GenChem Documentation

Release 0.9.0

David Simpson et al.

May 15, 2020

CONTENTS:

1 Introduction **1**
- 1.1 Code structure . . . 1
- 1.2 Conventions in documenentation naming . . . 1
- 1.3 Requirements . . . 2

2 Quick start **3**
- 2.1 Step 1: initial setup . . . 3
- 2.2 Step 2: do.testChems . . . 3
- 2.3 2a. Plotting? . . . 4
- 2.4 2b. Box-config . . . 4
- 2.5 Step 3: emep_setup.py . . . 6

3 Formatting of GenChem files **9**
- 3.1 Reactions files . . . 9
- 3.2 Species files . . . 9
- 3.3 Shorthands file . . . 11

4 Contributors **13**

5 References **15**

6 Indices and tables **17**

Bibliography **19**

Index **21**

CHAPTER

ONE

Introduction

GenChem is a system to generate and test chemical mechanisms for the EMEP MSC-W model [Simpson2012] and 1-D canopy model, ESX [SimpsonTuovinen2014]. GenChem consists of two main directories, **chem** and **box**.

The **chem** directory contains several chemical mechanisms written in chemist-friendly format (e.g. *k1* NO2 + OH = HNO3). A python script *GenChem.py* can be used to convert these files to fortran friendly input files for the EMEP model, usually with the help of some wrapper script, either *do.GenChem*, *do.testChems*, or *emep_setup.py*. The fortran files produced by these scripts have the prefix "**CM_**" or "**CMX_**", where CM denotes Chemical Mechanism.

Although GenChem can be run directly from within the **chem** directory, the strongly recommended approach is to use the scripts available in the **box** directory. In this approach GenChem is first applied, and then the resulting CM files are compiled and run as box-model simulations. Once all looks okay, a final script can be run to add additional code, and provide an EMEP-ready set of fortran files. This approach ensures that the CM files compile as they should, and allows rapid testing of several chemical mechanisms alongside each other.

1.1 Code structure

The directory structure for GenChem can be summarised as:

```
XXX/chem                  # GenChem's mechanism tree
XXX/chem/scripts          # scripts, including do.GenChem and GenChem.py
XXX/chem/base_mechanisms  # collection of main chemical schemes
XXX/chem/extra_mechanisms # collection of extra reactions for chemical schemes
XXX/chem/inputs            # emissplit files, see ...

XXX/box                  # Top of ESX directory tree
XXX/box/src              # source files
XXX/box/scripts          # scripts

XXX/doc               # documentation, as .rst files plus sphinx conf system
XXX/doc/_build        # processed documentation, as .pdf and html
XXX/doc/_build/html   #  .. as .html  (aim your browser at index.html here)
XXX/doc/_build/latex # .. as .pdf  (aim your pdfreader at GenChemDoc.pdf here)
```

(where XXX could any suitable user-directory into which GenChem was unpacked, e.g. /home/fred/chemwork/GenChem.)

1.2 Conventions in documenentation naming

The input files to GenChem (GenIn files) as used in box or emep model are usually built up by appending files from one *base* directory (from base_mechanisms) and one or more (usually many!) *extra* mechanisms from the extra_mechanisms directory. For example, GenIn_Species.csv used for the EMEP CTM's default EmChem19p scheme consists of Species files from base_mechanisms/EmChem19a,

and from twelve extra_mechanisms directories (e.g. extra_mechanisms/SeaSalt/SeaSalt_Species.csv, extra_mechanisms/PM_VBS_EmChem19/PM_VBS_EmChem19_Species, etc.). To avoid having to write out these names explicitly each time, we adopt generic names, as illustrated below for the EmChem19p case:

```
SCHEME                name for complete chemical mechanisms package.
                      (currently EmChem19a, EmChem19p, CB6r2, CRIv2emep, MCM_v3.3)

BASE_Species.csv      base_mechanisms/EmChem19a_Species.csv

EXTRAS_Species.csv    extra_mechanisms/SeaSalt/SeaSalt_Species.csv,
                      extra_mechanisms/Aqueous_EmChem16x/Aqueous_EmChem16x_Species.
↪csv,
                      ....

CMDIR_Species.csv     Either base or extras file, e.g.
                      base_mechanisms/EmChem19a_Species.csv **or**
                      extra_mechanisms/SeaSalt/SeaSalt_Species.csv,
```

1.3 Requirements

GenChem has been developed on Ubuntu linux systems, and should work on any modern linux/unix computer. The code has also been run on Windows via a virtual ubuntu environment. The minumum requirements are a modern fortran compiler and python3 (probably 3.5 or higher).

We have used for example

- gfortran (gcc 4.6.1) on Linux Xubuntu PC system
- gfortran (gcc 4.4.7) on HP supercomputer
- ifort 13.0.1

CHAPTER

TWO

Quick start

We will proceed directly to a run of the box-model system, to show how chemical schemes are normally compiled into CM_ and CMX_ files, and used in box-model simulations. This is actually the normal and recommended way to prepare files for the EMEP model, but also provides a good environment for comparing chemical mechanisms.

2.1 Step 1: initial setup

If not run previously, some preliminary steps are needed to set up a working directory. From the **GenChem/box** directory (cd GenChem/box), do:

```
cd somepath/GenChem-xxx/box

scripts/box_setup.sh tmp_work
```

where _somepath_ is the user's path. The name _tmp_work is just an example - anything can be used.

2.2 Step 2: do.testChems

At this stage, one can try compiling a chemical scheme. With the example of EmChem19a, and now from our tmp_work directory, try:

```
./do.testChems  EmChem19a
```

This script will run GenChem.py on the EmChem19a scheme (also adding a few extra reactions from a helper BoxAero mechanism), run "make", and then run the resulting box-model code. Results will appear in one log-file (RES.EmChem19a, way too wordy!), and as comma-separated results in the Output directory: file boxEmChem19a.csv. This file is readable with e.g. libreoffice. Plot scripts are also available (see BELOW), for easy visualisation and comparison of these csv results.

The CM_ and CMX_ fortran files produced by this process are saved in directories, e.g. here in ZCMBOX_EmChem19a. These files could be used in the EMEP model if wanted, but usually the more complex script emep_setup.py (described below) is used for that. (Hence we reserve the prefix ZCMBOX for files created by do.testChems and ZCM for those created with emep_setup.py, see below.)

Now, if one wants to compare several schemes, one can do e.g.:

```
./do.testChems  EmChem19a CRIv2R5Em MCMv3.3Em
```

This would produce 3 output .csv files, which again are easily plotted against each other.

Technical comments:

- do.testChems is just a simple wrapper, which cleans up files, runs another script (do.GenChem), and runs the box model.
- MCM is a very large scheme and this can take a while, or stress your PC's memory! Try with the smaller schemes first.

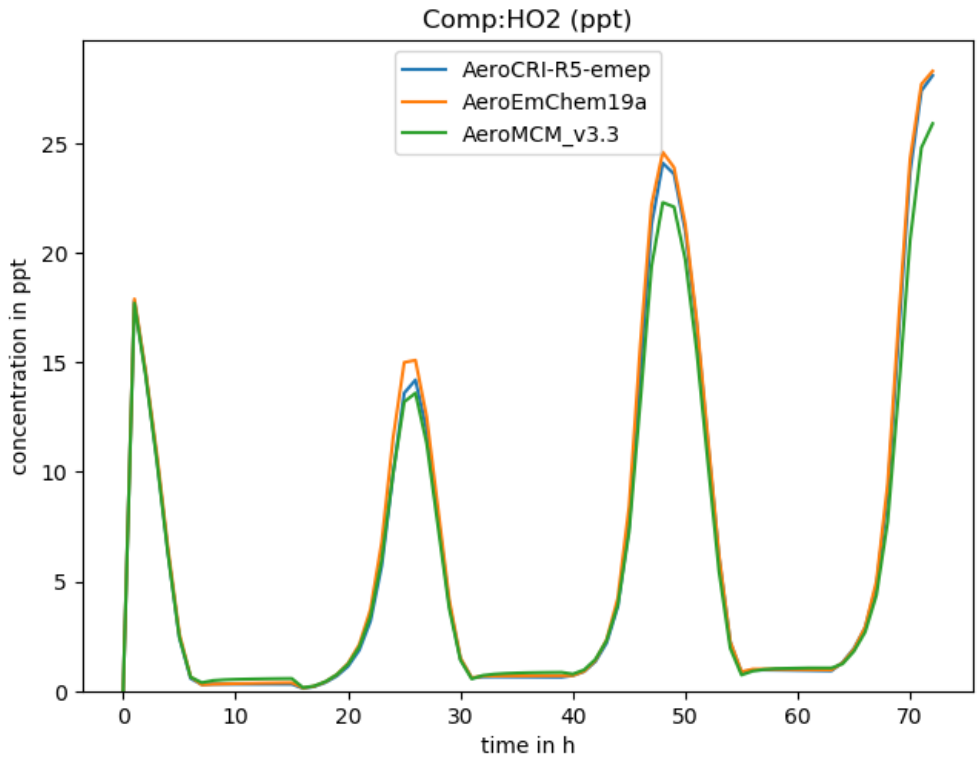
2.3 2a. Plotting?

If one has run say 3 chemical schemes using Step 2 above, the results are easily plotted from the *box/tmp_work/Output* directory:

```
../../scripts/boxplots.py -h     for help!

../../scripts/boxplots.py -v O3 -i boxEmChem19a.csv boxChem1.csv boxChem2.csv  -p
```

Using 'ALL' or 'DEF' with -v results in all or many common species being plotted at once (-p is assumed in this case. For example, here we can see a comparison of three schemes produced with this script:



Another crude+helpful script just grabs the concentrations:

```
../../scripts/getboxconcs.py  O3 boxEmChem19a.csv
```

which results in ResConcs_boxEmChem19a_O3_ppb.txt

2.4 2b. Box-config

The script do.testChems above compiles the executable boxChem for each mechanism in turn, and by default runs this using some settings from the default config_box.nml file. This file contains a number of important settings which by deault run a 24-hour simulation (starting at 12:00 GMT), with set emissions, temperature of 298.15 K, mixing height of 1000 m, and some boundary conditions. Default outputs are also given.

The user can of course change these settings (do this in your working directory, not in *src*). We explain the variables and choices here.

Note these config files follow fortran namelist conventions. Text following an exclamation mark (!) is ignored.

2.4.1 Time-related variables

```
! Time variables, all in seconds
! ------------------------------
 tstart = 43200., ! start at 12:00
! end time is absolute time -> total runtime is tend - tstart!
 tend = 302400.,  ! three days on top of 12 hours
 dt = 30.         ! time-step for numerical simulations
 doy = 182,        ! Day of the year
```

2.4.2 Geographical location

```
lat = 45.05,    ! degrees N
lon = 15.06,    ! degrees E
```

2.4.3 Emissions

```
 use_emis = T,     ! use emissions at all?
! directory with emissplit files:
 emissplit_dir = 'emissplit_run/'
 emis_kgm2day = 'nox', 18.3, ! NOx, kg/m2/day, as in MCM/CRI tests
                'voc', 15.4  ! NMVOC
 !emis_kgm2day = 'nox', 180.3, ! NOx, kg/m2/day, as in MCM/CRI tests
 !               'voc', 150.4  ! NMVOC

 ! BVOC emissions are set in chem/extra_mechanisms/BoxBVOCemis, where
 ! also a factor SUN is given for light-dependent emissions. These BVOC
 ! emissions can be adjusted with the factors below.

 fIso = 1.0,                    ! isoprene
 fMTL = 0.0,                    ! monoterpenes from light-dependent emissions
 fMTP = 0.0,                    ! monoterpenes from pool (Temp.)-dependent␣
↪emissions
 fSQT = 0.0,                    ! sesqui-terpenes
```

2.4.4 Outputs

```
! Can say just e.g. 'O3', to reduce size of outputs,
!  but in general usage 'all' is normally best.

  OutSpecs_list =
  'all', 'ppb'    ! Will switch to ug for OM

! Output Groups
! --------------
  OutGroups_list =
  'NOX',   'ppb',
```

2.4.5 Debug

Some flags produce more output. More documentation to be added later.

```
! -----

! For testing, one can assign all VOC to one species. Do that here:
!  dbgVOC = 'NODEBUG',
!  dbgVOC = 'C2H4',
  debug%Emis = 0
!  debug%VOC = 'C2H4'
  debug%Spec = 'NONE', !'C2H4'
  debug%SOA = 0
  debug%PM = .false.
  debug%Chem = .false.
```

2.5 Step 3: emep_setup.py

The do.testChems script described above is best for quickly testing and comparing different mechanisms. Usually these comparisons only involve gas-phase mechanisms such as EmChem19a or MCM_v3.3. However, the EMEP model usually requires a host of extra species and reactions to accommodate sea-salt, dust, organic aerosols, and pollen. It also requires files to specify how emissions and boundary conditions should be distributed among specific species, e.g. how a VOC emission should be split into C2H6, C2H4, nC4H10 etc.

In fact, for the EMEP model, GenChem produces many files which are copied into ZCM_XXX directories for the scheme XXX you wish to use:

$ls -x ZCM_EmChem19a/

CM_ChemDims_mod.f90 CM_ChemGroups_mod.f90 CM_ChemRates_mod.f90 CM_ChemSpecs_mod.f90 CM_DryDep.inc CM_EmisFile.inc CM_emislist.csv CM_EmisSpecs.inc CM_Reactions1.inc CM_Reactions2.inc CM_Reactions.log CM_WetDep.inc CMX_BiomassBurningMapping_FINNv1.5.txt CMX_BiomassBurningMapping_GFASv1.txt CMX_BoundaryConditions.txt config_box.nml run_emislist/ (with emislist.defaults.sox etc..)

The recommended way to get this directory is to use the script *emep_setup.py* from your temporary work directory within the **box** system. So, from e.g. box/tmp_work, do:

./emep_setup.py EmChem19a

or just:

```
./emep_setup.py
```

and this will provide a list of options.

You can edit the *emep_setup.py* scripts, maybe renaming it as *my_setup.py* directory. If selecting from the provided base_mechanisms and extra_mechanisms you only need to extend the possible command lines as provided by the *cmdx* dictionary:

```
cmdx['EmChem19a-vbs'] ='-b EmChem19a -e PM_VBS_EmChem19 '+common_IsoMT1
cmdx['CRIv2R5Em-M19'] ='-b CRIv2R5Em -e PM_JPAC_MT3 PM_Hodzic_Aromatics BVOC_XTERP_
↪CRI'+common_IsoMT3'
```

The '-b' argument gives the base mechanism, and then you can have any number of compatable extra mechanisms (-e argument).

(There are many possible combinations of packages - see Simpson et al., (2020, submitted) and the emep_setup.py code for many examples.)

Any keys from *cmdx* can be used by *emep_setup.py*. For example, if the user builds a new base scheme *usersChem* and some OA scheme, *usersSOA*, then *emep_setup.py* can be edited to add these as a new option:

```
cmdx['usersChem'] ='-b usersChem -e  usersSOA'+common
```

you could do:

```
do.testChems usersChem   # GOOD TO CHECK FIRST

emep_setup.py usersChem  # Creates ZCM_usersChem
```

CHAPTER

THREE

Formatting of GenChem files

3.1 Reactions files

Example:

```
* Some simple lines
1.4e-12*EXP(-1310.*TINV)          : O3 + NO    = NO2 + <O2>  ; acp2004
5.681e-34*EXP(-2.6*LogTdiv300)    : OP + <O2> + <M>  = O3 ; acp2004
2.15e-11*EXP(110.*TINV)           : OD + <N2>  = OP       ; Updated (IUPAC 2009)

emisfiles:sox,nox,co,voc,nh3
rcemis(NO,KDIM)                        : = NO  ;
```

- END-OF-LINE is ";". Text after this (e.g. references, or unused "products") will be ignored.
- Separator between rate coefficient and reaction is ":".
- lines beginning with "*" are comments (no ";" needed here)
- lines beginning with "rcemis" are emission terms
- lines beginning with "emisfiles" give name of emission files, e.g. nox
- Some coefficients are defined in GenIn.shorthand, e.g. TINV, LogTdiv300
- Anything else is simply used as the rate coefficient. (Do not add spaces!)

Four types of tracers/catalysts/yields are allowed, denoted by different types of parentheses:

1) e.g. [OH] + VOC -> SOA will put xnew(OH) into the loss rate of VOC, but will not change the loss rate of OH.
2) e.g. {O2} + OD -> OP will ignore the O2 term. Make sure it is in the reaction rate though if needed!
3) e.g. OP + <O2> + <M> -> O3 will ignore the O2 and M term AND add their concentrations to the reaction rate (multiply it). This system is only used for these "special" species (O2, N2, M) as they must be predefined, e.g. O2(k), in boxChem and/or EMEP codes.
4) e.g. 1.36e-11 : [OXYL] + [OH] = |YCOXY(0)| ASOC_ug1 + ... will replace the contents of the || term with yield coefficients which will be updated each time-step in the EMEP model. These variables (here YCOXY(0)) must be predefined in order for emep_setup.py and the emep model to compile.

3.2 Species files

The input to the GenChem.py script is GenIn_Species.csv, but this is assembled by do.GenChem from all needed _Species.csv files from the base_mechanisms and extra_mechanisms sub-directories. For example, for a typical emep run with base EmChem19a, do.GenChem appends EmChem19a_Species.csv, SeaSalt_Species.csv, and many more into one GenIn_Species.csv. The file contains columns with species name, type, formula, and various settings related to dry and wet deposition

The GenIn_Species.csv file is a spreadsheet-friendly comma-separated file where the characteristics of the chemical compounds are given:

```
Spec,adv,formula,MW,DRY,WET,Groups,!Comments
*
RO2POOL,1,RO2POOL,xx,xx,xx,xx,!
OD,0,O,xx,xx,xx,xx,!
NO2,1,NO2,xx,NO2,xx,NOx;OX;OXN;daObs,!
MACR,1,CH2=CCH3CHO,xx,MEK,xx,RCHO;carbonyl;Hstar_5p0e0;f0_0p05;DRx_2p6,!
BSOC_ng1e2,2,C,12.,ALD,ROOH,Cstar:0.1;DeltaH:30.0;OM25;PCM;BSOA,"! semi-volatile␣
↪OC from BVOC "
```

The meaning of the columns is:

Spec - Species name as used in model.

adv - Type of compound. CTMs usually distinguish between advected and non-advected (or short-lived) species, in order to minimise CPU needs (concentrations of short-lived compounds only need chemical reaction terms, not advection). In addition, the EMEP model handles semivolatile SOA species through special handling (see below), and some species are so long-lived (e.g. CH4) that they can be accurately calculated without multiple iterations. Allowed values of type are:

0 - for short lived compounds (e.g. OH), which are not advected in the EMEP model.

1 - for advected compounds (e.g. O3, HCHO)

2 - for semivolatile SOA compounds (e.g. BSOC_ng100). The EMEP model (and boxChem) tracks such species by compound rather than phase, and calculates the partitioning between the phases dynamically, based upon the compound's volatilty. Species labelled with type 2 are accounted within the list of advected species, but the start and end of the semivolatile list is calculated by GenChem.py, to produce integer variables which demarcate these semivolatile compounds, e.g. FIRST_SEMIVOL=136 and LAST_SEMIVOL=176.

3 - for compounds which react very slowly (e.g. CH4).

formula - If a true chemical formula is provided (e.g. CH3CHO, or O=CHC(O2)(CH3)CH2OH) then GenChem will calculate the number of atoms (C, H, O, S or N) and the molecular weight. Such formula must use capital letters; lower case letters are ignored as far as processing is concerned, but may be used to help document the intention, e.g. nC4H10 is identical to C4H10, or pm25 is particulate matter but whose formula we do not know. For example, an entry for an organic nitrate might have formula 'someNO3' which mixes lower and upper case. In this case the molecular weight must be given if this is needed for the chemical modelling. (Typically we do need the mass of emitted species, but not always the mass of other species since we usually use mixing ratios for advection and output in ppb units. Occassionally examples occur where mass is not strictly required, but where one wants to know the nitrogen content, typically where outputs are given in terms of e.g. ug(N)/m3. In this case, the 'someNO3' formula would be enough to allow GenChem to figure out that this compound contains one nitrogen atom.)

MW - can be dummy (xx) or a real number giving the molecular weight of the compound. When given, this value is used in place of any MW calculated from the formula. As noted above, the MW value is sometimes but not always needed. For some emitted compounds, usually connected with particulate matter where we do not know the composition, we have to give a dummy molecular weight. This information is used internally in the model to get associated mixing ratios, but outputs for such compounds should always be in mass-units so that consistency is preserved.

DRY - dry-deposition surrogate. The EMEP and ESX models calculate dry-deposition explicitly for a limited number of compounds, and here we can choose which of these compounds can be used as a surrogate for the desired species. For example, for O3 we simply use O3; for C2H5OOH we use the ROOH surrogate. If not dry-deposited, simply use xx. For the semivolatile SOA species EMEP/ESX CTMs will use this rate for the gas-phase fraction of the SOA.

WET - wet-deposition surrogate - similar to the dry deposition system. For example, for HCHO we simply use HCHO; for the semivolatile SOA species such as BSOC_ng100 we specify the same

wet-deposition as for fine-particulate matter (denoted PMf), and the EMEP/ESX CTMs will use this rate for the condensed fraction of the SOA.

Groups - specifies groups which species belong to (e.g. OXN for oxidised nitrogen, RO2 for peroxy radicals) and allows surrogate species or factors to be assigned to these groups, e.g. Cstar:10.0;Extinc:0.4 assigns a vapour pressure Cstar (used in SOA modelling) to be 10 (ug/m3) and an Extinc coefficient to be 0.4. It is important that these groups are separated by semi-colons, not commas. This rather powerful feature is discussed further in Simpson et al. (Submitted, 2020).

3.3 Shorthands file

Shorthands are text-strings used in the Reactions.txt file, usually to represent commonly used rate-coefficients. The meaning of the text-string is given in _Shorthand.txt file, e.g.

```
XT            temp
FH2O          (1.0+1.4e-21*h2o*exp(2200.0*TINV))
KHO2RO2       2.91e-13*exp(1300.*TINV) ! MCM2001 ...
KMT12         IUPAC_troe(2.8e-31*exp(2.6*Log300divT),2.0e-12,exp(-TEMP/472.),M,0.75-
↪1.27*(-TEMP/472.)/LOG(10.))
```

In these examples, XT is just a character-saving replacement for temp, FH2O gives a more complex expression, which also uses the pre-defined variable TINV = 1/temp. KHO2RO2 is a common rate-coefficient, but here we see that comments are allowed - anything afer the 2nd term. FInally, the KMT12 term shows that complex fuction calls are also allowed. IMPORTANT - avoid white space in any terms!

CHAPTER

FOUR

Contributors

The GenChem system was created over many years:

David Simpson, Norwegian Meteorological Institute & Chalmers, 1998-2019: wrote original GenChem.pl scripts, the boxChem system, assorted helper scripts (do.GenChem, boxplots.py, etc.), and python3 conversion.

Alan Briolat, Stockholm Environment Institute at York, 2013: wrote the first python version: GenChem.py.

Hannah Imhof, Chalmers, 2016: added extra flexibility and types of arrays (e.g. factor groups), plus further scripts. Added CRI and MCM chemical mechanisms.

John Johansson, Chalmers, 2017-2019: improved organisation and flexibility of GenChem system.

Robert Bergström, Chalmers and SMHI, 2017-2019: development of chemical mechanisms (e.g. EmChem19 family, VBS schemes) for gas and aerosols.

Alvaro Valdebenito, Norwegian Meteorological Institute, 2018-2019, various bug-fixes and updates. Added pollen.

CHAPTER

FIVE

References

CHAPTER

SIX

Indices and tables

- genindex
- modindex
- search

BIBLIOGRAPHY

[Bergstrom2020] Bergström, R.; Simpson, D.; others chemical mechanism paper - in preparation

[Simpson2012] Simpson, D., Benedictow, A., Berge, H., Bergström, R., Emberson, L. D., Fagerli, H., Flechard, C. R., Hayman, G. D., Gauss, M., Jonson, J. E., Jenkin, M. E., Nyri, A., Richter, C., Semeena, V. S., Tsyro, S., Tuovinen, J.-P., Valdebenito, Á., and Wind, P. The EMEP MSC-W chemical transport model – technical description *Atmos. Chem. Physics*, 2012, 12, 7825-7865

[SimpsonTuovinen2014] Simpson, D. and Tuovinen, J.-P., ECLAIRE Ecosystem Surface Exchange model (ESX), in Transboundary particulate matter, photo-oxidants, acidifying and eutrophying components. Status Report 1/2014, The Norwegian Meteorological Institute, Oslo, Norway, pp 147-154, 2014

[SimpsonEMEP2019] Simpson, D., Bergström, R., Tsyro, S. and Wind, P., Updates to the EMEP/MSC-W model, 2018–2019, in EMEP Status Report 1/2019, The Norwegian Meteorological Institute, Oslo, Norway, pp 145-155, www.emep.int, 2019

INDEX

I

INTRO, 1

Q

Quick Start, 2