Notes on gasphase_preproc and kpp4palm

1. Introduction

gasphase_preproc is a preprocessor that creates the file **chem_gasphase_mod.f90,** where the gas phase chemistry rate equations are solved within PALM4U.

gasphase_preproc is based on the original unchanged Kinetic PreProcessor KPP (Damian et al., 2002, Sandu et al., 2006), Release 2.2.3 from November 2012 (http://people.cs.vt.edu/asandu/Software/Kpp/, kpp-2.2.3.tar.gz) and an adapted version of the KPP postprocessor KP4 (Jöckel et al, 2010), which converts the KPP-generated code to a subroutine for PALM4U. The adapted version of KP4 is named **kpp4palm**.

KPP creates code for a box model from a list of chemical reactions, which must be written in a format that can be processed by KPP. This code is converted to a module for PALM4U by kpp4palm.

Besides the standard scalar version of the code, also a vectorized version of chem_gasphase_mod.f90 can be generated. However, only the different flavors of the Rosenbrock solvers have been vectorized, all other KPP solvers have to run in scalar mode.

The first version of this interface to PALM (still named kp4 at that time) was created by Klaus Ketelsen in November 2016 on the basis of his previous development of the handling of KPP in MESSy2 as described by Jöckel et al. (2010).

2. Requirements

Note that the FLEX library is required for KPP

3. Directory structure

Location: The **gasphase_preproc** directory is located in UTIL/chemistry (UTIL is located the trunk directory).

Contents of directory gasphase_preproc

- The script **run_kpp4palm.ksh** (script for running KPP and kpp4palmthe conversion program, see section 4 for usage)

- Directory **kpp:** KPP preprocessor creating code (Fortran in our case) from a list of chemical reactions.
- Directory **kpp4palm:** Interface for starting KPP and converting KPP output to a PALM4U subroutine.
- Directory **mechanisms:** Contains subdirectories with the input for KPP for some sample mechanisms and the file UserRateLaws.f90)
- This Readme file
- Optionally, the directory **tmp_kpp4palm** can be created when running run_kpp4palm.ksh. This directory contains intermediate files, e.g. the original KPP input and output.

Output of **run_kpp4palm.ksh** is the file **chem_gasphase_mod.f90** which contains the Fortran code of the chemistry subroutines for a user defined chemical mechanism. This file is automatically copied into the SOURCE directory.

The **required input files** of kpp4palm and KPP are located in the directories gasphase_preproc/mechanisms/def_*mech*, where *mech* stands for the name of any mechanism. A few sample mechanisms are already supplied in gasphase_preproc/mechanisms/. More mechanisms will be added and can also be added be the user.

Each of the def_*mech* directories contains the following input files:

- **chem_gasphase_mod.kpp** (containing some instructions for kpp4palm, such as the output language, directives for the photolysis reactions and vales of the compounds which are referred as 'fixed species'.
  Fixed species are compounds which are usually abundant and do not vary with time on the scale of tropospheric chemistry, e.g. O2 or N2. For some mechanisms also CO2 or methane are considered as fixed, i.e. which compounds considered as 'fixed' depend on the mechanism. Water vapor (H2O) is always considered as a 'fixed species' in the chemistry routines, since its concentration is calculated in the meteorological part of PALM-4U (if it were not considered as 'fixed', it would be transported twice).
- ***mech*.spc** (containing a list of the chemical species in KPP notation)
- ***mech*.eqn** (containing a list of the chemical reactions in KPP notation).

In addition, each gasphase_preproc/mechanisms/def_*mech* contains an already set output file chem_gasphase_mod.f90 just in case that kpp cannot be run on a user's system (usually due to the lack of the FLEX library). However, only preprocessed files for the scalar mode are supplied here. Sample files for the vector mode are not supplied since the optimum vector length depends on the

computer which is used.

The SOURCE directory also contains already a file named chem_gasphase_mod.f90.

If someone wants to switch to another mechanism than the one which is included in the SOURCE directory either run run_kpp4palm.ksh or copy the prepared chem_gasphase_mod.f90 from the respective def_*mech* directory into the SOURCE directory.

4. How to apply kpp4palm for available mechanisms

Make sure that the FLEX library is installed

1. Enter the UTIL/chemistry/**gasphase_preproc** directory.

- Choose a mechanism from the available sample of mechanisms in subdirectory **mechanisms** (subdirectories **def_*mech*** where *mech* stands for any of the sample mechanisms).

- Run **run_kp4palm.ksh**:
  run_kp4palm.ksh [-m *mech*] [-i *n*] [-v] [-l *vl*] [-k] [-u]

  -m *mech* permits the choice of the chemical mechanism. If -m *mech* is not specified, the mechanism *smog* will be used

  -i *n* (n=0,1,2) is optional and optimizes a part of the code by replacing indirect addressing of arrays by a sequence of statements without indirect addresses as described by Jöckel et al., 2010. If *n* is set to 0, then the code is not optimized. Default is 2.

  -k is an optional argument, which determines whether the temporary working directory tmp_kpp4palm is kept or deleted after termination of run_kp4palm.ksh. tmp_kpp4palm is removed when -k is omitted.

  -u is an optional argument. If set, the output file will also be copied into the def_*mech* directory. This option should be applied with caution since the original file chem_gasphase_mod.f90 in def_mech will be overwritten. The default setting is off.

  -v switches on the generation of the vector version of chem_gasphase_mod.f90. A vector length must be specified by **-l *vl***, with *vl* being the vector length.

-l *vl* can only be applied in combination with -v. see above.

During runtime a temporary directory tmp_kpp4palm is created. The newly created output file is copied from the temporary working directory tmp_kpp4palm to SOURCE/chem_gasphase_mod.f90 (the already existing file chem_gasphase_mod.f90 in the SOURCE directory is moved to chem_gasphase_mod.f90.sav and will be overwritten by the next run of run_kp4.ksh). If the -u option is applied, the ouput file will also be copied into the mechanisms/def_*mech* directory.

5. How to apply kpp4palm for a new mechanism

If you are not familiar with KPP, read the KPP documentation (PDF in UTILS/chemsitry/gasphase_preproc/kpp/doc) and have a look into the files of the existing def_*MECH* directories.

1) Create an new subdirectory def_*mech* in directory mechanisms.
2) Put your new *mech*.spc and *mech*.eqn into that new directory. Photolysis frequencies must be named according to the following examples: phot(j_no2), phot(j_hcho), phot(j_o3).
3) Copy a chem_gasphase_mod.kpp file into the directory and adapt it:
   a) Adapt the name of the mechanism in the two #include statement
   b) Adapt in #INLINE F90_DATA the number of photolysis frequencies `nphot`
   c) Adapt/extend in #INLINE F90_DATA the indices in the INTEGER, PARAMETER,PUBLIC statement
   d) Adapt/extend in #INLINE F90_DATA the character array phot_names: Note that the order of phot_names and the indices must match. Please note that the names are case sensitive. The available photolysis frequencies can be found in chem_photolysis_mod.f90 (array names_s).
   e) Adapt in the #INLINE F90_INIT section the 'fixed' species exactly to number of compound which are required for your mechanism. Please note that water vapor is considered as fixed within the chemistry module, as it is computed somewhere else.

6. Documentation and references

A local copy of the official documentation of KPP is found in gasphase_preproc/kpp/doc/ kpp_UserManual.pdf.

A brief documentation of first version kpp4palm (still named kp4) is located in gasphase_preproc/ kp4palm/doc/MESSy_in_PALM.pdf

References:

KPP web page:http://people.cs.vt.edu/asandu/Software/Kpp/

Damian, v., A. Sandu, M. Damian, F. Potra, and G.R. Carmichael: ``*The Kinetic PreProcessor KPP -- A Software Environment for Solving Chemical Kinetics*'', Computers and Chemical Engineering, Vol. 26, No. 11, p. 1567-1579, 2002.

Jöckel, P., Kerkweg, A., Pozzer, A., Sander, R., Tost, H., Riede, H., Baumgaertner, A., Gromov, S., and Kern, B.: Development cycle 2 of the Modular Earth Submodel System (MESSy2), Geosci. Model Dev., 3, 717-752, https://doi.org/10.5194/gmd-3-717-2010, 2010.

Sandu A., and R. Sander. "Technical Note: Simulating chemical systems in Fortran90 and Matlab with the kinetic preprocessor KPP-2.1", Atmospheric Chemistry and Physics, Vol. 6, p. 187-195, (2006).

Additional notes
(background information about KPP and kp4palm)

General remark

KPP and kpp4palm are strongly case sensitive. Adaptations to coding conventions for PALM should therefore only be applied after the essential processing is finalized.

How to add or modify expressions for rates

The file UserRateLaws.f90 in directory gasphase_preproc/mechanisms contains the rate laws which are actually used. This file is a copy of gasphase_preproc/kpp/util/UserRateLaws.f90 (which is one of the ‘auxiliary files’ mentioned in the kpp documentation).

Further rate laws may be added in gasphase_preproc/mechanisms/UserRateLaws.f90, e.g.

```
  REAL(kind=dp) FUNCTION ARR2( A0,B0, TEMP )
      REAL(kind=dp) :: TEMP
      REAL(kind=dp) A0,B0
      ARR2 = A0 * EXP( -B0 /TEMP )
  END FUNCTION ARR2
```

It can be extended by further rate laws. When run_kpp4palm.ksh is run UserRateLaws.f90 is copied from gasphase_preproc/mechanisms into gasphase_preproc/kpp/util.

Example:
In order to make kp4palm include ARR2 into chem_gasphase_mod.f90, kpp4palm/bin/kp4palm.ksh had to be edited:

Add ARR2 to KPP_SUBROUTINE_LIST, i.e.
KPP_SUBROUTINE_LIST="$KPP_SUBROUTINE_LIST Update_RCONST k_3rd **ARR2**"
This makes kpp4palm add ARR2 to the list of subroutines to be processed in the file KPP_SUBROUTINE_LIST.

Important: Within kpp/util/UserRateLaws.f90 the effective code lines of the subroutine must be the **after** any comment lines (otherwise kpp may create a memory fault).

About the type of rate ‘constants’.

Rate constants in *mech*.eqn (*mech* stand for any name of a mechanism) are handled differently, depending whether they are simple numbers or not.

In kpp/src/scanner.c there are three types of rates distinguished in StoreEquationRate (lines 578 ff): NUMBER, EXPRESION, and PHOTO.

NUMBER is clear. If rate is a number, then it is put to the initialization as a ‘constant rate coefficient’.

If the rate in mech.eqn includes anything which is different from a number (like brackets, _dp, or anything else), then the rate is of type EXPRESION. This type of rates is put into UpdateRconst.

PHOTO is identified by the occurrence of ‘hv’ in the reaction rate equations in *mech*.eqn: if(EqNoCase(spname,"HV")) isPhoto = 1; within scanner.c (line 692).

Appending _dp (which is requested by the PALM team) at the end of each number makes a number to an expression and the rate is put into SUBROUTINE UpdateRconst (which is what we want anyway).

The Update_Rconst calls issue

As the Box version of KPP generated code can also run for several hours, an updating of the rate constants is necessary. However, this is not required, when just a time step of a dynamical model must be covered. Then it is only necessary to call Update_Rconst only once at the beginning of each time step. Furthermore, Update_SUN is not necessary as photolysis will be calculated outside of the chemistry module.

Changes in kp4palm/src/fortran_file.C:

```
//  Update_RCONST has only to be called once per outer timeloop in KPP_FOR_PALM

    if(ip->get_token(0) == "CALL" && ip->get_token(1) == "Update_RCONST" ) {
      lo_line.insert(0,"!DELETE ");
    cout << lo_line << endl;
    }

//  Update_SUN must not be called within in KPP_FOR_PALM

    if(ip->get_token(0) == "CALL" && ip->get_token(1) == "Update_SUN" ) {
      lo_line.insert(0,"!DELETE ");
    cout << lo_line << endl;
    }
```

Modifications for photolysis

create_kpp_module.C:
Some additional lines were added after
void create_kpp_module::create_kpp_integrate()

Specification of indices for the photolyis frequencies and their names (which must match with the names of the available photolysis frequencies in chem_photolysis.f90) are specified in kp4palm/def_*mech*/chem_gasphase_mod.kpp

```
#INLINE F90_DATA
  !   Declaration of global variables for photolysis from INLINE
  INTEGER, PARAMETER :: nphot = 2
  !   phot Photolysis frequencies
  REAL(kind=dp) :: phot(nphot)

  INTEGER, PARAMETER,PUBLIC :: j_no2 = 1
  INTEGER, PARAMETER,PUBLIC :: j_rcho = 2

  CHARACTER(LEN=15), PARAMETER, DIMENSION(NPHOT) :: PHOT_NAMES =  (/ &
     'J_NO2          ','J_RCHO         '/)
#ENDINLINE
```

The declaration REAL(kind=dp) :: phot(NPHOT) does not really fit here, as this does not depend on the mechanism, but so far I did not find a better place.

Modifications for fixed species

Fixed species (when necessary) are initialized in chem_gasphase_mod.kpp as follows:

```
#INLINE F90_INIT
 fix(indf_h2o) = qvap
 fix(indf_o2)  = 0.2e+6_dp *  fakt
 fix(indf_co2) = 400.0_dp *  fakt
#ENDINLINE
```

Note that water vapor is considered as fixed as the water vapor variable q (qvap is already converted to molecules cm^-3) is computed somewhere else in PALM-4U. In the absence of a prognostic water vapor variable a constant value of 0.01 kp $kg^{-}1$ is assumed.)