Einbau der MESSy Chemie in PALM

Klaus Ketelsen

9. Nov. 2016

1 Strategie

Für das Stadtklimaprojekt ist es geplant eine auf den kinetischen Preprozessor kpp basierendes Chemiemodul in das Modell PALM einzubauen. Grundsätzlich gibt es zwei Möglichkeiten die kpp Chemie in PALM zu integrieren:

1.1 MESSy in PALM aufrufen

In diesem Fall würde die Schnittstelle zwischen MESSy und dem PALM Modell neu programmiert. Danach könnten sämtliche MESSy Submodelle und die MESSy Datenstruktur in PALM verwendet werden.

Für den Einbau müsste das MESSy *Base Model Interface Layer* (BMIL) für PALM neu entwickelt werden. Außerdem sind Anpassungen im *Submodel Interface Layer* (SMIL) erforderlich.

1.2 kpp Handling aus MESSy übernehmen

Die zweite Alternative für den Einbau des Chemiemodells ist nur das Handling des kinetischen Preprocessors kpp aus Messe zu übernehmen. In diesem Fall müssen folgende Komponenten neu entwickelt werden:

- eigene Datenstruktur für die chemischen Spezies
- Photolyse, Emission und Deposition
- Eingabe der Anfangswerte

2 Werkzeuge zum Generieren des kpp Codes

Für den Einsatz in PALM wurde die Version aus Kapitel 1.2 ausgewählt.

In Subdirectory Baum kpp_for_palm sind die Komponenten zum generieren des Codes für die kpp Chemie abgelegt. Dieser Baum könnte später parallel zu SOURCE ins PALM Repository aufgenommen werden.

2.1 kpp

Für den kinetischen Preprocessors *kpp* wird nicht die Version aus MESSy übernommen. Es wird stattdessen die aktuelle Version von dem unten angegebenen Link heruntergeladen.

Im Gegensatz zu MESSy werden in der vorliegenden ersten Version des Einbaus kpp in PALM keine Erweiterung vorgenommen. Es wird die Standardversion kpp verwendet und die im Kapitel 2.3.2 beschriebenen Eingabedateien werden wie *kpp* User Manuals beschrieben angelegt.

Inwieweit die MESSy Erweiterung auch im PALM gebraucht werden muss zu einem späteren Zeitpunkt geklärt werden.

- Download

  Download der Version kpp-2.2.3_Nov.2012.tar.gz von

  http://people.cs.vt.edu/~asandu/Software/Kpp/

- Installation

  1. export KPP_HOME=*install_dir*
  2. export PATH=$PATH:$ KPP_HOME/bin
  3. make
  4. Datei *bin* von regulären File in Directory umgewandelt
  5. make install

2.2 kp4

kp4.exe ist ein zweiter Preprozessor, der den mit kpp generierten Code nach behandelt. Folgende Aufgaben übernommen.

1. kpp generiert einzelne FORTRAN Files. Mit kp4 werden diese Files zu dem Modul kchem_kpp.f90 zusammengefasst.
2. In dem von kpp generierten Code werden viele indirekte Adressierung verwendet. kp4 ersetzt diese indirekten Adressierungen durch feste Adressen. Dadurch wird der Code deutlich schneller.
3. Optional kann ein Vektorcode erzeugt werden bei dem statt einer mehrere Gitterzellen parallel abgearbeitet werden. Die Vektorversion von kp4 muss noch für den Einsatz im PALM angepasst werden.

Übernahme aus MESSy und Anpassung der Installationsprozedur:

- Version

  cp -r $MESSY_ROOT/messy_2.52p1/messy/tools/kp4 .

  mv Makefile.m Makefile

  cd bin; mv kp4.ksh.orig kp4.ksh

- Installation
  - Make
- Änderungen
  - Die Directory *bin* parallel zu *src* gelegt.

- kp4.ksh Script angepasst
  - Default Arguments angepasst
  - initialize_indexarrays aus KPP_SUBROUTINE_LIST gestrichen
  - k_SIV_H2O2 aus KPP_SUBROUTINE_LIST gestrichen
  - k_3rd_iupac aus KPP_SUBROUTINE_LIST gestrichen
  - Precision aus KPP_SUBROUTINE_LIST gestrichen
    - Default Precision aus dem PALM Module *mo_kinds* wird verwendet

2.3 KPP_FOR_PALM

In diesem Subdirectory Baum sind die Komponenten zum generieren des Codes für die kpp Chemie abgelegt. KPP_FOR_PALM liegt parallel zur SOURCE Directory im PALM repository. Sowohl kpp als auch kp4 sind vorinstalliert und können wie folgt generiert werden:

- cd kpp_for_palm/kpp; make
- cd ../kp4; make install

2.3.1 Aufruf kp4

kp4.ksh ist ein Script, dass aus den kpp Eingabefiles (Kap. 2.3.2) das Module *kchem_kpp.f90* erzeugt. *kchem_kpp.f90* kann direkt im PALM Kontext übersetzt werden.

Vor dem Aufruf von kp4 muss KPP_HOME im folgenden Script gesetzt werden!

kpp_for_palm/kp4/bin/kp4.ksh

Aufrufparameter:

- -o dir Directory, in der das kpp Modul *kchem_kpp.f90* geschrieben werden soll
- -d dir Directory, in der die Definitionsfiles abgelegt sind
- -k Arbeitsverzeichnis wird nicht gelöscht
- -p Prefix

Im Verzeichnis KPP_FOR_PALM liegt ein Script *run_kp4.ksh* mit folgenden Funktionen:

- Generieren kpp
- Generieren kp4
- Erzeugen kchem_kpp.f90 im Directory SOURCE

run_kp4.ksh ist so voreingestellt, dass der Code für das Beispiel small_strato erzeugt wird.

2.3.2 Input Files KP4

in der mit -d angewählten Directory müssen folgende Dateien abgelegt werden:

- Prefix.kpp
- model.spc
- model.eqn

Bei diesen Dateien handelt es sich um Input Files des kpp Preprocessors. Das kpp User Manual befindet sich in kpp_for_palm/kpp/doc. Diese Beschreibung befindet sich in kpp_for_palm/kp4/doc.

Im Verzeichnis KPP_FOR_PALM/kp4 (Kap. 2.3) befinden sich in *def_chapman* und *def_small_strato* zwei Beispiele für einen kpp Input.

3 palm

Dieses Kapitel beschreibt die erforderlichen Änderungen um die kpp Chemie im PALM einzubauen.

In der vorliegenden Version sind folgende Komponenten installiert

1. Aufruf des Integrators der kpp Chemie aus prognostic_equations.
2. Transport der chemischen Spezies ähnlich wie bei passiv_scalar.
3. Ausgabe der chemischen Spezies als 3-D Felder.

Es wird zurzeit nur die Cache Version des PALM-Modells unterstützt. Die Vektor und die GPU Version können bei Bedarf seinen späteren Zeitpunkt nachgerüstet werden.

Es wird darauf hingewiesen dass die im Kapitel 2.2 beschriebene Vektorversion des kpp auch in die PALM Cache Version eingebaut werden kann.

Alle kpp Chemie Routine in PALM beginnen mit **kchem_.** Alle die Chemie betreffenden Änderungen sind in *#ifdef KPP_CHEM* eingeschlossen. Es ist grundsätzlich zu überlegen, ob die Preprocessor Directive beibehalten werden soll oder ob die Ausführung der Chemie nur über die Variable *use_kpp_chemistry* gesteuert werden soll.

3.1 kchem_kpp.f90

Das Modul *kchem_kpp* beinhaltet den mit *kpp* und *kp4* automatisch generierten Code.

Der automatisch generierte Code in kchem_kpp lässt sich zurzeit noch nicht OpenMP parallelisieren. Der Grund dafür sind im Code vorhandene mit SAVE deklarierte Variable.

3.2 kchem_driver.f90

kchem_driver ist die Schnittstelle zwischen kchem_kpp und dem PALM Modell. Alle Funktionen des automatisch generierten Codes in kchem_kpp werden von diesem Modul gerufen, d.h. das

PALM Modell selber ruft nur Funktion aus kchem_driver. Keine kchem_kpp Funktionalität wird direkt aus PALM gerufen.

Bei aktiver Chemie ist die PUBLIC Variable *use_kpp_chemistry* auf .TRUE. gesetzt.

Folgende TYPE Deklaration definiert die benötigten Variablen

```
TYPE   species_def
   CHARACTER(LEN=8)                                  :: name
   CHARACTER(LEN=16)                                 :: unit
   REAL(kind=wp),POINTER,DIMENSION(:,:,:)            :: conc
   REAL(kind=wp),POINTER,DIMENSION(:,:,:)            :: conc_p
   REAL(kind=wp),POINTER,DIMENSION(:,:,:)            :: tconc_m
   REAL(kind=wp),ALLOCATABLE,DIMENSION(:,:)          :: ssws, sswst
   REAL(kind=wp),ALLOCATABLE,DIMENSION(:,:)          :: flux_s, diss_s
   REAL(kind=wp),ALLOCATABLE,DIMENSION(:,:,:)        :: flux_l, diss_l
END TYPE species_def
```

Für jede Species wird ein Element dieses TYPEs im Feld *chem_species* angelegt. *species_def* und *chem_species* sind PUBLIC und können in PALM Routinen verwendet werden.

PUBLIC Unterprogramme in aus kchem_driver:

1. kchem_initialze
2. kchem_parin
3. kchem_integrate
4. kchem_swap_timelevel
5. kchem_check_data_output
6. kchem_define_netcdf_grid
7. kchem_define_netcdf_grid

3.3 kpp_chem Namelist

Für die kpp Chemie wird eine neue Namelist *kpp_chem* mit folgenden Größen eingeführt: Die Namelist wird mid dem Unterprogramm kchem_parin (kchem_driver.f90) eingelesen.

- icntrl(20) Fine tuning kpp (Integer Werte)
- rcntrl(20) Fine tuning kpp (Floating Point Werte)
- t_steps(50) Feste Zeitschrittweiten

  t_steps(1) = 0.0 → Automatische Zeitschrittberechnung

Ist diese Namelist in PARIN vorhanden, wird *use_kpp_chemistry* auf .TRUE. gesetzt. Es ist zu überlegen, ob *use_kpp_chemistry* als Variable in die Namelist aufgenommen werden soll.

Des weiteren ist zu überlegen, ob die Felder icntrl und rcntl so in der Namelist bleiben sollen oder ob die einzelnen Elemente in selbstbeschreibene Variablen umgewandelt sollen.

Beispiel: icntl(3) wird zur Character Variable IntegratorTyp mit dem möglichen Wert ros2

3.4 Änderungen in existierenden PALM Routinen

1. advect_ws.f90

   Für die Statistik Auswertung wird der Eingangsparameter sk_char um den Wert 'kc' erweitert. Die Statistik Behandlung für sk_char == 'kc' muss noch programmiert werden.

2. check_parameters.f90

   Aufruf des Unterprogramms *kchem_check_data_output* zum Setzen der Einheit ppm.

3. data_output_3d.f90

   Aufruf des Unterprogramms *kchem_data_output_3d* zur Vorbereitung der Ausgabe der 3-D Felder der chemischen Spezies.

4. netcdf_interface_mod.f90

   Aufruf des Unterprogramms *kchem_define_netcdf_grid* zur Auswahl des Grids in x,y und z Richtung. Für z wird zu angewählt, die gleiche Dimension wie für die u-Felder. Muss u.U. angepasst werden.

5. palm.f90

   Aufruf des Unterprogramms *kchem_initialize*, im Wesentlichen zum Allokieren der Felder

6. parin.f90

   Aufruf des Unterprogramms *kchem_parin*

7. prognostic_equation.f90

   Aufruf des Unterprogramms *kchem_integrate.*

   Neues Unterprogramm *tendency_caller,* Transport der chemischen Spezies.

8. swap_timelevel.f90

   Aufruf des Unterprogramms kchem_swap_timelevel

9. time_integration.f90

   Boundary Exchange für die chemischen Spezies.

10. Makefile

    - Neue Fortran Files *kchem_driver.f90* und *kchem_kpp.f90*
    - Neue Preprocessor Directive *-DKPP_CHEM*
    - Dependency Liste erweitern

4 Fragen und Todo Liste

1. Behandlung mehrerer Chemie Setup
2. Welche Einheiten werden von kpp erwartet?
3. Fehler bei Berechnung interner Zeitschritte
4. Erweiterung kp4 zum Speichern zusätzlicher 3D Variable in den kpp Kontext

   Ähnlich der fill_TEMP Routine zum setzten der Temperatur
5. Setzen oder Eingabe der Anfangswerte
6. Photolyse-Raten und Sonnenstand für kpp bereitstellen
7. Behandlung der Emission und Deposition
8. Überarbeitet des Handling der Felder *flux_s*, *diss_s*. *flux_l* und *diss_l*
9. Überarbeiten des Handling der Felder *ssws*, *sswst* und *wall_sflux*

   Zur Zeit werden *ssws* und *sswst* in *species*_def aufgenommen und in der Intitialisierungsphase auf Null gesetzt
10. Zur Zeit wird die potentielle Temperatur (3D Array *pt*) verwendet. Ist das OK?
11. *found* Variable in Spezies Struktur
12. Restart Handling
13. In der Namelist icntrl und rcntrl durch Klartext Character Variable ersetzen. (Kap 3.3)
14. Später: Vektor Version