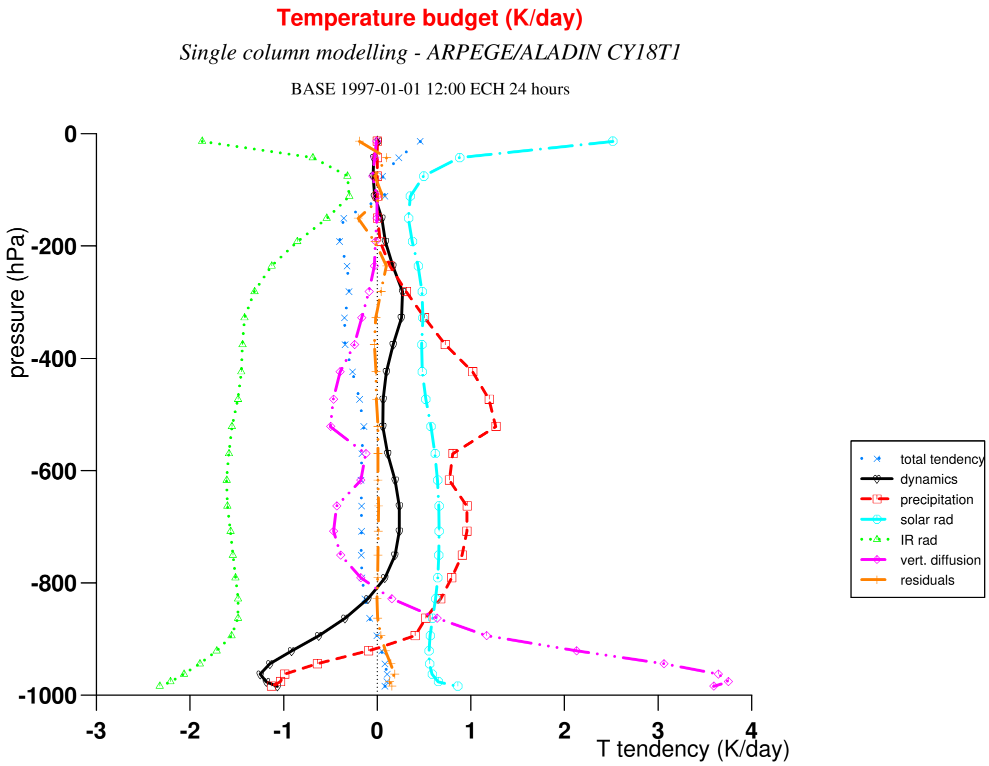
ARPEGE-ALADIN Single Column Model

Scientifical and Algorithmical Aspects

Jean-Marcel Piriou, François Bouyssel and Eric Bazile

METEO-FRANCE - CNRM/GMAP

December 31, 1999



Contents

Chapter 1

Main features

The present software provides single column modelling. The model is the 1D version of the 3D ARPEGE-ALADIN operational model. The aim is here to run the ARPEGE-ALADIN physical package in an interactive environment, to allow easier understanding and validating of this package. This single column model (SCM) can be used for diferents purposes:

- Make predictions in 3D model forced mode: in this mode, all dynamical terms are precribed from those got from a given 3D model run. This allows investigations on the physical package, while replaying in an interactive mode predictions done at a given point of a 3D run.
- Test the physical package on obervational datasets (like GATE, TOGA-COARE, BOMEX, etc...) as well as on idealized profiles.
- Develop surface variational assimilation, while using tangent linear and adjoint versions of the SCM.

Present state of this model: run single column simulations; free evolutions can be done, or forced ones, with dynamical forcing: impose advections, prescribe variables, relax them to a given profile (also called nudging). To be introduced soon: prescribe physical tendencies.

Some words now about this documentation: since the present SCM is the 1D version of the ARPEGE-ALADIN 3D model, for which a documentation already exists, this documentation has only two aims: firstly briefly remind basic equations and hypothesis of the 3D model; for further common elements with this 3D model, like the physics, see its own documentation. Secondly, to point out SCM specificities, like the forcing.

Chapter 2

Scientifical aspects

2.1 Basic equations

Prognostic variables

The atmospheric prognostic variables are

- $\vec{v}$ horizontal wind.
- T temperature.
- q_v water vapour mixing ratio.
- π surface pressure.

In the soil:

Vertical coordinate

If we note η the vertical coordinate, the 3D pressure field is written in the following way:

$$p(x, y, \eta, t) = A(\eta) + B(\eta)\pi(x, y, t)$$

were A and B depend on η, but not on time. A is zero at top and bottom of the atmosphere, while B goes from 0 to 1 from top to bottom of the atmosphere.

The vertical discretization is given by the choice of A and B functions.

Continuity equation

Let us note r_η the pseudo-density in η coordinate[1]. The continuity equation is

$$\frac{\partial r_\eta}{\partial t} = -\operatorname{div}_\eta(r_\eta \vec{v}) - \frac{\partial}{\partial \eta}(r_\eta \dot{\eta}) \tag{2.1}$$

[1] r_η is defined by $dm = r_\eta dS d\eta$.

Water vapour

The prognostic equation can be written as follows:

$$\begin{aligned}\frac{\partial q_v}{\partial t} = & -u\frac{\partial q_v}{\partial x} - v\frac{\partial q_v}{\partial y} - \omega\frac{\partial q_v}{\partial p} \\ & +T_{q_v-NUDG} + T_{q_v-ADD} \\ & +T_{q_v-PHYS} \\ & +T_{q_v-TFIL}\end{aligned} \tag{2.2}$$

where

T_{q_v-NUDG}	stands for the tendency due to	nudging
T_{q_v-ADD}		an additional term from your own (e.g. a prescribed tendency)
T_{q_v-PHYS}		physics
T_{q_v-TFIL}		time filtering

Temperature

$$\begin{aligned}\frac{\partial T}{\partial t} = & -u\frac{\partial T}{\partial x} - v\frac{\partial T}{\partial y} - \omega\frac{\partial T}{\partial p} + \frac{RT}{c_p}\frac{\omega}{p} \\ & +T_{T-NUDG} + T_{T-ADD} \\ & +T_{T-PHYS} \\ & +T_{T-TFIL}\end{aligned} \tag{2.3}$$

where

T_{T-NUDG}	stands for the tendency due to	nudging
T_{T-ADD}		an additional term from your own (e.g. a prescribed tendency)
T_{T-PHYS}		physics
T_{T-TFIL}		time filtering

Horizontal wind

Surface pressure

Soil fields

Stretching

Both Arpege and Aladin models are horizontally stretched ones; this means that x and y coordinates used for horizontal derivating or wind expression, are rotated and stretched with respect to real lengths on the Earth and West-East-North-South orientations.

The SCM is fully compatible with this stretching, and can be run with stretched or non stretched coordinates.

2.2 Temporal loop

With respect to the above prognostic equations, we compute next time step values from current ones in the following way:

1. We know the value of the prognostic variables $\vec{v}$, T, q_v, π for the current time step.

2. p is given by $p = A + B\pi$.

3. r_η is given under the hydrostatic hypothesis by $r_\eta = -\frac{1}{g}\frac{\partial p}{\partial \eta}$.

4. ϕ (geopotential) is given by the vertical integration of

$$\frac{\partial \phi}{\partial \eta} = -\frac{RT}{p}\frac{\partial p}{\partial \eta}$$

5. Next π is given by

$$\frac{\partial \pi}{\partial t} = g \int_0^1 \mathrm{div}_\eta \left(r_\eta \vec{v}\right) d\eta \tag{2.4}$$

6. The vertical velocity $\dot{\eta}$ is given by the vertical integral of continuity equation 2.1, using 2.4; this yields:

$$r_\eta \dot{\eta} = \frac{1}{g} B \frac{\partial \pi}{\partial t} - \int_0^\eta \mathrm{div}\left(r_\eta \vec{v}\right) d\eta$$

7. Next $\vec{v}$, T, q_v are given by the above prognostic equations.

2.3 Forcing

2.4 Observational datasets available

2.4.1 GATE

Purpose

GARP Atlantic Tropical Experiment The Global Atmospheric Research Program's (GARP) Atlantic Tropical Experiment (GATE) was conducted in the summer of 1974. One of the central objectives of GATE was to understand the scale interactions between convective activity and large-scale weather systems.

Dataset

The dataset was analysed by V. Ooyama and kindly made available to us by the NCAR. Radiative cooling data are from Cow and Griffith (1978), and the radar ones from Hudlow and Patterson (1979).

Forcing

2.4.2 BOMEX

Purpose

Barbados Oceanographic and Meteorologic Experiment The BOMEX experiment, designed by Davidson (1968), was carried out over a 500-km area east of Barbados during May and June 1969. This time period was selected in order to provide good convective activity without well developed storms. The "Core Experiment" of BOMEX proposed to determine rates of transfer of water vapor, heat, and momentum from the ocean to the atmosphere.

Dataset

Forcing

2.4.3 TOGA-COARE

Purpose

Tropical Ocean Global Atmosphere - Coupled Ocean-Atmosphere Response Experiment TOGA-COARE was conducted over the western Pacific warm pool in a region between 20N and 20S and bounded by Indonesia on the west and the date line on the east. The Intensive Observing Period for TOGA-COARE lasted from November 1992 through February 1993. The main goal of TOGA-COARE was to improve our understanding of the role of the western Pacific Ocean warm water pool (SSTs $> 28°C$) in the mean and transient state of the tropical ocean/global atmosphere system.

Dataset

Forcing

Chapter 3

Algorithmical aspects: SCM fortran code organization

3.1 1D <> 3D sources consistency

3.2 Calling tree

You can get the fortran calling tree from SCM by typing 'mtree' from 'src' directory. This has to be done AFTER the install process.

3.3 Data flow

3.4 Temporal loop

Let us take the example of water vapour temporal evolution: the other prognostic variables are managed the same way.

We show below how the different routines from SCM read this field from file, read forcings, and operate temporal evolution.

For equations, see scientifical part of the documentation.

```
- LITPROF: RQ0 read from initial profile file (article name PQT0).

- LITFORC: RQ0L and RQ0M read from forcing files (article name PQT0L and PQT0M).

- STEP0:

    - INTFORC: Interpolate forcings.
               (INTFORC/PQT0L = STEP0/ZQT0L) W
               PQT0L = PQT0M = 0 if no forcing.
               PQT0L and PQT0M interpolated from RQ0L and RQ0M if forcing.

    - CPG: Grid-point calculations.
           (CPG/PQT0 = STEP0/RQ0) R/W
           (CPG/PQT0L = STEP0/ZQT0L) R
           (CPG/PQT0M = STEP0/ZQT0M) R
        - IF(LFROG.AND..NOT.LDFSTEP) (not first step) LIT9: Read t-deltat variables from common.
                                                   (LIT9/PQT9 = CPG/ZQT9) W
                                                   PQT9 = RQT9
        - GPTF2: Second part of time filtering.
                 (GPTF2/PQT9 = CPG/ZQT9) W at 1st time step, R/W the other ones.
```

```
         (GPTF2/PQT0 = CPG/PQT0) R
         IF(LFROG.AND.LDFSTEP) THEN
              PQT9 = PQT0
         ELSE
              PQT9 = RKROMA*PQT9 + REPSM2*PQT0
                       1.           0.05
                                    0.1    IF(LECMWF)
                                    0.     IF(LCLIMA)
         ENDIF
- ZQT1 = ZQT9
- CPPHINP: Compute humidity convergence.
         (CPPHINP/PQT0 = CPG/PQT0) R
         (CPPHINP/PQT0L = CPG/PQT0L) R
         (CPPHINP/PQT0M = CPG/PQT0M) R
         (CPPHINP/PEVEL0 = CPG/ZEVEL0) R ETADOT DP/DETA
- APLPAR: Physics.
          (APLPAR/PQ = CPG/ZQT9) R
- CPTEND: Compute physical tendency.
          (CPTEND/PQ = CPG/ZQT9) R
- CPUTQY: Add physical tendency.
          (CPUTQY/PQT1 = CPG/ZQT1) R/W
         PQT1 = PQT1 + physical tendency.
- CPDYN: Add dynamical tendency.
         (CPDYN/PQT1 = CPG/ZQT1) R/W
         (CPDYN/PQT0 = CPG/PQT0) R
         (CPDYN/PQT0L = CPG/PQT0L) R
         (CPDYN/PQT0M = CPG/PQT0M) R
         PQT1 = PQT1 + dynamical tendency (function of PQT0, PQT0L, PQT0M).
- VARFORC: Force prognostic variables.
         (VARFORC/PQT1 = CPG/ZQT1) R/W
         (VARFORC/PFQV = CPG/PFQV) R
- CPDYDDH: DDH Diagnostics.
           (CPDYDDH/PQV = CPG/PQT0) R
- RECOPIE: Write results from current time step on file.
           (RECOPIE/PQ = CPG/ZQT9) R
           (RECOPIE/PQ1 = CPG/ZQT1) R
           CPG/ZQT9 is written on "PQ" article name.
           CPG/ZQT1 is written on "ZQ1" article name.
- If LFROG GPTF1: First part of time filtering.
         (GPTF1/PQT0 = CPG/PQT0) R
         (GPTF1/PQT9 = CPG/ZQT9) W at last time step, R/W the other ones.
         IF(LDLSTEP) (last step) THEN
              PQT9 = 0.
         ELSE
              PQT9 = REPSM1*PQT9          +   ZRESTM*PQT0
                     0.05                     (ZRESTM = (1.-(REPSM1+REPSM2))*RKROMA)
                     0.1 IF(LECMWF)
                     0.  IF(LCLIMA)
         ENDIF
- IF(LFROG.AND..NOT.LLLSTEP) (not last step) ECR9: Write t-deltat variables on common.
                                          (ECR9/PQT9 = CPG/ZQT9) R
                                          RQT9 = PQT9
- PQT0 = ZQT1
```

3.5 Stretching

3.6 Forcing

3.6.1 Free mode

3.6.2 Force all dynamical terms

3.6.3 Force variables

3.6.4 Relax variables

3.6.5 Force physical or dynamical tendencies

3.7 Real data precision

What is the role of real data precision in the model? We can say roughly speaking that the SCM can be run either in single precision (4 bytes real data) or double (8 bytes real data) with negligeable differences in final results.

The advantages and disadvantages between the two options are then the following:

- To run in single precision mode, some lines from two routines (radiation scheme, setup) have to be modified with respect to the 3D ones.
- The cost of doubling the precision is the following, all the values being estimated on a HP 9000/715 workstation, in fortran 90: time cost: +15%, memory size: +22%, output file size: +37%.

This shows that the choice depends if you are more regarding on human or machine time!...

Modifications to be done in sources to run in single precision mode:

ACNEBT: ZEPS1 to be changed from 1.E-12 to 1.E-4.

ACRANEB: 386c386,391

```
<         ZTDETA=ZRT**49*EXP(GCE4/PT(JLON,KTDIA))
---
>    !
>    ! Modification in acraneb and suphy3: GCE4 becomes the log
>    ! of itself, so that suphy3 and acraneb can be run in single precision.
>    !
>         ZTDETA=49.*LOG(ZRT)+GCE4/PT(JLON,KTDIA)
>         !ZTDETA=ZRT**49*EXP(GCE4/PT(JLON,KTDIA))
398c403,407
<         ZCTH(JLON)=ZRTH(JLON)*ZT4I*(1.+GCD4*ZQ*ZTDETA)
---
>    !
>    ! Modification in acraneb and suphy3: GCE4 becomes the log
>    ! of itself, so that suphy3 and acraneb can be run in single precision.
>    !
>         ZCTH(JLON)=ZRTH(JLON)*ZT4I*(1.+EXP(GCD4+LOG(ZQ)+ZTDETA))
487c496,501
<           ZTDETA=ZRT**49*EXP(GCE4/PT(JLON,JLEV))
---
>    !
>    ! Modification in acraneb and suphy3: GCE4 becomes the log
>    ! of itself, so that suphy3 and acraneb can be run in single precision.
>    !
>           ZTDETA=49.*LOG(ZRT)+GCE4/PT(JLON,JLEV)
>           !ZTDETA=ZRT**49*EXP(GCE4/PT(JLON,JLEV))
505c519,524
<           ZCTH(JLON)=ZCTH(JLON)+(ZDUH*ZT2I*ZRT)*ZT4I*(1.+GCD4*ZQ*ZTDETA)
---
>    !
>    ! Modification in acraneb and suphy3: GCE4 becomes the log
>    ! of itself, so that suphy3 and acraneb can be run in single precision.
>    !
```

```
>           ZCTH(JLON)=ZCTH(JLON)+(ZDUH*ZT2I*ZRT)*ZT4I
>      &    *(1.+EXP(GCD4+LOG(ZQ)+ZTDETA))
600c619,624
<           ZTDETA=ZRT**49*EXP(GCE4/PT(JLON,JLEV))
---
>    !
>    ! Modification in acraneb and suphy3: GCE4 becomes the log
>    ! of itself, so that suphy3 and acraneb can be run in single precision.
>    !
>           ZTDETA=49.*LOG(ZRT)+GCE4/PT(JLON,JLEV)
>           !ZTDETA=ZRT**49*EXP(GCE4/PT(JLON,JLEV))
618c642,647
<           ZCTH(JLON)=ZCTH(JLON)+(ZDUH*ZT2I*ZRT)*ZT4I*(1.+GCD4*ZQ*ZTDETA)
---
>    !
>    ! Modification in acraneb and suphy3: GCE4 becomes the log
>    ! of itself, so that suphy3 and acraneb can be run in single precision.
>    !
>           ZCTH(JLON)=ZCTH(JLON)+(ZDUH*ZT2I*ZRT)*ZT4I
>      &    *(1.+EXP(GCD4+LOG(ZQ)+ZTDETA))
699c728,733
<         ZTDETA=ZRT**49*EXP(GCE4/PT(JLON,KTDIA))
---
>    !
>    ! Modification in acraneb and suphy3: GCE4 becomes the log
>    ! of itself, so that suphy3 and acraneb can be run in single precision.
>    !
>         ZTDETA=49.*LOG(ZRT)+GCE4/PT(JLON,KTDIA)
>         !ZTDETA=ZRT**49*EXP(GCE4/PT(JLON,KTDIA))
711c745,750
<         ZCTH(JLON)=ZCTH(JLON)+(ZDUH*ZT2I*ZRT)*ZT4I*(1.+GCD4*ZQ*ZTDETA)
---
>    !
>    ! Modification in acraneb and suphy3: GCE4 becomes the log
>    ! of itself, so that suphy3 and acraneb can be run in single precision.
>    !
>         ZCTH(JLON)=ZCTH(JLON)+(ZDUH*ZT2I*ZRT)*ZT4I
>      &  *(1.+EXP(GCD4+LOG(ZQ)+ZTDETA))
```

SUCST: 119c119,122

```
<       RSIGMA=2. * RPI**5 * RKBOL**4 /(15.* RCLUM**2 * RHPLA**3)
---
> C     Modification to run in single precision mode.
> C     RSIGMA=2. * RPI**5 * RKBOL**4 /(15.* RCLUM**2 * RHPLA**3)
>       RSIGMA=2. * RPI**5.
>      &  *EXP(LOG(RKBOL)*4.-LOG(15.)-LOG(RCLUM)*2-LOG(RHPLA)*3.)
```

SUPHY3: 107c107,113

```
<       GCD4=.3608E-69
---
>  !
>  ! Modification to run in single precision:
>  ! done consistently in acraneb and suphy3: GCE4 becomes the log
>  ! of itself.
>  !
>       !GCD4=.3608E-69
>  GCD4=-159.897802
```

3.8 DDH diagnostics

Chapter 4

Practical aspects

4.1 Authors address

For questions or bug reports, please contact the co-authors:

- Jean-Marcel Piriou e-mail: Jean-Marcel.Piriou@meteo.fr
- Francois Bouyssel e-mail: bouyssel@cnrm.meteo.fr
- Eric Bazile e-mail: Eric.Bazile@meteo.fr

4.2 UNIX directories tree

documentation: the main documentation of the SCM is available here.

lfa: contains the LFA software, which purpose is to handle with model file I/O.

run: is the directory where SCM is to be run. This directory contains relevant scripts and data to run the SCM on special issues, like GATE, TOGA-COARE, make a validation of current SCM version againts previous reference ones, etc...

src: is the directory where sources are located.

bin: This directory contains the objects and SCM executable file. It contains also a copy of the last version of all sources from the SCM. NO MODIFICATION HAS TO BE DONE DIRECTLY IN THIS DIRECTORY.

src_alt_*: These directories contain the reference sources, from another origin than ARPEGE/ALADIN model: ECMWF, HIRLAM, ... NO MODIFICATION HAS TO BE DONE IN THIS DIRECTORY. Only copy reference sources as given by these developers.

src_arp_base: This directory contains the reference sources, *i.e.* the 3D ARPEGE/ALADIN operational model sources with no modification. NO MODIFICATION HAS TO BE DONE IN THIS DIRECTORY.

src_arp_mod: This directory contains copies from some sources of src_arp_base which have been slightly modified for SCM use. NO MODIFICATION HAS TO BE DONE IN THIS DIRECTORY.

- **src_m1d_specific:** This directory contains two kinds of sources: 1. Sources which have been created for the SCM. 2. Sources which exist in the 3D model, but have been entirely rewritten for SCM purpose. NO MODIFICATION HAS TO BE DONE IN THIS DIRECTORY.
- **src_user:** YOUR MODIFICATIONS HAVE TO BE DONE IN THIS DIRECTORY: you can add new sources, or copy a source from another src* directory and change it here. This is your directory, and it has the highest priority as compilation works. Since all your modifications are here, it will be easy to know what were your own developments, to be able to come back on them or to make them enter the next "official" release!...

util: contains SCM post-processing utilities.

4.3 Install

TO INSTALL THE SCM:

1. Put '.' (the current directory) in your PATH, with the highest priority. This is required since some scripts like 'install' have the same name on different SCM directories, and it is required during install or run processes to be sure that the local one is used.

2. If you want to use the SCM on one of the following machines: (HP / f90) or (SUN / f90 from FUJITSU), you simply have to type 'install all' (the script recognizes the machine architecture).

   If your architecture is not one of these, fill in the 'install' files from present, lfa, util and src/bin directories your compiler options before running 'install all'.

3. Put in the EDITOR environment variable the name of your usual editor, for example: export EDITOR=vi.

4. Put in the WINEDITOR environment variable the name of an editor that can be called in background mode, *i.e.* in a separate window, for example: export EDITOR=emacs, or dtpad, or crisp...

4.4 Run

'D' will stand below for the "root" SCM directory, *i.e.* the directory from which you have run the install process.

Go into the D/run directory, and HERE YOU ARE!... This directory contains relevant scripts and data to run the SCM on special issues, like a demonstration run, GATE, TOGA-COARE, make a validation of current SCM version against previous reference ones, etc...

In each directory you will find a 'mr' or 'mr_validate' script which runs the SCM. All those scripts will run the CURRENT version of the model, *i.e.* will use the D/src/bin/m1d.exe binary. In those scripts you will find

1. a namelist, defining SCM prediction parameters: time step, input profile name, forcings, etc...

2. the call to the model execution, done by invoking 'scm'.

You can choose normal mode or debug mode directly from command line: 'mr' or 'mr -d'.

4.5 File I/O: LFA software

The input and output data from the model are read or written on file through the LFA software: this software is designed to read or write real, integer or character arrays, on portable files (IEEE binaries), with a portable code. Articles in the files are accessed through their name. See in lfa directory for a documentation in postscript form (in english or french) to know how to read the resulting files or create your own ones.

4.6 Use resulting data

No graphic facility is given with SCM software. But some utilities, going from SCM put files to ASCII text ones, ready for graphics are available: you can so make this post-processing by different ways:

1. Use the 'm1d_pp' utility; this is the most synoptic one, to get a global view of the main facts which occured during the prediction: rise of the Sun, presence of clouds, precipitations, ... This utility has no argument (but the files names!...) and produces ASCII output files containing time and the "synoptic" fields. 'm1d_pp' with no argument gives help about the command.

2. Use 'm1d_evol': this is a way of producing on ASCII files time evolutions of any field present in the SCM LFA output files. 'm1d_evol' with no argument gives help about the command.

3. Get the budget of the prognostic variables. This can be done from DDH files produced by the SCM runs. A separate documentation is available for DDH use, please contact the authors.

4. Read the resulting files by your own way; this can be done using LFA software facilities, described in the LFA documentation referenced in page 17.

4.7 Create your own profiles and forcings

4.7.1 From Arpege-Aladin 3D models

It is sometimes useful to replay in the SCM a prediction done by Arpege-Aladin 3D models at a given location, in order either to study physical feedback or to check the role of interactive or non-interactive dynamics in the prediction.

A 3D prediction is done at a given place by the action of

1. Physics

2. Dynamics (which stands here for advection terms, adiabatic term from T equation, and baroclinic term from wind equation).

3. Semi-implicit time evolution

4. Time filtering

If you want to reproduce in the SCM the 3D prediction, you will need to force the SCM with the dynamics and semi-implicit from the 3D, since these two operators are unavailable in the SCM.

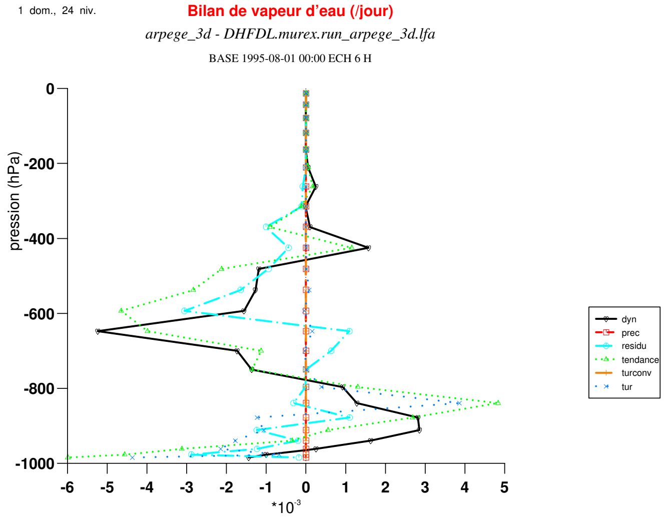


Figure 4.1: Water vapour budget of the Murex column in ARPEGE 3D model

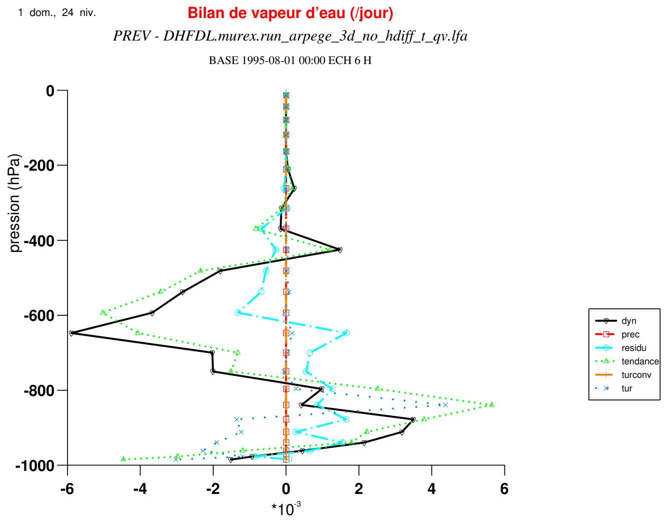


Figure 4.2: Water vapour budget of the Murex column in ARPEGE 3D model, with neither temperature nor water vapour horizontal diffusion

4.7.2 From observed or idealized cases

4.7.2.1 Vertical velocity

This field is computed by the SCM from the wind divergence field ζ, integrating vertically the equation:

$$\zeta = -\frac{\partial \omega}{\partial p} \tag{4.1}$$

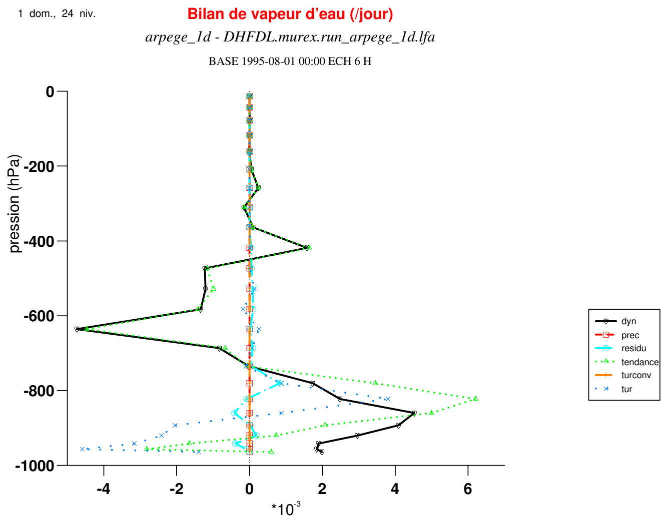


Figure 4.3: Water vapour budget of the Murex column in ARPEGE SCM forced by 3D dynamics only

ζ has to be given to the SCM in the file article "PDIVT0", in s^{-1}. If ζ is not available in your dataset, you should compute it from ω or w.

4.7.2.2 Temperature

If you choose the "NFT=4" option in NAMFORC, the file article "DYN-T" will have to contain the total dynamical tendency: horizontal advection + vertical advection + adiabatic conversion term, in K/s. In that case, the vertical velocity in pressure coordinate ω will not be used by the temperature prognostic equation.

If your dataset provides total dynamical θ tendency (K/s), that field is directly the one to be put in the article "DYN-T", since advection of s/c_p is nothing else but temperature advection + adiabatic conversion term.

4.8 Stretching

4.9 Deal with sources

4.9.1 Priority between sources, compile process

4.9.2 Which source version is used?

4.9.3 Locate text in the whole code

How to locate a given text (for example 'tstep') in the whole source code? Do 'mgrep tstep'. mgrep loops over all sources from 'bin' directory, which are, as explained upper, a copy of the highest priority ones.

4.9.4 Modify sources

Go in the D/src directory. To make a modification, for instance change tstep default value "mgrep tstep" will show you that tstep is initialized in sunam.F. "me sunam.F" will propose to you to copy this source on your own directory of modifications, and to edit it. When this is done, you compile your modifications and produce the new SCM executable file by simply typing "mc", and can rerun by "mr" from your usual run directory.

4.9.5 Remove sources

4.9.6 Phase in your modifications on a new cycle

4.9.7 Create a new reference cycle

WARNING: this section is a reminder for "SCM mainteners", *i.e.* people in charge with updating SCM sources to keep them consistent with Arpege-Aladin versions. The SCM users can simply skip over this section.

To create a new SCM reference version (called CYnew below) starting from a SCM version phased on an older Arpege-Aladin version (called CYold below), the following operations are to be done:

1. No item has to be in the "user" directory.
2. Check that every item from CYold/src_m1d_specific is present neither in CYold/src_arp_base nor CYold/src_arp_mod.
   This is done by the script D/src/old_cycle_to_new/unicity_of_specific_items.
3. Check that every item from CYold/src_arp_mod is present in CYold/src_arp_base.
   This is done by the script D/src/old_cycle_to_new/base_and_mod.
4. Check the modules: before the phasing with the new cycle it is useful to verify the following rules:
   (a) No unused module!...
   (b) The module sources have to be in the right directory, *i.e.* in the directory where the major part of references to this module are. For example a module used only by ECMWF physics should be in the ECMWF source directory!...

   This checking is done by the script D/src/old_cycle_to_new/check_modules. This script has to be run twice:
   (a) Firstly to look for unused modules in the bin directory. These unused modules can be removed with the D/src/old_cycle_to_new/rm_from_list script.
   (b) Secondly to check the src_arp_base, src_alt_ecmwf, (etc...) base directories. The unused modules from a directory can be moved to another where they are unsatisfied with the D/src/old_cycle_to_new/mv_from_list script.
5. Check the namelists: before the phasing with the new cycle it is useful to verify the following rules:
   (a) No unused namelist!...
   (b) The namelist sources have to be in the right directory, *i.e.* in the directory where the major part of references to this namelist are. For example a namelist used only by ECMWF physics should be in the ECMWF source directory!...

   This checking is done by the script D/src/old_cycle_to_new/check_namelists. This script has to be run twice:

   (a) Firstly to look for unused namelists in the bin directory. These unused namelists can be removed with the D/src/old_cycle_to_new/rm_from_list script.
   (b) Secondly to check the src_arp_base, src_alt_ecmwf, (etc...) base directories. The unused namelists from a directory can be moved to another where they are unsatisfied with the D/src/old_cycle_to_new/mv_from_list script.

6. Clean out CYold: "mc clean".

7. Compile and validate these modifications: no change has to be found at that stage in the SCM results!...

8. Create a global copy from SCM CYold version to the new one CYnew. Remove all CYnew/src_arp_base and src_arp_mod items.

9. Clean out CYnew: "mc clean".

10. How to fill in CYnew/src_arp_base: get a list L of CYold/src_arp_base items. Put in CYnew/src_arp_base the L items, coming from CYnew ARPEGE-ALADIN cycle.

11. How to fill in CYnew/src_arp_mod: the L list can be partioned into four lists. This partition is is done by the script D/src/old_cycle_to_new/partition; this scripts (and some of the following ones) reads CYold and CYnew directory paths on the CYold and CYnew files.
    (a) L_nothing is the list of items unmodified between CYold and CYnew ARPEGE-ALADIN cycles, and unmodified in CYold for SCM purpose (*i.e.* not present in CYold/src_arp_mod). This is the trivial case: nothing particular is supposed to be done.
    (b) L_cp is the list of items unmodified between CYold and CYnew ARPEGE-ALADIN cycles, and modified in CYold for SCM purpose (*i.e.* present in CYold/src_arp_mod). To deal with this case you simply need to copy the item from CYold/src_arp_mod to CYnew/src_arp_mod.
    This job is done by the script D/src/old_cycle_to_new/L_cp_action.
    (c) L_args is the list of items modified between CYold and CYnew ARPEGE-ALADIN cycles, and unmodified in CYold for SCM purpose (*i.e.* not present in CYold/src_arp_mod). The work to be done on these items will be
        i. To check changes in the dummy arguments of these items, as they are called from src_m1d_specific items.
        ii. To check changes in global variables, in order mainly to initialize these variables properly in the SCM setup.

    An help to this job is provided by the script D/src/old_cycle_to_new/L_args_action.
    (d) L_phase is the list of items modified between CYold and CYnew ARPEGE-ALADIN cycles, and modified in CYold for SCM purpose (*i.e.* present in CYold/src_arp_mod). The work to be done on these items will be a phase in, *i.e.* introduce the difference between CYold/src_arp_base and CYold/src_arp_mod in the CYnew/src_arp_base item. An help to this job is provided by the script D/src/old_cycle_to_new/L_phase_action.

12. What to do in CYnew/src_m1d_specific: check that the dummy arguments from the routines called from CYnew/src_m1d_specific did not change. If there is a change, phase in in the CYnew/src_m1d_specific on the "CALL" lines.

13. Compile and correct problems: these problems will come mainly from differences in modules, for example, a global variable used CYnew/src_m1d_specific, which disappeared in ARPEGE-ALADIN CYnew.

14. Run and check if results in free and forced modes are the same, or correspond to an expected evolution.

Chapter 5

History

1993: Initial version of this model, allowing running the physics on a single profile and time step (Eric Bazile and Jean-Marcel Piriou).

1996:
- Implementation of temporal loop (Eric Bazile and Jean-Marcel Piriou).
- Introduce Asselin filter and forcing environment, organize setups to be closer to 3D ones, compile with makefile utility (Francois Bouyssel).

1997: Usage of LFA software for SCM I/O (Jean-Marcel Piriou).

1998:
- Environment to deal with source phasing, automatic install, translate documentations to english, use of GATE and TOGA COARE Case 1 data, introduce new kinds of dynamical forcings: total or partial advections, impose variables, nudge variables (Jean-Marcel Piriou).
- Use of ECMWF and ARPEGE-CLIMAT parameterization schemes, introduce GNUmakefiles (Jozef Vivoda).