Ensemble Forecast

Polyphemus Training Session

December 11, 2009

About

Purpose: generation of ensembles, Monte Carlo simulations, ensemble combination

Author: Vivien Mallet, `vivien.mallet@inria.fr`, Damien Garaud, `damien.garaud@cerea.enpc.fr`

Polyphemus version: 1.6

Location: `http://www.enpc.fr/cerea/polyphemus/sessions.html`

Contents

Introduction

Extracting the archive creates a directory `ensemble/` in `polyphemus-sessions/`, which has also been created if necessary. Place the source `Polyphemus-1.6` in `polyphemus-sessions/` also.

Among Polyphemus specific features, there is uncertainty analysis through ensemble forecasts. This explains why the system is made of many parts (see Figure 1).

A simulation is defined by

- its input data (raw data such as land use cover);
- the physical parameterizations used to compute new fields, that is, the physical formulation;
- the numerical algorithms.

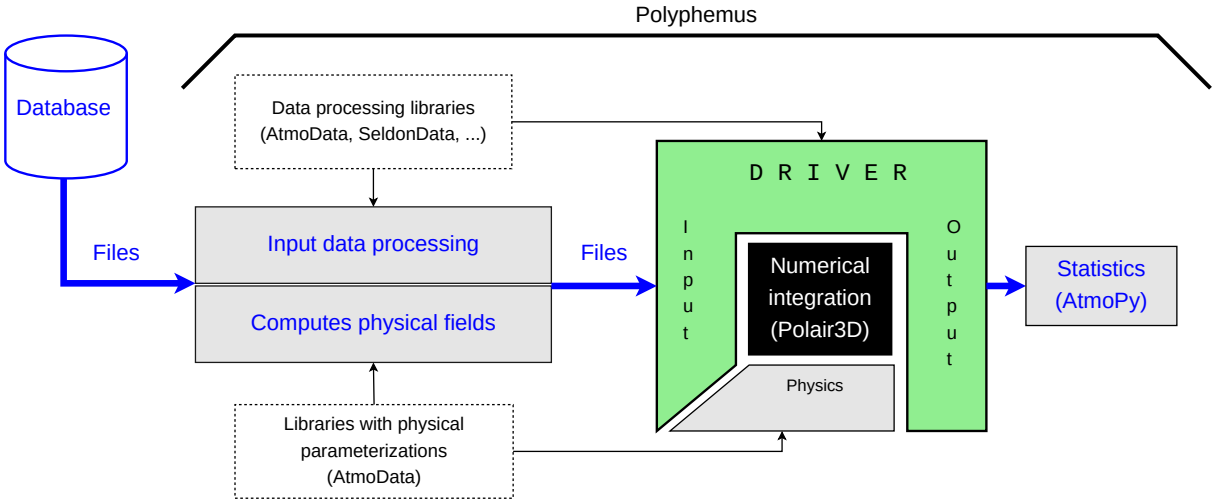

Figure 1: Polyphemus overall work flow. Data is first processed with dedicated C++ libraries (mainly SeldonData and AtmoData). The library AtmoData provides in addition most physical parameterizations needed in the preprocessing step. The preprocessing step output most coefficients of the reactive-transport equation. The numerical model (e.g., Polair3D) integrates in time this equation, and a driver manages the numerical model (e.g., to perform data assimilation). For post-processing (statistics, visualization, ...), Python scripts and the Python library AtmoPy are used.

Each part may be modified in Polyphemus with changes in preprocessing and in the model itself.

Preprocessing of data is split into several programs on purpose. It increases the flexibility of the system. Several programs can be written and executed in order to propose alternative parameterizations for a given field. For instance, vertical diffusion may be computed with Louis parameterization (in `preprocessing/meteo/MM5-meteo` or `preprocessing/meteo/Kz`) or with Troen&Mahrt parameterization (in `preprocessing/meteo/Kz_TM`). Furthermore, several input data are supported: primarily USGS and GLCF for land use cover, and ECMWF and MM5 for meteorological fields. Finally the programs have options that may be easily changed (minimum vertical diffusion, ground flux calculations, ...). Almost all parameterized fields are preprocessed; they are outside the transport model to preserve flexibility and save computation costs in ensemble simulations.

Polyphemus is platform built to include several models. These models can be included in ensemble forecasts. They can be plugged into drivers such as a Monte Carlo driver to further describe uncertainties.

In this training session, it will be shown how to create new ensemble members based on preprocessing choices (Section 1) and model changes (Section 2). Monte Carlo simulation will be performed in Section 3. Finally the ensemble forecasting through linear combination of models will be explained (Section 4).

1 Preprocessing

In this section, two parameterizations are replaced in the reference simulation:

- the vertical diffusion parameterization;

- the deposition parameterization.

1.1 Test Case

All configurations files used to generate the test case are in `polyphemus-sessions/ensemble/config/`. You may find configuration files from `luc-glcf.cfg` (`luc-glcf` is the very first step) to `polair3d.cfg` (for the simulation).

The commands of the preprocessing steps are provided in file `commands.txt`. The raw input data are not provided (boundary conditions, meterological files, ...), but one should be able to download them (possibly for an other period), following the user's guide instructions. You may use the MM5 file distributed with the Polyphemus test case (see `http://www.enpc.fr/cerea/polyphemus`, section "Polair3D Test Case").

Run the reference simulation with Polair3D, using `polair3d.cfg`.

1.2 Vertical Diffusion

The vertical diffusion coefficients have been generated with Louis parameterization by `MM5-meteo`. We now want to use Troen&Mahrt parameterization. *Generate vertical diffusion coefficients with Troen&Mahrt parameterization. Run the Eulerian model with it. Compare the results in IPython.* The file `polyphemus-sessions/ensemble/config/disp.cfg` has been provided for convenience.

1.3 Deposition Velocities

Deposition velocities have been generated with Zhang parameterization, with program `dep`. *Use this program to generate deposition velocities using Wesely parameterization.* You *only* need to change the value of the entry `Rc` in section `[Options]`.

Do not forget to change the path of the output files. You may save the new deposition velocities in `polyphemus-sessions/ensemble/data/dep/wesely/`. *Perform a simulation using Wesely deposition velocities and Louis parameterization for vertical diffusion. Generate results with Troen&Mahrt parameterization too.*

Since all preprocessing steps are well split, it is possible to make many changes from the reference configuration and to combine these changes. You may save the different results in new directories in `polyphemus-sessions/ensemble/results/`. Do not forget to change the path of the output files in `polyphemus-sessions/ensemble/config/polair3d-saver.cfg`.

2 Model Changes

Eulerian models include modules for transport (advection and diffusion) and chemistry. These modules may be changed: it is therefore possible to change the chemical mechanism or to change the numerical algorithms. We illustrate this with RADM. Open `polyphemus-sessions/Polyphemus-1.6/processing/radm/polair-radm.cpp`. This program is the same as `polair3d.cpp` except that it relies on RADM. *Compare `polair3d-radm.cpp` and `polair3d.cpp`. Compile it (`scons polair3d-radm`) and launch the simulation.* The main configuration file is `polyphemus-sessions/ensemble/config/polair3d-radm.cfg`.

If you are interested in the implementation, you may have a look in `polyphemus-sessions/Polyphemus-1.6/include/modules/chemistry/ChemistryRADM.cxx` (module) and in `polyphemus-sessions/Polyphemus-1.6/include/models/Polair3DChemistry.cxx` (model).

3 Monte Carlo Simulations

At model level, it is possible to take into account uncertainties in the input data with Monte Carlo simulations. A driver called `MonteCarloDriver` is used by `polyphemus-sessions/Polyphemus-1.6/processing/assimilation/polair3d-mc`. Its main configuration files are `polyphemus-sessions/ensemble/config-mc/polair3d-mc.cfg` and `polyphemus-sessions/ensemble/config-mc/perturbation.cfg`. *Open and read these files. Compile* ***`polair3d-mc`****.*

File `polair3d-mc.cfg` includes a section `[MonteCarlo]` where the number of members in the ensemble is set. Section `[save]` differs from usual output saver sections. It uses a saver unit of type `ensemble_prediction` which allows to save concentrations of all ensemble members.

File `perturbation.cfg` is described in the user's guide, in the section about the perturbation manager (drivers chapter).

Launch a Monte Carlo experiment. Perform other simulations with different numbers of members in the ensemble, and with different perturbations. It is advocated to save the results (copy directory `results/`) in order to compare them in IPython.

Display the results: time evolution of all members at some point in the domain, and map of the standard deviation of the ensemble. You should read and launch `polyphemus-sessions/ensemble/config-mc/load_ensemble.py`. The standard deviation of the ensemble may be computed with the AtmoPy function `stat.spatial_distribution`. Help about this function may be found in AtmoPy documentation (`http://www.enpc.fr/cerea/polyphemus/doc/atmopy/index.html`, module `atmopy.stat.miscellaneous`) or in IPython with command `help stat.spatial_distribution`.

4 Models Combination

Models combinations (superensembles, machine learning agregation) may be performed with AtmoPy. Help about these features may be found in AtmoPy documentation (`http://www.enpc.fr/cerea/polyphemus/doc/atmopy/index.html`, module `atmopy.ensemble`). There is no further description in this session because no ensemble forecast at hand can be distributed (rights limitations at meteorological level – ECMWF data).