Verdandi: generic library for data assimilation

July 2012

Introduction to Verdandi

- Co-developed with EPI MACS
- Licence : GNU LGPL
- Languages : C++, Python
- Multi platform : Linux, MacOS, Windows

Objective : providing methods and tools for data assimilation, designed to be relevant to a large class of problems involving high-dimensional numerical models (meteorology, oceanography, numerical heart modelling, ...).

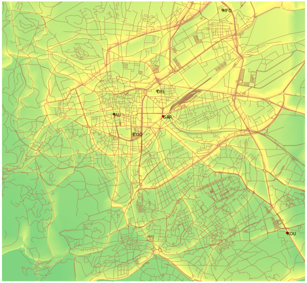
Benefits and users

- Faciliate the application of methods to a great number of problems.
- Provide a framework for perennial development, improving the diffusion of codes.

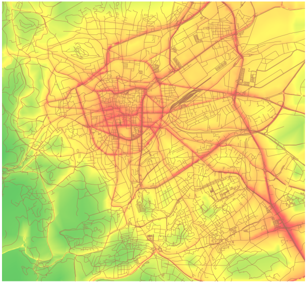
Potential users :

- Specialists who will take advantage of the robust framework.
- Non specialists who can directly use the available data assimilation methods in Verdandi.

NO2 concentration at Clermont-Ferrand (10/07/2008)



Forecast model without assimilation.



Forecast model with assimilation.

Conception

Simple example of a sequential data assimilation algorithm :
x_h^a is the analysis vector, computed from data assimilation A, x_h^f the background state vector. $\mathcal{M}_h$ is the numerical model.
For every time t_h, if observations o_h are available :

$$\begin{gathered} x_h^a = A(x_h^f, o_h) \\ x_{h+1}^f = \mathcal{M}_h(x_h^a) \end{gathered}$$

Assimilation methods A are generic and can be writter independantly of the system to which they are applied. They can therefore be put together in a library.

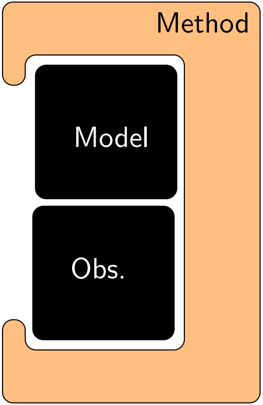
Some methods available in Verdandi : optimal interpolation, Kalman filters, four dimensional variational...

Conception

Language : C++

Three main objects

- A **model**
- An **observation manager**
- An **assimilation method** which uses the other two objects to compute the simulation



Python

With SWIG we can generate a high level Python interface.

```
In [1]: import verdandi
In [2]: oi = verdandi.Method1()
In [3]: method.Initialize(conf_file)
In [4]: oi.InitializeStep()
In [5]: method.Forward()
In [6]: method.Analyze()
```

Python interface

```
Verdandi::OptimalInterpolation<double,
Verdandi::Model<double>,
Verdandi::ObservationManager<double> >
driver;

driver.Initialize(conf_file);

driver.InitializeStep();
driver.Forward();
driver.Analyze();
```

C++ interface

Some main changes by version

0.9/1.0 (05/2011)

- Added an implementation of the 4D-Var method
- Largely improved the compatibility with Visual C++
- Large improvement of the online documentation

1.1 (10/2011)

- Added an implementation of the ensemble Kalman filter
- Made the SWIG interface compatible with Windows

1.2 (03/2012)

- Added the possibility to use a model and an observation manager written in Python
- Add of an interface for Petsc, which enables to manage parallel data structures in models and observations
- Added sequential aggregation for ensemble forecasting with discounted ridge regression

1.3 (06/2012)

- Final modifications on the model and observations interfaces

Current applications of Verdandi

- Image assimilation (EPI CLIME)
- Air quality (NUMTECH, INERIS, AirParif, IRSN)
- Sequential aggregation for meteorology (EPI CLASSIC, NUMTECH)
- Medical simulation (EPI MACS, EPI ASCLEPIOS, Stanford university, euHeart Project)
- Reduced minimax filter (Kiev univeristy, IBM research)

Some works in progress

- Creation of a test module to check if a model has been correctly interfaced with Verdandi
- Carry on the work on parallelization : methods should be able to run in parallel with parallel models
- Compatibility with the HDF5 format for input/output files
- Add some algorithms/tools specifics to image data assimilation