Building Ensemble-Based Data Assimilation Systems for High-Dimensional Models

Lars Nerger, Paul Kirchgessner
Alfred Wegener Institute for Polar and Marine Research
Bremerhaven, Germany
3 Components of an Assimilation System

Model
- initialization
- time integration
- post processing

DA method
- initialization
- analysis step
- ensemble transformation

Observations
- obs. vector
- obs. operator
- obs. error

mesh data/coordinates

state
- time

state
- observations

Ensemble-based Kalman Filter

Kalman filter: express probability distributions by mean and covariance matrix

EnKF (Evensen, 1994): Use ensembles to represent probability distributions
Offline Coupling – Separate Programs

Model

- Simple to implement
- Inefficient:
  - file reading/writing
  - model restarts

Assimilation program

For each ensemble state
- Initialize from restart files
- Integrate
- Write restart files

• Read restart files (ensemble)
• Compute analysis step
• Write new restart files

For each ensemble state
- Initialize from restart files
- Integrate
- Write restart files

Lars Nerger, Paul Kirchgessner – Building Ensemble DA Systems
Ensemble Filter Analysis Step

Analysis operates on state vectors (all fields in one vector)

Filter analysis
1. update mean state
2. ensemble transformation

Ensemble of state vectors
X

Vector of observations
y

Observation operator
H(…)

Observation error covariance matrix
R

For localization:
Local ensemble
Local observations

Lars Nerger, Paul Kirchgessner – Building Ensemble DA Systems
Online Coupling

Single program

Model
- initialization
- time integration
- post processing

Observations
- obs. vector
- obs. operator
- obs. error

Generic Assimilation Core

DA method
- initialization
- analysis step
- ensemble transformation

Explicit interface

Indirect exchange (module/common)

Lars Nerger, Paul Kirchgessner – Building Ensemble DA Systems
Extending a Model for Data Assimilation

**Model**

```
Start

Initialize Model
  generate mesh
  Initialize fields

Do i=1, nsteps

Time stepper
  consider BC
  Consider forcing

Post-processing

Stop
```

**Extension for data assimilation**

```
Start

init_parallel_DA

Initialize Model
  generate mesh
  Initialize fields

Init_DA

Do i=1, nsteps

Time stepper
  consider BC
  Consider forcing

Assimilate

Post-processing

Stop
```

**ensemble forecast enabled by parallelization**

*plus: Possible model-specific adaption.*

NEMO: Euler time step after assimilation
2-level Parallelism

1. Multiple concurrent model tasks
2. Each model task can be parallelized
   - Analysis step is also parallelized
   - Configured by “MPI Communicators”
Lars Nerger, Paul Kirchgessner – Building Ensemble DA Systems

Framework Solution with Generic Filter Implementation

Model with assimilation extension

Core-routines of assimilation framework

Case specific callback routines

Start

init_parallel_DA

Initialize Model

Init_DA

Do \( i = 1, n_{\text{steps}} \)

Time stepper

Assimilate

Post-processing

Stop

Generic

DA_Init
Set parameters
Initialize ensemble

DA_Model_Error
Subroutine calls or parallel communication

DA_Analysis
Check time step
Perform analysis
Write results

Read ensemble from files

Initialize state vector from model fields

Initialize vector of observations

Apply observation vector to a state vector

multiply R-matrix with a matrix

Dependent on model and observations

Case specific call-back routines
Assimilation Example with NEMO

Model configuration

- medium size SANGOMA benchmark
- box-configuration SQB (SEABASS)
- wind-driven double gyre
- 1/12° resolution
- 361x241 grid points, 11 layers

Sea surface height

Lars Nerger, Paul Kirchgessner – Building Ensemble DA Systems
PDAF: A tool for data assimilation

PDAF - Parallel Data Assimilation Framework

- provide support for parallel ensemble forecasts
- provide fully-implemented filter and smoother algorithms
- makes good use of supercomputers (Fortran with MPI & OpenMP parallelization)
- easily useable with (probably) any numerical model (coupled e.g. to NEMO, MITgcm, HBM, ADCIRC, FESOM)
- allows for separate development of model and assimilation algorithms

Open source:
Code and documentation available at
http://pdaf.awi.de

Add to *mynode* (lin_mpp.F90) just before init of myrank

```
#define key_USE_PDAF
   CALL init_parallel_pdaf(0, 1, mpi_comm_opa)
#define key_USE_PDAF
```

Add to *nemo_init* (nemogcm.F90) at end of routine

```
CALL init_pdaf()
```

Add to *stp* (step.F90) at end of routine

```
CALL assimilate_pdaf()
```

For Euler time step after analysis step:

Modify *dyn_nxt* (dynnxt.F90)

```
#define key_USE_PDAF
   IF((neuler==0 .AND. kt==nit000) .OR. assimilate)
#define key_USE_PDAF
```
Assimilation Example with NEMO - Observations

Observations – twin experiment
- Simulated satellite SSH (Envisat & Jason-1 tracks), 5cm error
- Temperature profiles on 3°x3° grid, 0.3°C error

Ensemble data assimilation
- Local ESTKF
- Assimilate each 48h

Case-specific routines utilize mesh information from Fortran modules of NEMO
Parallel Performance

- Speedup of NEMO-PDAF SEABASS 1/12° assimilation
- Ensemble size 32
- State dimension $\sim 3 \times 10^6$
- Speedup determined by speedup of NEMO
- Almost same speedup with assimilation
- Analysis step takes < 8% of total time (0.9s for largest case)
• Simulate a “model”
• Choose an ensemble
  • state vector per processor: $10^7$
  • observations per processor: $2 \cdot 10^5$
• Ensemble size: 25
• 2GB memory per processor
• Apply analysis step for different processor numbers
  ▪ 12 – 120 – 1200 – 12000
  ▪ Increase total state and obs. accordingly

• Very small increase in analysis time (~1%)
• Didn’t try to run a real ensemble of largest state size (no model yet)
Summary

- Online coupling more efficient than offline coupling
- Generic model interface for online ensemble data assimilation
- Minimal changes to model code
- Parallelization allows for ensemble forecasts
- Data assimilation framework PDAF (http://pdaf.awi.de) supports high-dimensional models
- Coding you own Ensemble Kalman filter usually not necessary
References

• http://pdaf.awi.de
