Supplement of Geosci. Model Dev., 13, 2925–2944, 2020
https://doi.org/10.5194/gmd-13-2925-2020-supplement

Geoscientific
Model Development
Open Access

Supplement of

A multiphase CMAQ version 5.0 adjoint

Shunliu Zhao et al.

Correspondence to: Amir Hakami (amir.hakami@carleton.ca)

CMAQ v5.0 Adjoint User's Manual

The Adjoint Development Team

May 22, 2020

Contents

1 Introduction

The CMAQ adjoint is an active open-source development project of academic collaborators that augments CMAQ model development by the U.S. EPA to estimate sensitivities of functions of modeled concentrations with respect to initial concentrations or emissions parameters.

The adjoint model is comprised of three major components:

- the forward model, which generates checkpointing files for the science processes and average concentrations needed for forcing calculation
- the forcing generator, which creates forcing files
- the backward model, which reads the checkpointing and forcing files to calculate the sensitivity parameters.

To run the adjoint model, you need to

- prepare your system with the required libraries
- compile the adjoint source code with your favorite compilers
- run sequentially the forward model, the forcing generator, and the backward model.

This document is to provide a general guidance on building and running the adjoint model. For a detailed description of the adjoint, please refer to [Shunliu Zhao, 2019].

The latest version of this document comes with the model.

2 Installation

2.1 Prerequisites

To retrieve, compile, and run the adjoint model, the following software are required.

- Git
- netCDF 3.6.3 or later
- I/O API 3.0 or later
- MPI, e.g. OpenMPI. MPICH2, or MVAPICH2
- PGI/Intel/GCC Fortran and C compilers

2.2 Retrieve the adjoint

A copy of the CMAQ adjoint code can be obtained using Git.

```
git clone ssh://git@adjoint.colorado.edu/yanko.davila/cmaq_adj.git
```

To create a local Git branch for testing, change into the model directory and execute the following command.

```
git checkout -b test origin/5.0
```

2.3 Build the builder

The adjoint uses the CMAQ model builder to assemble and/or compile the code ([CMASWIKI, 2015]);

To build the builder from source, change directory to $ADJHOME/BLDMAKE_git, customize the Makefile if necessary, and then make:

```
cd $ADJHOME/BLDMAKE_git
make |& tee make.bld.log
```

Here, $ADJHOME is your work directory for the adjoint.

Ensure that bldmake is created.

2.4 Build CMAQ libraries

The STENEX and PARIO libraries from CMAQ are required for the adjoint for parallel job management and parallel input/output. Serial jobs needs only the NOOP version of STENEX and not PARIO. More details about the two libraries could be found at [CMASWIKI, 2015].

To compile STENEX, change directory to the corresponding directory, revise the Makefile, and then make:

```
cd $ADJHOME/stenex/se
make |& tee make.stenex.se.log
```

Check the log file for errors and the target directory for libse_snl.a and module files (.mod).

To compile pario,

```
cd $ADJHOME/pario

##Revise the Makefile
make |& tee make.pario.log
```

Check the log for errors and the target directory for libpario.a and module files (.mod).

2.5 Build the forward model

Change directory to scripts/

```
cd $ADJHOME/scripts
```

Using the build script as a template, cmaq_adj/scripts/bldit.adjoint.fwd.sample, create an executable for the forward sweep (comparable to CMAQ).

To configure a specific build, copy bldit.adjoint.fwd.sample to a file specific to your system and then customize the new file.

Execute the script (as exemplified with the sample script below).

```
./bldit.adjoint.fwd.[system-specific-name] |& tee bldit.fwd.log
```

A new directory should be made in $ADJHOME named BLD_fwd_bnmk. Change to it.

```
cd ../BLD_fwd_bnmk
```

Edit the Makefile to ensure compatibility with your system. Then, make the forward sweep executable.

```
make |& tee make.fwd.log
```

Check that the executable ADJOINT_FWD is created.

2.6 Build backward sweep executable

Follow the same procedure for the backward sweep of the model. To configure a specific build, copy bldit.adjoint.bwd.sample to a file specific to your system and then customize the new file.

Execute the script (as exemplified with the sample script below).

```
./bldit.adjoint.bwd.[system-specific-name] |& tee bldit.bwd.log
```

A new directory should be made in $ADJHOME named BLD_bwd_bnmk. Change to it.

```
cd ../BLD_bwd_bnmk
```

Edit the Makefile to ensure compatibility with your system. Then, make the forward sweep executable.

```
make |& tee make.bwd.log
```

Check that the executable ADJOINT_BWD is created.

3 Benchmark the adjoint

In this section, we will go through the steps listed in the introduction section about running the adjoint. The output files from each stage of the run could be compared with the ones we provide for benchmark.

3.1 Run the forward model

Running the forward sweep for the benchmark episode should be feasible after making modifications to the sample run script.

Change directory back to $ADJHOME/scripts.

Make a copy of the sample run script to modify it for your system.

```
cp run.adj.fwd.bnmk run.adj.fwd.bnmk.[system-specific-name]
```

Find all the instances of CHANGE to edit the run.adj.fwd.bnmk.[system-specific-name] to direct the model to the location of the unzipped CMAQ adjoint benchmark data.

Successful completion of the run will produce ctm log files for each date (20070610 - 20070613) and will be indicated by Normal Completion of program DRIVER_FWD in the last log file.

The intermediate values of CMAQ state variables are saved to the checkpointing files at each synchronization time step for each science process, if necessary. In the output directory, the following checkpointing files should be present for each day ($WDATE):

- CHEM_CHK_$WDATE
- VDIFF_CHK_$WDATE
- HA_RHOJ_CHK_$WDATE
- VA_RHOJ_CHK_$WDATE
- AERO_CHK_$WDATE
- CLD_CHK_$WDATE.

Comparisons can be made to the reference results provided in the benchmark data set.

3.2 Create adjoint forcing

The backward sweep requires an adjoint forcing file, which can be thought of as similar to the emissions input of the forward sweep. The forcing file must be an IOAPI-compatible netCDF file for the purposes of being read accurately. The layer and species being forced should be indicated in the backward run script through environment variables ADJNLAYS_FRC and ADJNSPC_FRC.

Python is an efficient scripting tool for creating an adjoint forcing file from the concentration files output by the forward sweep. If your system does not have Python installed, consider using the free Anaconda distribution, which will make package management easy. To use the provided script, ensure that you have the netCDF4 and numpy packages installed.

To create an adjoint forcing to test the backward sweep, please modify scripts/BnmkAdjForcCalc.py where CHANGE indicates the need. Then, execute it.

```
python BnmkAdjForcCalc.py
```

Four files entitled ADJ_FORCE.20070610, etc. should be added to your output directory.

3.3 Run the backward model

In the same way the forward sweep run script template was configured for your system, refine the backward sweep run template.

```
cp run.adj.bwd.bnmk run.adj.bwd.bnmk.[system-specific-name]
```

For the sample adjoint forcing script, set the following specifications:

```
 setenv ADJNLAYS\_FRC 1
 # Species forced (see /ICL/mech/cb05cl_ae5_aq_noaero/GC_EMIS.EXT
 # for number of species)
 #    examples: 1 is NO2, 2 is NO, 4 is O3
 setenv ADJNSPC_FRC 4

#> finite difference perturbation selection
### CHANGE
 # conduct finite difference test (T) or not (F)
 setenv FDM_TEST F
```

Execute the run script.

Comparisons can be made to the reference results provided in the benchmark data set.

4 Finite difference test

4.1 The finite difference module

A Fortran module (ADJ_FDM_TEST.F) comes with the forward model of the adjoint for finite difference test. The perturbation parameters such as perturbation species, types and locations can be specified via environment variables.

Below is an example in BASH.

```
export FDM_TEST=N ##Y
export PTB_SPC_NAME=SO2
export PTB_ABS=N ##N - percentage perturbation; Y - absolute perturbation
export PTB_SIZE=0.1 ##10% perturbation or absolute change,
                    ## depending on perturbation type
export PTB_TIME=230000  ##perturbation at the hours
                        ## so that the CONC file (instantaneous conc)
                        ## can be used for sens calc
export PTB_CRL1="107 68 1" ##starting col/row/lay
export PTB_CRL2="107 68 1" ##ending col/row/lay
```

4.2 Compare with the adjoint

A perturbation introduced in the finite difference test would propagate over time across the computational domain and through other species. In other words, the finite difference comprises one perturbation source and multiple receptors. The finite difference test thus provides results that can be used to calculate the sensitivities of the affected apecies at all locations at all times to the perturbed species at the specic perturbation location at the specific perturbation time.

The adjoint, on the contrary, involes one receptor and multiple sources. One comparable sensitivity pair is generated from a single run of the models. In practice, a single receptor, which could be in the form of a function of several concentrations, might be used and the less computationally expensive finite difference test is then repeated to produce a desirable number of sensitivity pairs for comparison.

Another way is to reduce the full models to column/box models by turning off, for example, the transport and cloud mixing processes. This can be done by commenting out the corresponding science processes in sciproc.F. It is important that the science processes for the forward/backward models of the adjoint, and the finite difference test have to be consistent.

5 Strategies to implement the adjoint

5.1 About checkpointing

To generate the checkpointing files, a fixed synchronization time step is required. This can be achieved by assigning the same value to the environment variables, CTM_MAXSYNC and CTM_MINSYNC.

It is recommended to perform a test forward run with the default values of CTM_MAXSYNC (720 seconds) and CTM_MINSYNC (60 seconds) and then adopt the minimum value of the synchronization time step present in the log files for checkpointing files.

The checkpointing files could take gigantic storage space for long-period simulations with large domains and fine resolutions. To reduce the storage requirement, one might split the entire modeling period into smaller intervals, and only start generating for the last interal the checkpointing files which are required for the following backward model runs. The checkpointing files are then regenerated when needed for the other intervals. The environment variable, CREATE_CHK, can be used to switch on/off checkpointing generation.

5.2 About forcing

Subroutine RD_FORCE_FILE in ADJOINT_FILES.F reads the forcing files and then applies the forcing to the corresponding species in your customized cost function.

Below is the piece of code that performs the forcing application. The names and indices of the species to be forced can be referred to Table 1. A DO loop, as demonstrated in the commented-out lines, can be used if multiple species are involved.

```
         DO R = 1, LENROW
            DO C = 1, LENCOL
!slz-pm25              DO CNT = 1,N_PM25_SPC
!slz-pm25                 V = PM25_SPC(CNT)
!slz-pm25                 ARRAY(C, R, 1, V) = ARRAY(C, R, 1, V)
!slz-pm25     &           + BUFFER(C, R, 1) * FRCFAC
!slz-pm25              END DO

               ARRAY(C, R, 1, 4) = ARRAY(C, R, 1, 4)  !slz layer 1; spc#4, o3
     &            + BUFFER(C, R, 1) * FRCFAC

!              ARRAY(C, R, 1, 1) = ARRAY(C, R, 1, 1)  !slz layer 1; spc#1, no2
!     &            + BUFFER(C, R, 1) * FRCFAC
            END DO
         END DO
```

Index	Name	Index	Name	Index	Name	Index	Name
1	NO2	36	ETH	71	SESQ	106	NUMCOR
2	NO	37	IOLE	72	SESQRXN	107	SRFATKN
3	O	38	TOL	73	RHOJ	108	SRFACC
4	O3	39	CRES	74	ASO4J	109	SRFCOR
5	NO3	40	TO2	75	ASO4I	110	AH2OJ
6	O1D	41	TOLRO2	76	ANH4J	111	AH2OI
7	OH	42	OPEN	77	ANH4I	112	ANAJ
8	HO2	43	CRO	78	ANO3J	113	ANAI
9	N2O5	44	MGLY	79	ANO3I	114	ACLJ
10	HNO3	45	XYL	80	AALKJ	115	ACLI
11	HONO	46	XYLRO2	81	AXYL1J	116	ANAK
12	PNA	47	ISOP	82	AXYL2J	117	ACLK
13	H2O2	48	ISPD	83	AXYL3J	118	ASO4K
14	XO2	49	ISOPRXN	84	ATOL1J	119	ANH4K
15	XO2N	50	TERP	85	ATOL2J	120	ANO3K
16	NTR	51	TRPRXN	86	ATOL3J	121	AH2OK
17	ROOH	52	SO2	87	ABNZ1J	122	AISO3J
18	FORM	53	SULF	88	ABNZ2J	123	AOLGAJ
19	ALD2	54	SULRXN	89	ABNZ3J	124	AOLGBJ
20	ALDX	55	ETOH	90	ATRP1J	125	NH3
21	PAR	56	ETHA	91	ATRP2J	126	SV_ALK
22	CO	57	CL2	92	AISO1J	127	SV_XYL1
23	MEO2	58	CL	93	AISO2J	128	SV_XYL2
24	MEPX	59	HOCL	94	ASQTJ	129	SV_TOL1
25	MEOH	60	CLO	95	AORGCJ	130	SV_TOL2
26	HCO3	61	FMCL	96	AORGPAJ	131	SV_BNZ1
27	FACD	62	HCL	97	AORGPAI	132	SV_BNZ2
28	C2O3	63	TOLNRXN	98	AECJ	133	SV_TRP1
29	PAN	64	TOLHRXN	99	AECI	134	SV_TRP2
30	PACD	65	XYLNRXN	100	A25J	135	SV_ISO1
31	AACD	66	XYLHRXN	101	A25I	136	SV_ISO2
32	CXO3	67	BENZENE	102	ACORS	137	SV_SQT
33	PANX	68	BENZRO2	103	ASOIL		
34	ROR	69	BNZNRXN	104	NUMATKN		
35	OLE	70	BNZHRXN	105	NUMACC		

Table 1: List of CGRID species of the cb05cl_ae5_aq mechanism.

5.3 About adjoint sensitivities

The adjoint model produces two sensitivity files for each simulation day, i.e, the 'lgrid' file for sensitivities to concentrations and the 'lgrid_em' file for sensitivities to emissions.

The frequency of writing sensitivities to file can be controlled by the environment variable of ADJ_LGRID_FREQ. The value of 'OUTPUT_STEP' indicates that writing occurs at each output time step (hourly by default), which is defined by the environment variable of CTM_TSTEP; the value of 'SYNC_STEP' suggests writing at each synchronization time step. The latter produces times larger sensitivity files than the former and should be used with caution, especially for long-term simulations with large domain and fine resolution.

References

[CMASWIKI, 2015] CMASWIKI (2015). Cmaq version 5.0 (february 2010 release) ogd — cmaswiki,. https://www.airqualitymodeling.org/index.php?title=CMAQ_version_5.0_(February_2010_release)_OGD&oldid=682. [Online; accessed 26-September-2019].

[Shunliu Zhao, 2019] Shunliu Zhao, Amir Hakami, S. L. N. J. O. B. K. M. F. S. L. C. M. D. T. D. K. H. P. B. P. J. R. A. G. R. A. N. J. B. G. R. C. C. O. S. A. S. T. C. (2019). A multiphase cmaq version 5.0 adjoint. *Geoscientific Model Development Discussions*, 2019:1–37. doi:10.5194/gmd-2019-287.