplace2/cp080/NARSTO_APT/12km/pigsrc

# Library directory
LIBDIR      = $(homedir)/lib

# Models-3 include directories
M3MODELS  = /laplace1/models3/aug2000/models
Reso      = 12
Grid      = 150X129
Layers    = 13
Mechanism = cb4
Tracer    = trac0
Decomp    = 1X1

ICLDIR1 = $(M3MODELS)/include/release
ICLDIR2 = $(M3MODELS)/include/release/$(Mechanism)
ICLDIR3 = $(M3MODELS)/include/release/$(Tracer)

…….
…….
….

# IOAPI LIBRARIES
M3TOOLS = /laplace1/models3/aug2000/tools
libIOAPI = ${M3TOOLS}/IOAPI/release/m3io/lib/SunOS5
# netCDF LIBRARIES
libnetCDF = ${M3TOOLS}/netCDF/SunOS5
…….
…….
….
```

Figure 4-14. Portion of the makefile for the “merge_deps” post-processor.

```
# This makefile creates the read_dmp executable
# Use this makefile only after the libsci.a has been created by
Makefile.scilib

# Compiler
FTN = /opt/6.0/SUNWspro/bin/f90

# Directories

# HOME DIRECTORY
homedir  = /laplace2/cp080/NARSTO_APT/12km/pigsrc

…….
…….
…..

# IOAPI LIBRARIES
M3TOOLS = /laplace1/models3/aug2000/tools
libIOAPI = ${M3TOOLS}/IOAPI/release/m3io/lib/SunOS5
# netCDF LIBRARIES
libnetCDF = ${M3TOOLS}/netCDF/SunOS5

…….
…….
…..
```

Figure 4-15. Portion of the makefile for the “read_dmp” post-processor.

5. CONDUCTING A MODELS-3/CMAQ-APT SIMULATION

Once Models-3/CMAQ-APT is built as described in Section 4, the basic steps in conducting a simulation are:

- Preparation of input files
- Preparation of run scripts
- Running the model
- Post-processing the output files for subsequent analysis

Section 3 described the input files that are required for a Models-3/CMAQ-APT in addition to the standard files required for a base Models-3/CMAQ application. In this section, we will focus on the preparation of the run scripts for applying the model and post-processing. We will also briefly discuss the additional steps required to conduct a nested grid simulation. We will again use the NARSTO-Northeast simulation as a demonstration.

5.1 File Organization

The run scripts demonstrated in this section assume that the directories and files are organized in a certain manner. The user will need to make the appropriate modifications to the run scripts if the files are organized differently.

Figure 5-1 shows the directory organization for the NARSTO-Northeast 12 km grid simulation. The main project directory is called "NARSTO_APT" and is located in the user login root directory. There are two subdirectories within this directory, for the coarse and nested grid simulations, referred to as "12km" and "04km", respectively. In Section 4.2 and Figure 4-1, we discussed the subdirectories of the "12km" directory that are relevant to building the SCICHEM libraries and the grid model. Here, we discuss the subdirectories that are relevant to applying the model. While the run script and file organization for the nested grid "04km" simulation will be similar, there are some

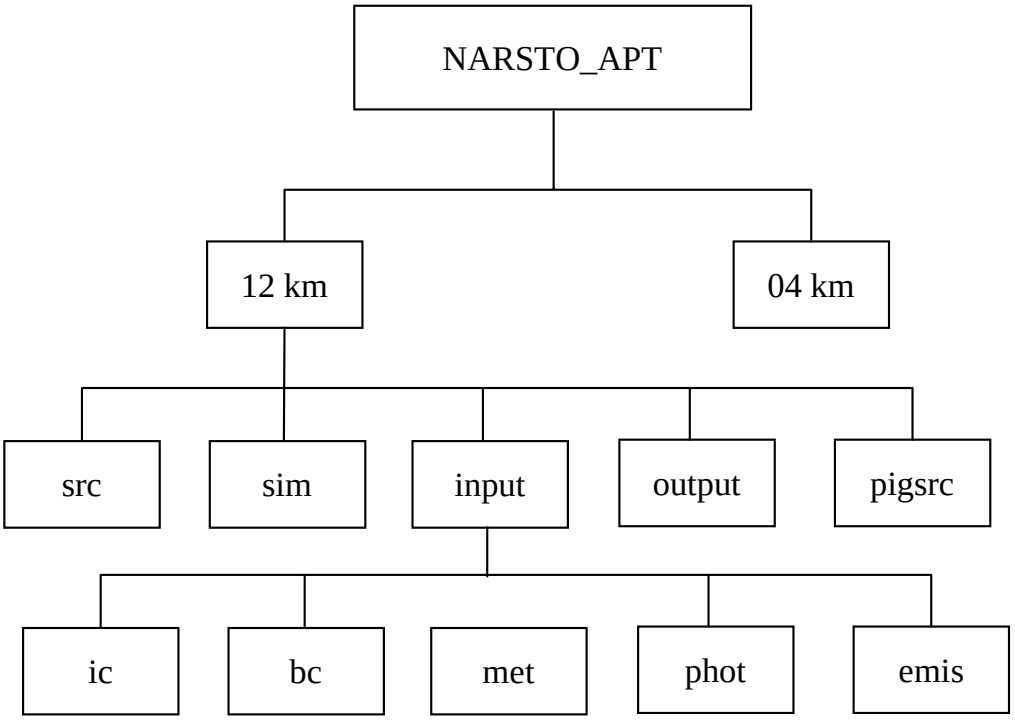

Figure 5-1. Directory structure for Models-3/CMAQ-APT for the NARSTO-Northeast application.

additional considerations involved in conducting a nested grid simulation and they will be discussed later, in Section 5.4.

The control file, "**ProjName.inp**" for the APT simulation (see Section 3.1.1) is located in the "**input**" subdirectory. For the NARSTO-Northeast application, the name of the project (specified in the run script) is "**narsto**", so the input file name is "**narsto.inp**". All the other input files required for the simulation are located in various subdirectories of the "input" subdirectory. These subdirectories are "**ic**" (initial condition files), "**bc**" (boundary condition files), "**met**" (meteorology and related files), "**phot**" (photolysis rate look-up tables), and "**emis**" (emission files).

The run scripts are located in the "**sim**" subdirectory. The output files from the simulation are created in the "**output**" subdirectory. The additional output files created by SCICHEM are described in Section 3.2. For the NARSTO-Northeast application, the names of these files are "**narsto.prj**", "**narsto.puf**", etc.

5.2 Models-3/CMAQ-APT Run Scripts

Depending on the modeling domain, Models-3/CMAQ can simulate a multi-day episode in multiple simulations (e.g., one day at a time) or in one continuous simulation. For large domains, it is necessary to split the simulations into smaller periods because of file size constraints. For the NARSTO-Northeast 12 km application, the domain was large (150 x 129 x 13 grid cells), and a separate simulation was conducted for each day. Thus, multiple run scripts were required, one for each day of simulation. In this section, we examine two run scripts, one for the first day of simulation (new run), and one for the second day of simulation (restart run). The scripts for subsequent days are similar to those for the second day. For a continuous simulation, only one script (the new run script) is required.

5.2.1 Run script for the first day of simulation

Figure 5-2 shows selected sections of the run script for the first day of the episode, July 11, 1995. Models-3/CMAQ users will be familiar with most aspects of this run script, so

```
#! /bin/csh -f
……..
……..
#set APPL     = narsto_cb4_yb_12km   # base model
set APPL     = narsto_cb4_yb_ping_12km  # plume-in-grid version
set CFG      = $APPL
set EXEC     = CCTM_$CFG       # ctm version
……..
……..
# timestep run parameters
set STDATE   = 1995192       # beginning date
set STTIME   = 050000        # beginning hour (GMT)
set NSTEPS   = 240000        # time duration (HHMMSS) for this run
                             # so last time = 050000 on 1995193
set TSTEP    = 010000        # output time step interval (HHMMSS)
set GDATE    = 950711        # calendar date to identify files
set runid      = 18
set runid12km  = ${runid}a
……..
……..
# set run parameters and flags required by APT
# create APT NetCDF diagnostic files? [ T | Y | F | N ]
setenv APT_DIAG Y
# restarted APT run? [ T | Y | F | N ]
setenv CTM_RESTART F
# project name
setenv CTM_PROJNAME narsto
……..
……..
# output files and directories
set OUTDIR1  = $HOMEDIR/output
if ( ! -d "$OUTDIR1" ) mkdir -p $OUTDIR1
set OUTDIR2  = $HOMEDIR/output
if ( ! -d "$OUTDIR2" ) mkdir -p $OUTDIR2
set CONCfile = "CONC".$APPL.${GDATE}.run${runid12km}   # CTM_CONC_1
set DD1file  = "DRYDEP".$APPL.${GDATE}.run${runid12km} # CTM_DRY_DEP_1
……..
……..
# APT output diagnostic files
set EMIS_DGN    =   "${OUTDIR}/EMIS_DGN.${GDATE}.run${runid12km} -v"
set STATICS_DGN =   "${OUTDIR}/STATICS_DGN.${GDATE}.run${runid12km} -v"
set BDRY_DGN    =   "${OUTDIR}/BDRY_DGN.${GDATE}.run${runid12km} -v"
set DEPOS_DGN   =   "${OUTDIR}/DEPOS_DGN.${GDATE}.run${runid12km} -v"
set CHEM_DGN    =   "${OUTDIR}/CHEM_DGN.${GDATE}.run${runid12km} -v"
set ACTIVE_DGN  =   "${OUTDIR}/ACTIVE_DGN.${GDATE}.run${runid12km} -v"
set DUMP_DGN    =   "${OUTDIR}/DUMP_DGN.${GDATE}.run${runid12km} -v"
……..
……..
# input directory for SCICHEM projectname.inp file
setenv INDIR ../input
# output directory for SCICHEM projectname.* output files
setenv OUTDIR $OUTDIR1
```

Figure 5-2. Selected sections of the run script for the first day of simulation.

```
set EMISpath  = $HOMEDIR/input/emis
set EMISfile  = narsto.emis3d.no58src.${GDATE}.12km
#ping_apt files
set PINGdir  = $HOMEDIR/input/emis
setenv PING0 $PINGdir/narsto.ping0_combined30.12km
setenv PING  $PINGdir/narsto.ping_combined30.${GDATE}.12km
……..
……..
set GC_ICpath = $HOMEDIR/input/ic
set GC_ICfile = icon_narsto_12_13lyr.${GDATE}.cb4
……..
……..
```

Figure 5-2. Selected sections of the run script for the first day of simulation (continued).

we will only focus on those aspects of the script that are relevant to Models-3/CMAQ-APT. These are highlighted in Figure 5-2.

In the first section of the run script shown in Figure 5-2, the executable for Models-3/CMAQ-APT ("narsto_cb4_yb_ping_12km") is selected in the variable "APPL". The selection of the base Models-3/CMAQ ("narsto_cb4_yb_12km") is also shown but is commented out.

The next shaded section shows the three environment variables ("APT_DIAG", "CTM_RESTART" and "CTM_PROJNAME") that need to be set for a Models-3/CMAQ-APT simulation. The first variable is a logical flag indicating if the NetCDF format plume-in-grid diagnostic or process analysis files should be created (i.e., APT_DIAG = "T" or "Y") or not (APT_DIAG = "F" or "N"). The second variable is a logical flag indicating whether the run is a new run (i.e., CTM_RESTART = "F" or "N") or a continuation run (CTM_RESTART = "T" or "Y"). The third variable is the name of the project ("narsto" in this example).

If APT_DIAG is selected as "True" then it is necessary to specify the pathnames for the NetCDF format diagnostic output files (see Section 3.2 for a description of these files), as shown in the next shaded section in Figure 5-2. Note that the file names include the "GDATE" identifier to distinguish them from files created from simulations of subsequent days. This identifier is also used to distinguish input files and other standard Models-3/CMAQ output files as discussed later below.

Next, the environment variables for the directory ("INDIR") in which SCICHEM will search for the "narsto.inp" file, and the directory ("OUTDIR") in which the remaining SCICHEM output files (e.g., "narsto.prj", "narsto.puf", etc.) will be created are set.

The environment variables, "PING0" and "PING" are the pathnames to the point source emission files required for a plume-in-grid simulation (see Section 3.1.2). The variable EMISfile points to the three-dimensional emissions file for the remaining sources (recall that for a non-PinG simulation, the "PING0" and "PING" files are not required and all emissions are included in the three-dimensional emissions file).

The rest of the run script is similar to a standard Models-3/CMAQ run script and is not discussed here. We now turn our attention to the run script for the next day's simulation for the case when a multi-day episode is simulated one day at a time, such as the NARSTO-Northeast application.

5.2.2 Run script for a restarted simulation

Before discussing the differences between the run scripts for a new run and a restarted run, it is useful to review the general procedure for simulating an episode in multiple simulations.

Each time-dependent input file (emissions, boundary conditions, meteorology) should have an extra hour of data for multi-day simulations. For example, the first day's input files should have 25 hours of data, consisting of 24 hours of data for that day and the first hour of the second day. This is applicable to both Models-3/CMAQ and Models-3/CMAQ-APT and is required because the model performs temporal interpolation of input fields between time steps.

The standard Models-3/CMAQ output files and the plume-in-grid diagnostic files should either be moved to a different directory for each day of simulation or named appropriately for each day. In the example run scripts provided with this User's Guide and shown in Figure 5-2 above and Figure 5-3 below, the latter option is selected by adding the variable "GDATE" to the file name. The three-dimensional concentration file

created from the first day's simulation (shown as the variable "CONCfile" in Figure 5-2) serves as the initial conditions file for the subsequent day's restart run.

The remaining SCICHEM binary output files ("narsto.prj", "narsto.puf", "narsto.dep", "narsto.dgn", and "nartso.dmp") should not be moved or renamed, since they are updated for subsequent simulation days, as described in Section 3.2. The SCICHEM ASCII output files ("narsto.log" and "narsto.err") can either be renamed or deleted, since a new file is created for each day of simulation.

Figure 5-3 shows the relevant sections of the run script for the second day's simulation. Only the differences between this run script and the run script for the previous day are highlighted. The variables "STDATE" and "GDATE" are set to the current day (i.e., second day). However, the variables, "GDATE_PREV" and "runidprev", are added to identify the previous day's output concentration file as the current day's initial condition file. The restart flag, "CTM_RESTART" is set to "True" since this is a restarted run. Finally, the path name for the initial condition file is set to the output concentration file from the previous day's simulation in variables "GC_ICpath" and "GC_ICfile".

The run scripts for subsequent days are similar with appropriate values assigned to the "STDATE", "GDATE", "GDATE_PREV", "runid12km" and "runidprev" variables.

5.3 Run Scripts for Post-Processing Models-3/CMAQ-APT Output Files

The following utilities are used to post-process the output files from a Models-3/CMAQ-APT simulation:

- **merge_concs**
- **merge_deps**
- **read_dmp**

These post-processors have been discussed earlier in Section 3.2. The "**merge_concs**" utility adds the active puff information in the SCICHEM output file "projectname.puf"

```
#! /bin/csh -f
……..
……..
# timestep run parameters
set STDATE   = 1995193       # beginning date
set STTIME   = 050000        # beginning hour (GMT)
set NSTEPS   = 240000        # time duration (HHMMSS) for this run
                             # so last time = 050000 on next day
set TSTEP    = 010000        # output time step interval (HHMMSS)
set GDATE    = 950712
set GDATE_PREV = 950711
set runid      = 18
set runid12km  = ${runid}b
set runidprev  = ${runid}a
……..
……..
# set run parameters and flags required by APT
# create APT NetCDF diagnostic files? [ T | Y | F | N ]
setenv APT_DIAG Y
# restarted APT run? [ T | Y | F | N ]
setenv CTM_RESTART T
# project name
setenv CTM_PROJNAME narsto
……..
……..
# output files and directories
set OUTDIR1  = $HOMEDIR/output
if ( ! -d "$OUTDIR1" ) mkdir -p $OUTDIR1
set OUTDIR2  = $HOMEDIR/output
if ( ! -d "$OUTDIR2" ) mkdir -p $OUTDIR2
set CONCfile = "CONC".$APPL.${GDATE}.run${runid12km}   # CTM_CONC_1
set DD1file  = "DRYDEP".$APPL.${GDATE}.run${runid12km} # CTM_DRY_DEP_1
……..
……..
# APT output diagnostic files
set EMIS_DGN    =   "${OUTDIR}/EMIS_DGN.${GDATE}.run${runid12km} -v"
set STATICS_DGN =   "${OUTDIR}/STATICS_DGN.${GDATE}.run${runid12km} -v"
set BDRY_DGN    =   "${OUTDIR}/BDRY_DGN.${GDATE}.run${runid12km} -v"
set DEPOS_DGN   =   "${OUTDIR}/DEPOS_DGN.${GDATE}.run${runid12km} -v"
set CHEM_DGN    =   "${OUTDIR}/CHEM_DGN.${GDATE}.run${runid12km} -v"
set ACTIVE_DGN  =   "${OUTDIR}/ACTIVE_DGN.${GDATE}.run${runid12km} -v"
set DUMP_DGN    =   "${OUTDIR}/DUMP_DGN.${GDATE}.run${runid12km} -v"
……..
……..
# input directory for SCICHEM projectname.inp file
setenv INDIR ../input
# output directory for SCICHEM projectname.* output files
setenv OUTDIR $OUTDIR1

set EMISpath  = $HOMEDIR/proc/emis/cnvt_m3/output
set EMISfile  = narsto.emis3d.no58src.${GDATE}.12km
#ping_apt files
set PINGdir  = $HOMEDIR/proc/emis/combine_pingsrc/output
```

Figure 5-3. Selected sections of the run script for the second day of simulation.

```
setenv PING0 $PINGdir/narsto.ping0_combined30.12km
setenv PING  $PINGdir/narsto.ping_combined30.${GDATE}.12km
……..
……..
#set GC_ICpath = $HOMEDIR/input/ic
set GC_ICpath = $HOMEDIR/output
#set GC_ICfile = icon_narsto_12_13lyr.${GDATE}.cb4
set GC_ICfile = CONC.${APPL}.${GDATE_PREV}.run${runidprev}
……..
……..
```

Figure 5-3. Selected sections of the run script for the second day of simulation (continued).

("narsto.puf" for the NARSTO-Northeast example) to the gridded host model concentrations in the three-dimensional NetCDF output file ("CONCfile" in Figures 5-2 and 5-3) produced in the simulation to create a new three-dimensional, time-dependent NetCDF file containing the combined puff and host model concentrations.

Figure 5-4 shows the run script for the "merge_concs" utility for merging the first day's results (July 11, 1995). The run script assumes that the post-processing is being performed in the "output" subdirectory (see Figure 5-1) and that the "merge_concs" program is located in the "bin" subdirectory of directory "pigsrc" (see Figure 4-1 and Section 4.2). The variables that need to be set in the run script are discussed below.

The variable "PUFFILE" is a pointer to the "projectname.puf" file created in the Models-3/CMAQ-APT simulation. The variables "STDATE" and "STTIME" are the starting date and time for the entire multi-day simulation (5:00 GMT on July 11, 1995 for the NARSTO-Northeast application). The variables "JDATE" and "JTIME" are the merge starting date and time. For the first day of simulation, these variables are the same as STDATE and STTIME. For the second and subsequent days, these variables will change appropriately, but the variables "STDATE" and "STTIME" will stay the same. Similarly, the variables "JEDATE" and "JETIME" are the merge ending date and time. The variable "OUTTS" is the output time step which is the same as the variable "TSTEP" in the Models-3/CMAQ-APT run scripts shown in Figures 5-2 and 5-3. The variables "runid" and "GDATE" are also the same as the variables "runid12km" and "GDATE" in the Models-3/CMAQ-APT run scripts.

```
#! /bin/csh -f
# C-shell script to merge Models-3 conc. file with scichem puff file
# Set up for the NARSTO domain

date; set timestamp; set echo

# These are the parameters for this run, set them carefully

set PUFFILE  = 'narsto.puf' # name of SCICHEM puff file
                            # ("projectname".puf)
# next 2 lines always stay the same for NARSTO 12km run
set STDATE   = 1995192      # simulation start date: year & Julian date
                            # (YYYDDD)
set STTIME   = 50000        # simulation start hour in GMT (HHMMSS)
set JDATE    = 1995192      # merge start date: year & Julian date
                            # (YYYDDD)
set JTIME    = 50000        # merge start hour in GMT      (HHMMSS)
set JEDATE   = 1995193      # merge end date: year & Julian date
                            # (YYYDDD)
set JETIME   = 50000        # merge end hour in GMT        (HHMMSS)
set OUTTS    = 10000        # output time step             (HHMMSS)

set runid    = 18a

set GDATE    = 950711       # calendar date
set METGRID  = G2           # grid for met domain (12km)

#################################################################
#     Set up script base and executable directories        #
#################################################################

# Need user input
set BASE     = .
set EXEDIR   = ../pigsrc/bin
set EXENAME  = merge_concs      # executable file name

#################################################################
#     Set up input files and directories             #
#################################################################

set METDIR    = ../input/met
set PHOTDIR   = ../input/phot

set GC2_G0   = GRIDCRO2D_NARSTO_${METGRID}.$GDATE   # GRID_CRO_2D_G0
set GD2_G0   = GRIDDOT2D_NARSTO_${METGRID}.$GDATE   # GRID_DOT_2D_G0
set MC2_G0   = METCRO2D_NARSTO_${METGRID}.$GDATE    # MET_CRO_2D_G0
set MC3_G0   = METCRO3D_NARSTO_${METGRID}.$GDATE    # MET_CRO_3D_G0
set MD3_G0   = METDOT3D_NARSTO_${METGRID}.$GDATE    # MET_DOT_3D_G0
set XJ_DATA  = JTABLE_$JDATE                        # XJ_DATA
```

Figure 5-4. Portion of the run script for the “merge_concs” utility for the first day of simulation.

```
################################################################
#     Set up conc files and directories                        #
################################################################

set CONCDIR = .
set CC3_G0  = CONC. narsto_cb4_yb_ping_12km.${GDATE}.run${runid}
set CC3_OUT = CONC_MERGED.narsto_cb4_yb_ping_12km.${GDATE}.run${runid}
……..
……..
```

Figure 5-4. Portion of the run script for the "merge_concs" utility for the first day of simulation (continued).

The variables "EXEDIR" and "EXENAME" provide the pathname to the "merge_concs" utility while the variables "METDIR" and "PHOTDIR" provide the locations of the Models-3/CMAQ meteorological and photolysis rate input files, respectively. The file variables ("GC2_G0", "GD2_G0", "MC2_G0", "MC3_G0", "MD3_G0" and "XJ_DATA") are pointers to these input files which are also required by "merge_concs". The "CONCDIR" variable points to the directory in which the Models-3/CMAQ-APT three-dimensional concentration file is located. For this example, this directory is the same as the post-processing directory, i.e., subdirectory "output". The variable "CC3_G0" points to the Models-3/CMAQ-APT three-dimensional concentration output file that is to be merged with the puff file by "merge_concs". This is the same as the file pointed to by variable "CONCfile" in the run script shown in Figure 5-2. Finally, "CC3_OUT" is the variable pointing to the output file of the "merge_concs" utility. This output file has exactly the same format as the Models-3/CMAQ-APT concentration file pointed to by "CC3_G0".

The rest of the "merge_concs" run script, not shown in Figure 5-4, sets up the links to the input and output files and executes the utility. Figure 5-5 shows a schematic of the input and output files of the post-processor.

The "**merge_deps**" utility adds the puff dry deposition data in the SCICHEM output file "projectname.dep" ("narsto.dep" for the NARSTO-Northeast example) to the gridded host model dry deposition values contained in the two-dimensional NetCDF deposition file ("DD1file" in Figures 5-2 and 5-3) produced in the simulation to create a new two-

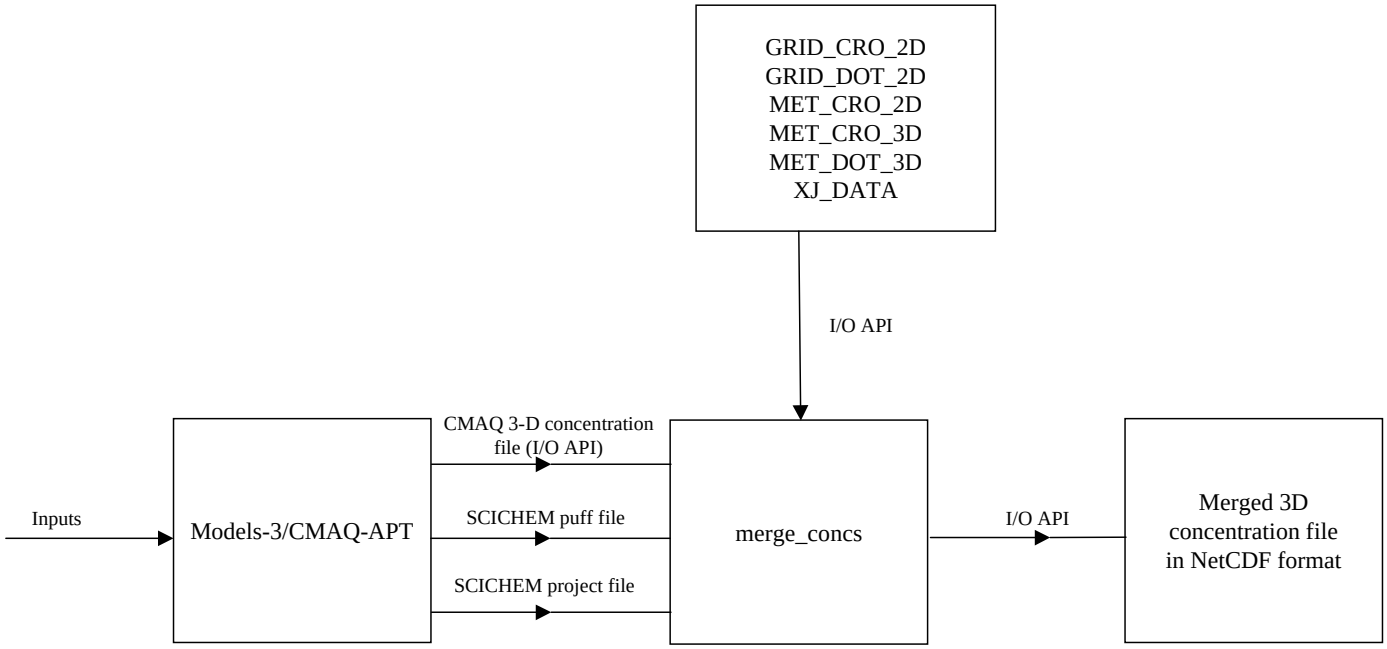

Figure 5-5. The “merge_concs” post-processing utility.

dimensional, time-dependent NetCDF file containing the combined puff and host model dry depositions.

An example run script for "merge_deps" for the first day of the NARSTO-Northeast simulation is shown in Figure 5-6. This script is similar to the "merge_concs" script shown in Figure 5-4, so we only point out the major differences here. The variable "DEPFILE" is a pointer to the "projectname.dep" file created in the Models-3/CMAQ-APT simulation. The variable "EXENAME" points to the "merge_deps" executable located in the directory specified in the variable "EXEDIR". The variable "DEPDIR" points to the directory in which the Models-3/CMAQ-APT two-dimensional deposition output file is located, while the variable "CD2_G0" points to this file. The output of the "merge_deps" utility is identified by variable "CD2_OUT". This output file has exactly the same format as the Models-3/CMAQ-APT deposition file pointed to by "CD2_G0". The remainder of the run script is straightforward and is not shown here. Figure 5-7 shows a schematic of the input and output files of "merge_deps".

Finally, the "**read_dmp**" utility reads the SCICHEM tracer dump output file "projectname.dmp" ("narsto.dmp" for the NARSTO-Northeast example) to produce three NetCDF files that can be viewed with PAVE, as described in Section 3.2. An example run script for this utility for the first day of the NARSTO-Northeast simulation is shown in Figure 5-8. The script is similar to those shown in Figures 5-4 and 5-6. The only differences are in the executable file names as well as the input and output files. The "read_dmp" utility does not require the meteorological inputs required by "merge_concs" and "merge_deps". In addition to the "projectname.dmp" file, it requires the header information from one of the NetCDF files produced in the Models-3/CMAQ-APT simulation to write the header information for the three NetCDF files that it creates. For the example shown in Figure 5-8, this header information is obtained from the three-dimensional concentration file pointed to by the "CC3_G0" variable. The variables pointing to the three "read_dmp" output files are "DUMP_3D", "DUMP_2D", and "DUMP_CUMULATIVE". Figure 5-9 shows a schematic of the input and output files of "read_dmp". Note that all three post-processing utilities require the "projectname.prj" (or "narsto.prj") file in the same directory where the post-processing is performed.

```
#! /bin/csh -f
# C-shell script to merge Models-3 dep file with scichem dep file
# Set up for the NARSTO domain

date; set timestamp; set echo

# These are the parameters for this run, set them carefully
set DEPFILE  = 'narsto.dep' # name of SCICHEM dep file
                            #("projectname".dep)
set STDATE   = 1995192      # simulation start date: year & Julian date
                            # (YYYDDD)
set STTIME   = 50000        # simulation start hour in GMT (HHMMSS)
set JDATE    = 1995192      # merge start date: year & Julian date
                            # (YYYDDD)
set JTIME    = 50000        # merge start hour in GMT      (HHMMSS)
set JEDATE   = 1995193      # merge end date: year & Julian date
                            # (YYYDDD)
set JETIME   = 50000        # merge end hour in GMT        (HHMMSS)
set OUTTS    = 10000        # output time step             (HHMMSS)

set GDATE    = 950711       # calendar date
set METGRID  = G2           # grid for met domain (12km)

set runid    = 18a

#################################################################
#     Set up script base and executable directories        #
#################################################################

# Need user input
set BASE     = .
set EXEDIR   = ../pigsrc/bin
set EXENAME  = merge_deps       # executable file name

#################################################################
#     Set up input files and directories             #
#################################################################

set METDIR    = ../input/met
set GC2_G0   = GRIDCRO2D_NARSTO_${METGRID}.$GDATE   # GRID_CRO_2D_G0
set GD2_G0   = GRIDDOT2D_NARSTO_${METGRID}.$GDATE   # GRID_DOT_2D_G0
set MC2_G0   = METCRO2D_NARSTO_${METGRID}.$GDATE    # MET_CRO_2D_G0
set MC3_G0   = METCRO3D_NARSTO_${METGRID}.$GDATE    # MET_CRO_3D_G0
set MD3_G0   = METDOT3D_NARSTO_${METGRID}.$GDATE    # MET_DOT_3D_G0

#################################################################
#     Set up dep files and directories               #
#################################################################
set DEPDIR = .
set CD2_G0 = DRYDEP.narsto_cb4_yb_ping_12km.${GDATE}.run${runid}
set CD2_OUT = DDEP_MERGED.narsto_cb4_yb_ping_12km.${GDATE}.run${runid}
……..
……..
```

Figure 5-6. Portion of the run script for the “merge_deps” utility for the first day of simulation.

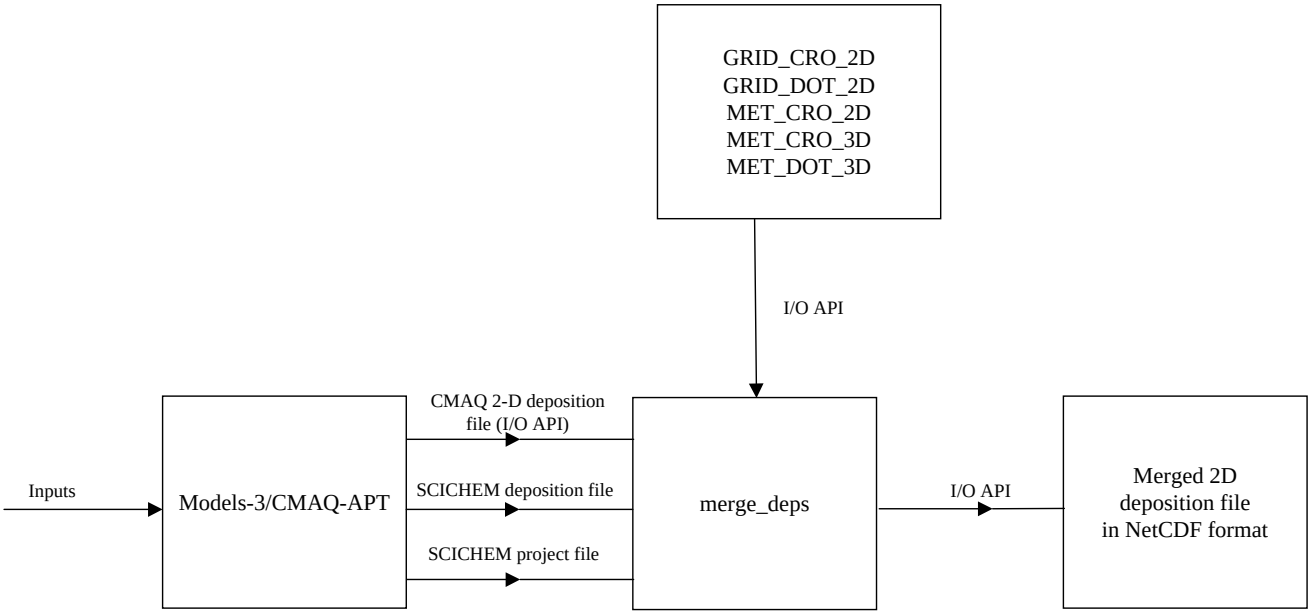

Figure 5-7. The “merge_deps” post-processing utility.

```
#! /bin/csh -f
# C-shell script to convert SCICHEM dmp file to netCDF format
# Set up for the Narsto domain
# Note: link to narsto.dmp should exist in current dir

date; set timestamp; set echo

# These are the parameters for this run, set them carefully
set DMPFILE  = 'narsto.dmp' # name of SCICHEM dump file
                            #("projectname".dmp)
set STDATE   = 1995192      # simulation start date: year & Julian date
                            # (YYYDDD)
set STTIME   = 50000        # simulation start hour in GMT (HHMMSS)
set JDATE    = 1995192      # file start date: year & Julian date
                            # (YYYDDD)
set JTIME    = 60000        # file start hour in GMT       (HHMMSS)
set JEDATE   = 1995193      # file end date: year & Julian date
                            # (YYYDDD)
set JETIME   = 50000        # file end hour in GMT         (HHMMSS)
set OUTTS    = 10000        # output time step             (HHMMSS)

set runid    = 18a

#################################################################
#     Set up script base and executable directories        #
#################################################################

# Need user input
set BASE     = .
set EXEDIR   = ../pigsrc/bin
set EXENAME  = read_dmp   # executable file name

#################################################################
#     Set up conc files and directories              #
#################################################################

set CONCDIR = .
set DUMPDIR = $CONCDIR

set CC3_G0  = CONC. narsto_cb4_yb_ping_12km.${GDATE}.run${runid}
set DUMP_3D = DUMP_3D.narsto_cb4_yb_ping_12km.${GDATE}.run${runid}
set DUMP_2D = DUMP_2D.narsto_cb4_yb_ping_12km.${GDATE}.run${runid}
set DUMP_CUMULATIVE = \
      DUMP_CUMULATIVE.narsto_cb4_yb_ping_12km.${GDATE}.run${runid}
……..
……..
```

Figure 5-8. Portion of the run script for the “read_dmp” utility for the first day of simulation.

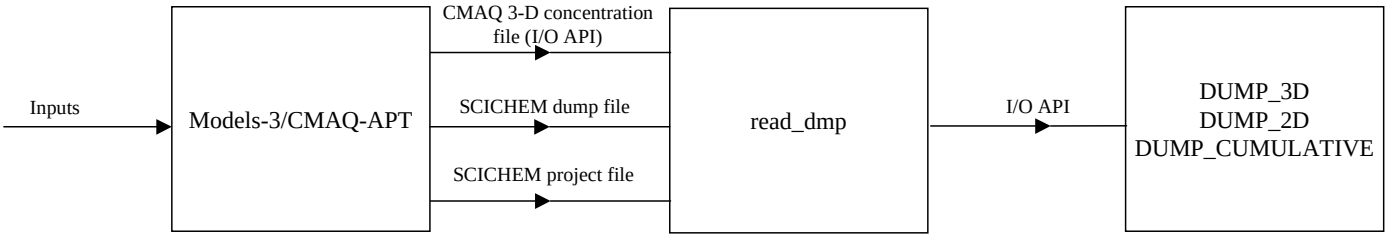

Figure 5-9. The “read_dmp” post-processing utility.

5.3 Performing a Nested Grid Simulation

For the NARSTO-Northeast application, a one-way nested grid simulation was conducted using the 12 km outer grid results to provide the boundary conditions for the inner 4 km grid. The build scripts, makefiles and run scripts for the inner grid are similar to those described in Section 4 and this section and are not discussed here. The main difference between a single grid simulation and the nested grid simulation is that the outer coarse grid simulation is conducted first and the boundary conditions for the inner fine grid are obtained from the coarse grid results. For a Models-3/CMAQ simulation (i.e., without plume-in-grid) the three-dimensional concentration file produced in the simulation is processed using the boundary condition processor that is provided with Models-3/CMAQ, as described below. For a Models-3/CMAQ-APT simulation, the three-dimensional concentration file produced with the "merge_concs" utility (see Section 5.3) is processed with the boundary condition processor.

Figure 5-10 shows a sample directory structure for the boundary condition processing. The "src" subdirectory contains the source code and build script for the boundary condition processor ("bcon"). The source code and a sample build script may be obtained from the Models-3 distribution. No changes need to be made to the source code. The input files for the processor (which are essentially the outputs from the parent or outer grid simulation) are located in the "input" subdirectory (either physically or as links to the output files). The "sim" directory contains the run script and the output boundary condition files for the nested grid are created in subdirectory "output".

Figure 5-11 shows portions of the sample build script for the NARSTO-Northeast application. The shaded sections identify user-defined variables that may need to be modified depending on the application. These include the environment variables, M3MODEL and M3TOOLS, the name of the application (APPL), the domain and mechanism variables (Reso, Grid, Layers, Mechanism, Tracer) and the pathname for the Fortran compiler (FC). If the Sun Workshop Compiler 5.0 is used, the path of the "f77compat" library must be set in the LIB3 variable (not required for the Sun Workshop Compiler 6.0).

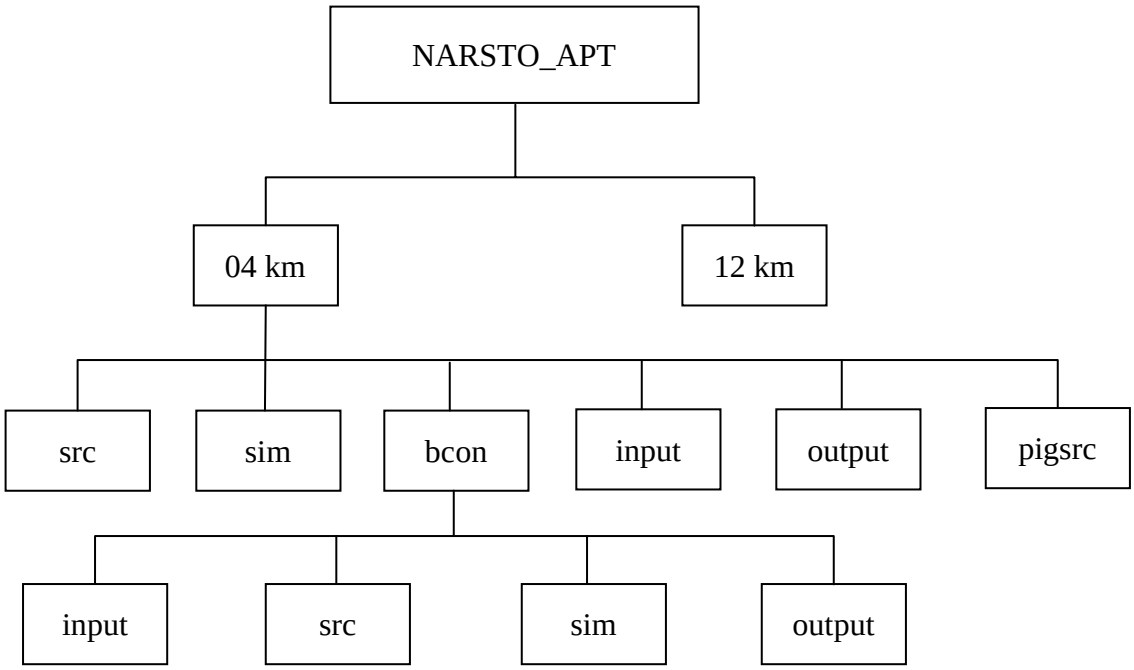

Figure 5-10. Directory structure for the NARSTO-Northeast application for the 4 km nested grid boundary condition processor.

```
#! /bin/csh -f
….
….
setenv M3MODEL /laplace1/models3/aug2000/models
setenv M3TOOLS /laplace1/models3/aug2000/tools
….
….
# user choices: base directory
#set Base = /your_dir/you/working_dir
set Base = $cwd

set APPL  = narsto_cb4_4km
….
….
# user choices: domain and mechanism

set Reso      = 04
set Grid      = 99X135
set Layers    = 13
set Mechanism = cb4
set Tracer    = trac0              # default: no tracer species

set Decomp    = 1X1
….
….
#setenv FORTRAN f90
set FC = /opt/5.0/SUNWspro/bin/f90

set LIB1 = "-L${M3TOOLS}/IOAPI/release/m3io/lib/SunOS5 -lm3iof90"
set LIB2 = "-L${M3TOOLS}/netCDF/SunOS5 -lnetcdf"
set LIB3 = "-L/opt/5.0/SUNWspro/SC5.0/lib -lf77compat"
set LIBS = "$LIB1 $LIB2 $LIB3"
….
….
```

Figure 5-11. An example build script for the nested grid boundary condition processor.

A sample “bcon” run script is also available in the Models-3 distribution. Figure 5-12 shows portions of the run script for the NARSTO-Northeast application for the first day of simulation (July 11, 1995) and highlights those sections that may need to be modified for other applications. These include the names of the applications for the nested grid and parent grid (APPL and APPLparent), the variables, “GDATE” and “runid12km” that are used to identify the input and output BC files, the variable “BNDY_CONC_1” that provides the pathname of the output BC file and the variable “CTM_CONC_1” that points to the parent output concentration file from which concentration are extracted. For

```
#! /bin/csh -f
….
….
set APPL     = narsto_cb4_yb_4km
….
….
set APPLparent = narsto_cb4_yb_12km

#set BASE     = /tmp/you/your_directory
set BASE     = $cwd
set HOMEDIR  = ..
set SRCDIR   = ${HOMEDIR}/src

set GDATE    = 950711
set runid12km = 18a

# set the path for the output file containing the BC's
set OUTDIR   = ${HOMEDIR}/output
if ( ! -d "$OUTDIR" ) mkdir -p $OUTDIR
setenv BNDY_CONC_1 $OUTDIR/bcon_${APPL}.${runid12km}.${GDATE}
….
….
#set BC = profile      # use default profile file
set BC = m3conc       # use CMAQ CTM concentration files (nested runs)
….
….
if ( $BC == m3conc ) then

setenv CTM_CONC_1
${HOMEDIR}/input/CONC.${APPLparent}.${GDATE}.run${runid12km}
….
….
endif
….
….
```

Figure 5-12. An example run script for the nested grid boundary condition processor.

the base Models-3/CMAQ simulation, this file is the direct output of the simulation. For a plume-in-grid simulation with Models-3/CMAQ-APT, this file is the merged concentration file created with the “merge_concs” processor described previously.

6. REFERENCES

Byun, D. W. and J. K. S. Ching, 1999. *Science Algorithms of the EPA Models-3 Community Multiscale Air Quality (CMAQ) Modeling System*, EPA-600/R-99/030, U. S. Environmental Protection Agency, Washington, DC.

Coats, C.J., A. Hanna, D. Hwang and D.W. Byun, 1993. Model engineering concepts for air quality models in an integrated environmental modeling system, presented at the *AWMA Specialty Conference on Regional Photochemical Measurement and Modeling Studies*, San Diego, CA.

Coats, C.J., 1995. The EDSS/Models-3 I/O Applications Programming Interface. http://envpro.ncsc.org/products/ioapi/index.html, MCNC Environmental Programs, RTP, NC, 1995-2001.

Coats, C.J., A. Trayanov, J.N. McHenry, A. Xiu, A. Gibbs-Lario and C.D. Peters-Lidard, 1999. An extension of the EDSS/Models-3 I/O API for coupling concurrent environmental models. Presented at the 15th IIPS Conference on Applications to Air Quality and Hydrology, American Meteorological Society, Dallas, TX.

EPA, 1998. *EPA Third-Generation Air Quality Modeling System, Models-3, Vol. 9B: User Manual*, U. S. Environmental Protection Agency, Washington, DC.

EPRI, 2000a. *SCICHEM Version 1.2: Technical Documentation*, EPRI Report 1000713, EPRI, Palo Alto, CA.

EPRI, 2000b. *SCICHEM User's Guide*, EPRI Report 1000843, EPRI, Palo Alto, CA.

Karamchandani, P., L. Santos, I. Sykes, Y. Zhang, C. Tonne and C. Seigneur, 2000a. Development and evaluation of a state-of-the-science reactive plume model, *Environ. Sci. Technol.*, **34**, 870–880.

Karamchandani, P., K. Vijayraghavan, S.-Y. Wu, C. Seigneur, L. Santos, I. Sykes, J.-F. Louis, T. Nerhkorn and J. Henderson, 2000b. *Development and Evaluation of a State-of-the-Science Plume-in-Grid Air Quality Model*, Draft Report, EPRI, Palo Alto, CA.

Karamchandani, P., C. Seigneur, K. Vijayaraghavan, S.-Y. Wu, L. Santos and I. Sykes, 2000c. *Further Development and Evaluation of Models-3/CMAQ-APT*, Draft Report, EPRI, Palo Alto, CA.

Rew, R.K. and G.P. Davis, 1990. NetCDF: An interface for scientific data access, *IEEE Comput. Graph. and Appl.*, **10**, 76–82.

Sykes, R. I., W. S. Lewellen, S. F. Parker and D. S. Henn, 1988. *A Hierarchy of Dynamic Plume Models Incorporating Uncertainty, Volume 4: Second-order Closure Integrated Puff*, EPRI, EPRI EA-6095 Volume 4, Project 1616-28.

Sykes, R. I., S. F. Parker, D. S. Henn and W. S. Lewellen, 1993. Numerical simulation of ANATEX tracer data using a turbulence closure model for long-range dispersion, *J. Appl. Met.*, **32**, 929–947.

Sykes, R. I. and D. S. Henn, 1995. Representation of velocity gradient effects in a Gaussian puff model, *J. Appl. Met.*, **34**, 2715-2723.

Appendix C
The CMU Cloud Module

The cloud-CMU routines are listed in Table C-1. Due to different chemical species in the two MADRID modules, several of the subroutines and include files listed in Table C-1 differ slightly between the cloud_CMU_aero_MADRID1 and cloud_CMU_aero_MADRID2 directories. There is no difference, however, in the subroutine arguments and functions. Two NETLIB subroutines are used in the cloud_CMU modules: the Variable-coefficient Ordinary Differential Equation solver package (DVODE) in double precision and the Powell hybrid method package (HYBRD). DVODE is used to solve the ordinary differential equations (ODEs) for the aqueous-phase reaction system, and HYBRD is used to obtain the steady-state concentrations of the aqueous-phase steady-state species. Note that the subroutines called by DVODE and HYBRD are given in Table C-1, but their functions are not provided (except for the subroutines FEX and STATE, which provide the functions for DVODE and HYBRD, respectively).

All the cloud include files except for AQ_PARAMS.EXT are located in the same source code repository directory as the source code and are copied into the working directory with cloud module source codes before CMAQ is compiled. A specific "AQ_PARAMS.EXT" is required for a specific gas-phase mechanism (e.g., CBM-IV or RADM2, RADM2-CI4, or CACM) and a specific size resolution (e.g., 2 or multiple size sections). Those mechanism- and size-dependent AQ_PARAMS.EXT files are located in the respective include directories.

Table C-1. The subroutines contained in the cloud_CMU modules used with CMAQ-MADRID.

Subroutine (Arguments)	Function	Called by	Routine(s) Called
ADDIT (RR, ARYTM, RP, RL)	Sums up the rates of the aqueous-phase kinetic reactions to give the rates for the main aqueous-phase species.	AQRATES	
AQCHEM (JDATE, JTIME, TEMP, PRES_PA, TAUCLD, PRCRATE, WCAVG, WTAVG, AIRM, ALFA0, ALFA3, GAS, AEROSOL, GASWDEP, AERWDEP, HPWDEP, HMSAWDEP, HSO5WDEP, SO4WDEP, SO5WDEP, PHRATENO2)	Maps species concentrations for the call to the Strader et al. aqueous-phase chemistry module; Computes concentration changes in cloud due to aqueous-phase chemistry, scavenging, and wet deposition.	AQMAP	AQ_STRADER
AQINTEGR (Y, STIME, STOUT)	Prepares the necessary variables for the call to the stiff ODE integrator (DVODE).	AQ_STRADER	DVODE AQRATES
AQMAP (JDATE, JTIME, WTBAR, WCBAR, TBARC, PBARC, CTHK1, AIRM, PRATE1, TAUCLD, POLC, CEND, REMOV, REMOVAC, REMOHMSA, REMOHSO5, REMOSO4, REMOSO5, ALFA0, ALFA3, PHRATENO2)	Provides an interface processor between the cloud dynamics module(s) and the aqueous-phase chemistry module.	RESCLD RADMCLD	CGRID_MAP AQCHEM
AQRATES (YAQ, AQPROD, AQDEST, YAQPRIME)	Calculates the rates of change of the species concentrations at a time step for the aqueous-phase species.	FEX	FULLEQUIL VALUES STEADY HYBRD REACT ADDIT MASS DIFFER
AQ_STRADER (TBEG,DELTAT,GAS,CCO2,AEROSOL,HSO5,HMSA,LWC, PRATE,TEMP,P,IRAD,PHOTOFAC,NSECT, DAER, FDIST)	Initializes the aqueous-phase chemistry calculation with the Strader et al. aqueous-phase chemical mechanism.	AQCHEM	DROPINIT CONSTANTS AQINTEGR
BADDIT (RR, R)	Calculates the reaction rates for the aqueous-phase steady-state species.	STATE	
BMASS (WL, RADIUS, TEMP, FGL, FLG, GFGL, GFLG)	Calculates the mass fluxes for the aqueous-phase steady-state species.	STATE	

Table C-1. The subroutines contained in the cloud_CMU modules used with CMAQ-MADRID (continued).

Subroutine (Arguments)	Function	Called by	Routine(s) Called
CLDPROC SUBST_GRID_ID (CGRID, JDATE, JTIME, TSTEP)	Provides an interface processor between the cloud module and the modules for other processes.	SCIPROC	CGRID_MAP RESCLD RADMCLD NEXTIME
CLDPROC_LWC SUBST_GRID_ID (JDATE, JTIME, TSTEP, TOT_WCBAR)	Calculates total cloud liquid water content to be used in the calculation of the gas-cloud heterogeneous reaction rate constant.	CHEM	RESCLD_LWC RADMCLD_LWC NEXTIME
CONSTANTS (TEMP)	Calculates the equilibrium constants, the Henry's law constants, and the reaction rate constants.	AQ_STRADER	
DIFFER (RP, RL, FGL, FLG, GFGL, GFLG, DP, DL)	Calculates the differentials for the gas- and aqueous-phase species.	AQRATES	
DROPINIT	Sets molecular weights for aqueous-phase species.	AQ_STRADER	
DVODE* (F, NEQ, Y, T, TOUT, ITOL, RTOL, ATOL, ITASK, ISTATE, IOPT, RWORK, LRW, IWORK, LIW, JAC, MF, RPAR, IPAR)	Solves the system of the ODEs for the aqueous-phase reaction system.	AQINTEGR	DCOPY F DEWSET DVHIN DSCAL DVINDY XERRWD DVSTEP DVJUST DVSET VNLS DAXPY DVJAC DVSOL JAC DGEFA DACOPY DGBFA DGESL DGBSL
ELECTRO (X, F, CON, SPRES, DCMET, AKEQ, AKHEN, WV, TEMP)	Calculates the electron balance for the electroneutrality equation.	FULLEQUIL	
FEX (NEQA, T, Y, F)	Provides the function for the aqueous-phase chemistry integrator.	DVODE	AQRATES

Table C-1. The subroutines contained in the cloud_CMU modules used with CMAQ-MADRID (continued).

Subroutine (Arguments)	Function	Called by	Routine(s) called
FULLEQUIL (ASPRES, AWV, ATEMP, AXSOL)	Solves the electroneutrality equation using the bisection method.	AQRATES	ELECTRO
GETALPHA (MASSI, LWC, T, P, RHOAIR, ALFA0, ALFA3)	Calculates the in-cloud scavenging coefficients for aerosol number and mass.	SCAVWDEP	
HLCONST (NAME, TEMP, PBARC)	Sets the Henry's law constants of species at a given temperature.	SCAVWDEP	
HYBRD (STATE, N, X, FVEC, XTOL, MAXFEV, ML, MU, EPSFCN, DIAG, MODE, FACTOR, NPRINT, INFO, NFEV, FJAC, LDFJAC, R, LR, QTF, WA1, WA2, WA3, WA4)	Computes the steady-state concentrations for the aqueous-phase steady-state species using the Powell hybrid method.	AQRATES	STATE FDJAC1 QRFAC QFORM DOGLEG R1UPDT R1MPYQ
INDEXN (NAME1, N, NAME2, INDICES)	Searches for all occurrences of NAME1 in list NAME2.	SCAVWDEP AQMAP	
MASS (WL, RADIUS, TEMP, FGL, FLG, GFGL, GFLG)	Calculates the mass fluxes for the mass balances.	AQRATES	
RESCLD (CGRID, JDATE, JTIME, TSTEP, N_SPC_WDEP, WDEP_MAP, DEP)	Calculates cloud characteristics for grid-resolved clouds and uses them to generate cumulative and net timestep deposition, cloud top, cloud bottom, and pressure at the lifting level.	CLDPROC	CGRID_MAP NEXTIME SCAVWDEP AQ_MAP
RESCLD_LWC SUBST_GRID_ID (JDATE, JTIME, TSTEP, TOT_WCBAR)	Calculates liquid water content for resolved-scale clouds to be used in the calculation of the gas-cloud heterogeneous reaction rate constant.	CLDPROC_LWC	NEXTIME
RADMCLD (CGRID, JDATE, JTIME, TSTEP, ICLDTYPE, N_SPC_WDEP, WDEP_MAP, DEP)	Calculates cloud characteristics for subgrid convective precipitating clouds and subgrid non-precipitating clouds, and uses them to generate cumulative and net timestep deposition, cloud top, cloud bottom, and pressure at the lifting level.	CLDPROC	CGRID_MAP NEXTIME SCAVWDEP AQ_MAP

Table C-1. The subroutines contained in the cloud_CMU modules used with CMAQ-MADRID (continued).

Subroutine (Arguments)	Function	Called by	Routine(s) Called
RADMCLD_LWC SUBST_GRID_ID (JDATE, JTIME, TSTEP, ICLDTYPE, TOT_WCBAR)	Calculates liquid water content for subgrid convective clouds be used in the calculation of the gas-cloud heterogeneous reaction rate constant.	CLDPROC_LWC	NEXTIME
REACT (RR, ARYTM)	Calculates the rates of the aqueous-phase reactions.	AQRATES STATE	
SCAVWDEP (JDATE, JTIME, WTBAR, WCBAR, TBARC, PBARC, CTHK1, AIRM, PRATE1, TAUCLD, POLC, CEND, REMOV, REMOVAC, REMOHMSA, REMOHSO5, REMOSO4, REMOSO5, ALFA0, ALFA3)	Computes in-cloud scavenging and wet removal.	RESCLD RADMCLD	CGRID_MAP GETALPHA
STATE (N, X, F, PAR)	Provides the vector of the functions for the aqueous-phase steady-state species.	HYBRD	VALUES REACT BADDIT BMASS
STEADY (RADIUS, TEMP)	Calculates the steady-state species concentrations.	AQRATES	
VALUES (X)	Calculates the concentrations of the aqueous-phase species.	AQRATES STATE	
Include Files	**Content**	**Used by**	
AERPAR.EXT	Aerosol parameters used in the CMU aqueous-phase chemistry module	DROPPAR.EXT	
AER_SECTIONS.EXT	Aerosol size information used in the calculation of aerosol activation by cloud droplets	AQMAD, AQCHEM, AQ_STRADER	
AQCONSTS.EXT	Defines constants used for the CMU aqueous-phase chemistry module	AQCHEM, AQ_STRADER, AQRATES, BMASS, ELECTRO, MASS, STEADY	
AQPAR.EXT	Species indices and molecular weight used for aqueous-phase chemistry module	ADDIT, BADDIT, DIFFER, ELECTRO, HLCONST, DROPCOM.EXT	
AQUEOUS.EXT	Defines fraction of crustal material, flag for chlorine chemistry, and error tolerance for numerical integration	AQCHEM, AQINTEGR, AQ_STRADER, AQRATES, REACT	

Table C-1. The subroutines contained in the cloud_CMU modules used with CMAQ-MADRID (continued).

Include Files	Content	Used by
AQ_PARAMS.EXT	Pointers for arrays GAS and AEROSOL used in the CMU aqueous-phase chemistry module and parameters used in aerosol activation parameterization	AQCHEM, AQMAP, AQ_STRADER, AQRATES
DROPCOM.EXT	Variables and arguments shared by the CMU aqueous-phase chemistry subroutines	AQCHEM, AQ_STRADER, BMASS, CONSTANTS, DROPINIT, FULLEQUIL, MASS, REACT, STATE, STEADY, VALUES
DROPPAR.EXET	Aqueous-phase parameters and variables	AQCHEM, AQINTEGR, AQ_STRADER, AQRATES, FEX, REACT
HYBRD.EXT	Input parameters for subroutine HYBRD	AQRATES
PHOTNO2.EXT	NO_2 photolysis rate used in the CMU aqueous-phase chemistry module	PHOT, RADMCLD, RESCLD

* subroutines of DVODE are located in the ae_aq_solver_MADRID subdirectory under the common_MADRID class of modules