PDAF Parallel Data Assimilation Framework

ALFRED-WEGENER-INSTITUT
HELMHOLTZ-ZENTRUM FÜR POLAR-
UND MEERESFORSCHUNG

Ensemble-Based Data Assimilation: Concepts, Methods, and Hands-On Tutorials

Lecture 4: Practical aspects

Lars Nerger

Alfred Wegener Institute
Helmholtz Center for Polar and Marine Research
Bremerhaven, Germany

HELMHOLTZ
SPITZENFORSCHUNG FÜR
GROSSE HERAUSFORDERUNGEN

MarDATA Workshop on Data Assimilation, Bremerhaven, July 24-25, 2024

Overview – Lecture 4

Discuss some aspects relevant for the practical application of ensemble data assimilation

- Ensembles and how to generate them
- Observation operators
- Data assimilation software (PDAF)

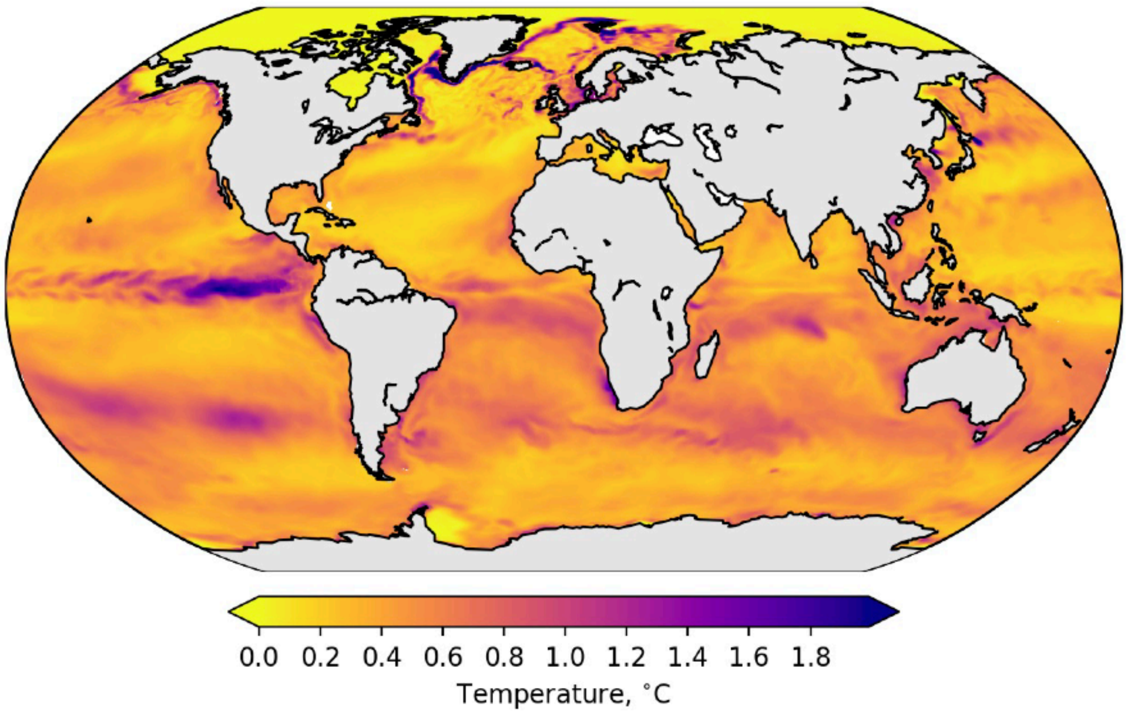
- Summary

Ensembles

Ensemble Covariance Matrix

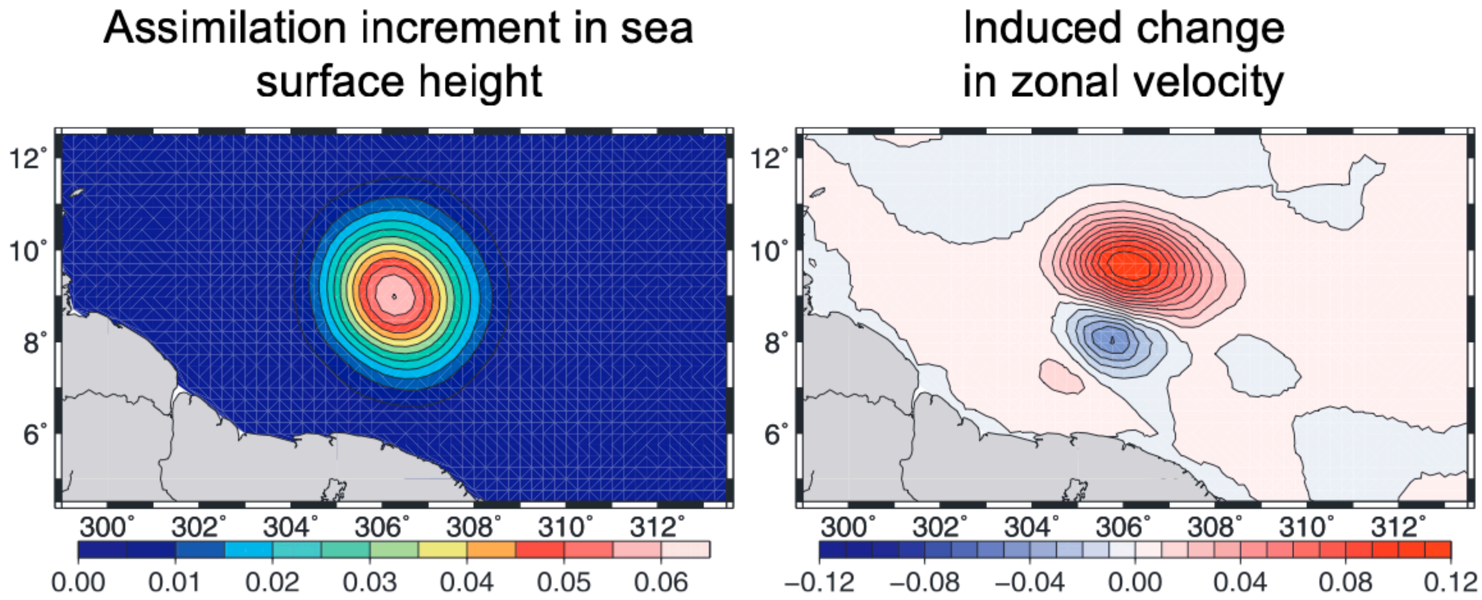
- Provide uncertainty information (variances + covariances)
- Generated dynamically
  by propagating ensemble of model states

SST: Ensemble standard deviation – March 1



Effect of cross-correlations – multivariate increments

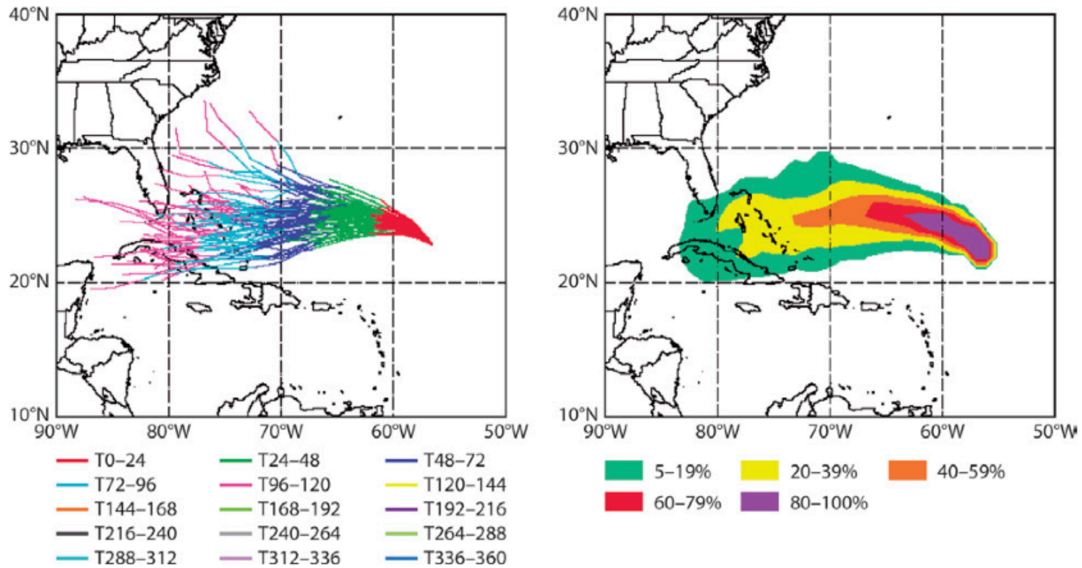
- Also:
  Provide information on error correlations
  (between different locations and different fields)
- Example: Assimilation of sea surface height
  (Brankart et al., Mon. Wea. Rev. 137 (2009) 1908-1927)



Ensemble Simulations

- Run model with different forcings, parameters, initial condition (or even run different models)
- Ensembles spread provides uncertainty information
- Can derive probabilities from ensemble distribution

Ensemble of hurricane tracks   Hurricane strike probabilities



Hurricane Ike, forecasts from 1200 UTC on 2008-09-04

Source: Bourgeault et al. BAMS 91 (2010) 1059

The Ensemble

- The ensemble
  - A set of N model state realizations: $\mathbf{x}_i^{(1)}, \ldots, \mathbf{x}_i^{(N)}$
  - Ensemble matrix $\mathbf{X}_i = \left(\mathbf{x}_i^{(1)}, \ldots, \mathbf{x}_i^{(N)}\right)$

Ensemble represents state estimate and its uncertainty

$$\mathbf{X}_i = \overline{\mathbf{X}}_i + \mathbf{X}'_i \qquad \text{with} \quad \mathbf{X}'_i = \mathbf{X}_i - \overline{\mathbf{X}}_i$$

(Each column of $\overline{\mathbf{X}}_i$ holds the ensemble mean)

Need to initialize 2 parts:

- $\overline{\mathbf{X}}_i$ ensemble mean – or *central state* — State estimate
- $\mathbf{X}'_i$ ensemble perturbations — Uncertainty estimate

Important: States in $\mathbf{X}_i$ need to be realistic realizations

The Initial Ensemble – Central State

Initial ensemble $\mathbf{X}_0$ represents uncertainty and state at initial time

Initial central state $\overline{\mathbf{x}}_0$

- Will be the initial ensemble mean state
- Choose it freely as your best estimate
  - E.g. from operational model run

With regard do data assimilation

- A 'good' state is difficult to improve, but it's realistic
- A 'bad' state is easy to improve, but the result might still have high error (DA studies in the past sometimes used a long time mean)
- Generally: Improving the state estimate is not a success on its own

The Initial Ensemble – Ensemble Perturbations

Ensemble perturbation $\mathbf{X}_0'$ represent uncertainty (error) in state

Sample covariance matrix

$$\mathbf{P}_i = \frac{1}{N-1}\left(\mathbf{X}_i - \overline{\mathbf{X}}_i\right)\left(\mathbf{X}_i - \overline{\mathbf{X}}_i\right)^T$$

- **variance**: uncertainty of each value
- **covariances**: relation of errors of different variables or at different grid points

From the ensemble we can compute any of the components of P:

variances: $\mathbf{P}_{ij}, i = j$

covariances: $\mathbf{P}_{ij}, i \neq j$

correlations: $\frac{\mathbf{P}_{ij}}{\sqrt{\mathbf{P}_{ii}\mathbf{P}_{jj}}}$

We intent to obtain $\mathbf{X}_0'$ from model dynamics

- unlike parameterized covariances in 3D-Var
- Provide uncertainty and error covariance at analysis time

The Initial Ensemble – Sampling possibilities (I)

Possibility 1: **Sample directly from model trajectory**

- select model states from a simulation:
    - Choose model states systematically (e.g. January 1 from various years)
    - Choose model states randomly (e.g. randomly from December to February from some year)

Advantages

- Each state is physically balanced
- Cross-covariances between different fields from model dynamics

Disadvantages

- Difficult to represent the uncertainty
- Slow convergence
- Replacing sample mean by central state can lead to unbalanced states

The Initial Ensemble – Sampling possibilities (2)

Possibility 2: **Generate perturbations dynamically**

1. Perturb initial state $\tilde{\mathbf{x}}_0^{(i)} = \mathbf{x}_0 + \delta\mathbf{x}_0^{(i)}$
2. Do a short model run (few days) with original initialization
3. Do a short model run (few days) with perturbed initialization
4. Perturbation *i* is given by difference $\mathbf{x}_k'^{(i)} = \tilde{\mathbf{x}}_k^{(i)} - \mathbf{x}_k$
5. Repeat to obtain *N* perturbations

Different schemes have been proposed on this basis, e.g.

- **NMC method** (Parrish & Derber, 1992)
  - Short-term forecasts
- **Bred vectors** (Toth & Kalnay, 1993)
  - Sequence of short forecasts with rescaling of perturbations (‚breeding' of perturbations; finite-time Lyaponov exponents)

The Initial Ensemble – Sampling possibilities (3)

Possibility 3: **Use model state variability**

Our standard method in PDAF

Second-order exact sampling from EOFs

1. Perform a model run over sufficient time period (or use one at hand), store snapshots of model states $\mathbf{Z} = \mathbf{z}_1, \ldots, \mathbf{z}_M$
2. Subtract a suitable mean $\mathbf{Z}' = \mathbf{Z} - \overline{\mathbf{Z}}$
3. Perform an SVD $\mathbf{Z}' = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T$ — $\mathbf{U}$ holds the EOFs
4. Specify ensemble size *N (≤ M+1)*
5. Generate a random matrix $\mathbf{\Omega}$ of size *N x N-1* whose columns are orthonormal and orthogonal to the vector $(1, \ldots 1)^T$
6. With the first *N-1* columns of $\mathbf{U}$ compute
$$\mathbf{X}' = \sqrt{N-1}\,\mathbf{U}\mathbf{\Lambda}\mathbf{\Omega}^T$$

$\mathbf{\Omega}$ can be obtained iteratively with orthogonal projections (Hausholder reflections; we have code for this)

The Initial Ensemble – Sampling possibilities (4)

Advantages of second-order exact sampling

- The method explicitly computes a square root of the covariance matrix (Gaussian assumption)
- EOFs $\mathbf{U}$ are eigenvectors of model operator

  → Important are eigenvectors with eigenvalue > 1
  these are unstable directions of the dynamics

- One can precompute the EOFs to be able to generate ensembles up to size *M+1* later
- EOFs yield best low-rank approximation for $\mathbf{P}$

Disadvantage

- Perturbations do not account for physical balances

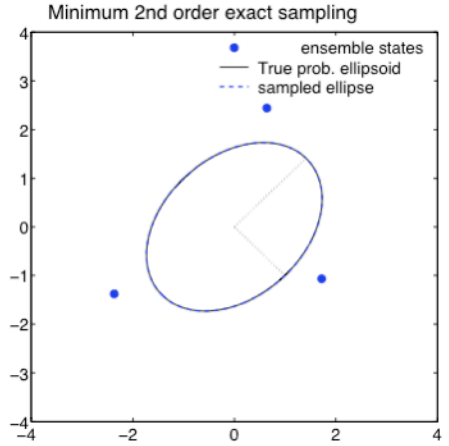
Sampling Example

Example matrix and state

$$\mathbf{P}_t = \begin{pmatrix} 3.0 & 1.0 & 0.0 \\ 1.0 & 3.0 & 0.0 \\ 0.0 & 0.0 & 0.01 \end{pmatrix}; \quad \mathbf{x}_t = \begin{pmatrix} 0.0 \\ 0.0 \end{pmatrix}$$

2nd order exact sampling

➜ rank 2 matrix is exactly sampled using 3 state realizations

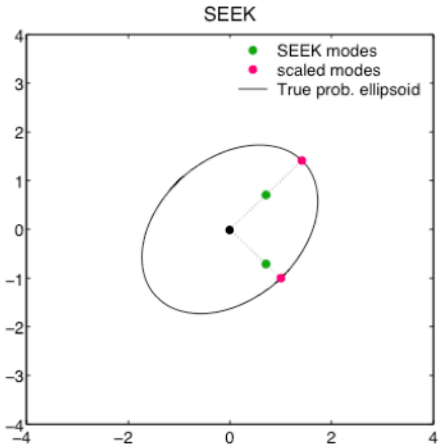


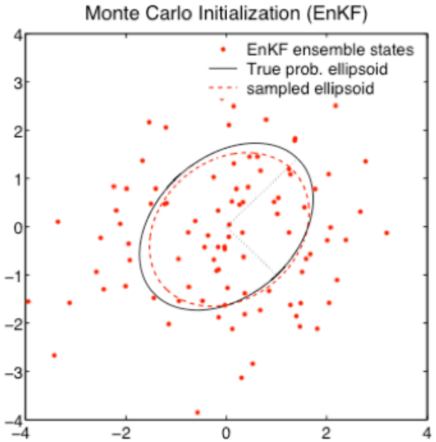
Same as spherical simplex sampling (Wang et al., 2004)

Some possible samplings

Eigenvectors

ensemble size N=r+1;
not an ensemble of
equivalent states



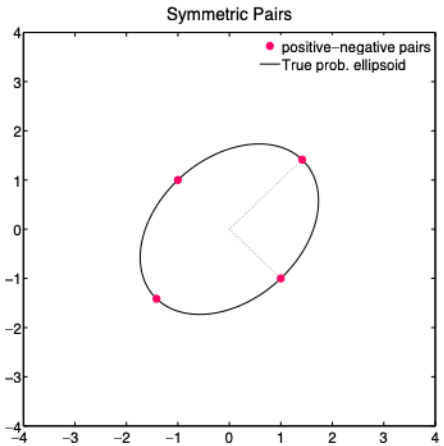


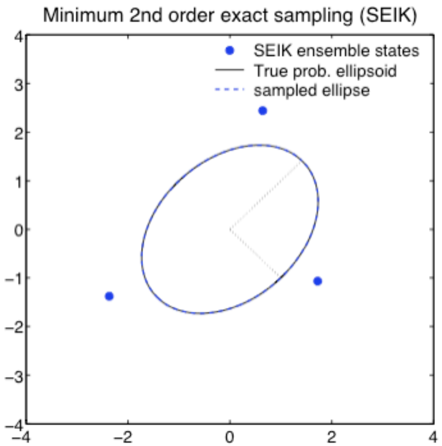
Random sampling

slow convergence;
needs large ensemble;
equivalent states

Symmetric pairs

ensemble size N=2r;
not an ensemble of
equivalent states





2nd-order exact sampling

ensemble size N=r+1;
convergence depends on
eigenvalues;
equivalent states

The Ensemble Size

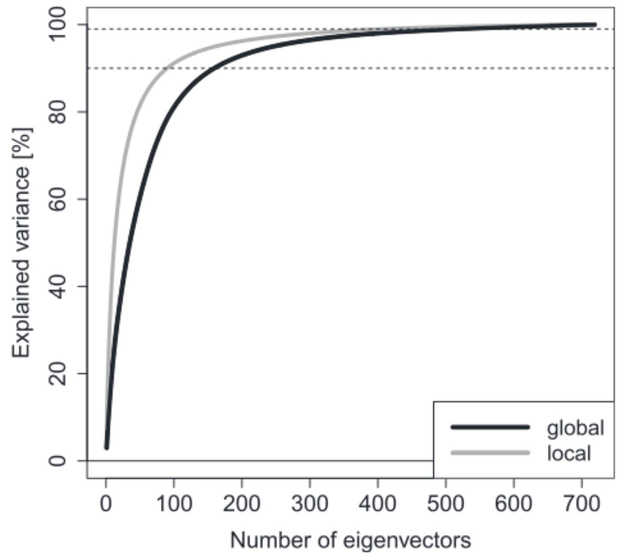
Which ensemble size is ‚correct‘

Ensemble size determines sampling quality of covariance matrix

Some insights

- Ensemble should cover the unstable directions/modes or unstable subspace of model dynamics
- eigenvalues of EOFs can give indication
  - Common argument in papers ~15-20 years ago: A certain ensemble size contains e.g. 90% of the variability
  - But this says nothing about sampling quality
    - in particular of cross-covariances
    - variances can be to low or too high; covariances can have wrong sign

Eigenvalues in case of NEMO double gyre



Which ensemble size is ‚correct' (II)

Ensemble size in practice

- Published studies use between 4 and ~200 members
  (there are now also cases with ~10 000 members, but exceptions)
- Determine ensemble size experimentally:
  - There will be a minimum limit to overall functioning
    (perhaps, never go below 8, but this is only experience
    and Liang et al (2017) used N=4 for successful DA of sea ice)
  - Further increased size will lead to incremental improvements
    (But there can be steps in the improvement if error in some cross-covariance is significantly improved)
  - Variances are easy to sample; covariances more difficult;
    cross-covariances between different fields even more difficult
  - We typically use between 20 and 50 members
    (e.g. with coastal application HBM-ERGOM we saw better subsurface updates with N=40 instead of N=20)

Observation operators and errors

Observation Operator

Obervations: $\mathbf{y} \in \mathbb{R}^m$ (contains different observed fields)

Observation equation (relation of observation to state $\mathbf{x}$):

$$\mathbf{y}_k = H_k\left[\mathbf{x}_k\right] + \epsilon_k$$

ϵ_k : observation error

Linear Observation Operators

Linear observation operators $\mathbf{Hx}$ – examples:

- Model value at a grid point
- Average of model values at some grid points
- Interpolation from model grid to observation location
- Sum (integration) of model values

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$$

Let $\mathbf{y} = \begin{pmatrix} \text{Average of } x_1 \text{ and } x_2 \\ \text{Observation of } x_4 \end{pmatrix}$

Observation operator?

$$\mathbf{H} = \begin{pmatrix} 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Nonlinear Observation Operators

Non-linear observation operators

$$H = \begin{pmatrix} x_1^2 \\ \sin(x_2) \\ \sqrt{x_3^2 + x_4^2} \end{pmatrix}$$

Now ***H*** is nonlinear operator, no matrix

- Common for atmospheric observations (radiances)
- Most operations in ocean are linear
- Nonlinear H has to be applied to state value, not increment

Nonlinearity can have implications on performance of assimilation scheme

BLUE assumes Gaussian errors ➜ not fulfilled with nonlinear ***H***

Relation of state vector and observations

Observation operator

- maps from state vector to observation vector

Requirements

- fields needed for $\boldsymbol{H}$ have to be stored in $\boldsymbol{x}$
- information how fields are stored in state vector
- Interpolation also needs coordinate information

Observation errors ϵ_k

ϵ_k contains two parts:

Measurement errors	Representation errors

Measurement errors

Measurement is never perfect

E.g. measure temperature

- At home with digital thermometer
  - error +/- 0.1 °C
- SST from satellite
  - larger error (> +/-0.3 °C)

(satellite measures radiation)

Representation errors

Measurement and model do not represent the same

- Ocean models have resolutions between ~900m (HBM) and ~150 km (global)
- In situ measurement is local
- Satellite has certain footprint

➜ Additional error

Assimilation Software

Computational and Practical Issues

- Running a whole model ensemble is costly
- Ensemble propagation is naturally parallel (all independent)
- Ensemble data assimilation methods need tuning
- No need to go into model numerics (just model forecasts)
- Assimilation analysis step only needs to know:
  - Values of model fields and their location
  - Observed values, their location and uncertainty
- We need to handle large matrices and a large amount of data,

  → Require optimized and parallelized implementation

Ensemble data assimilation can be implemented
in form of a generic code
+ case-specific routines

PDAF: Parallel Data Assimilation Framework

PDAF Parallel Data Assimilation Framework

A unified tool for interdisciplinary data assimilation …

- a program library for data assimilation
- provide support for parallel ensemble forecasts
- provide assimilation methods – fully-implemented & parallelized
- provide tools for observation handling and for diagnostics
- easily useable with (probably) any numerical model
  (coupled to with range of models)
- run from laptops to supercomputers (Fortran, MPI & OpenMP)
- Usable for real assimilation applications and to study assimilation methods
- ensure separation of concerns (model – DA method – observations – covariances)

Open source:
Documentation and tutorial at

http://pdaf.awi.de

github.com/PDAF

Python interface:

https://github.com/yumengch/pyPDAF

L. Nerger, W. Hiller, Computers & Geosciences 55 (2013) 110-118

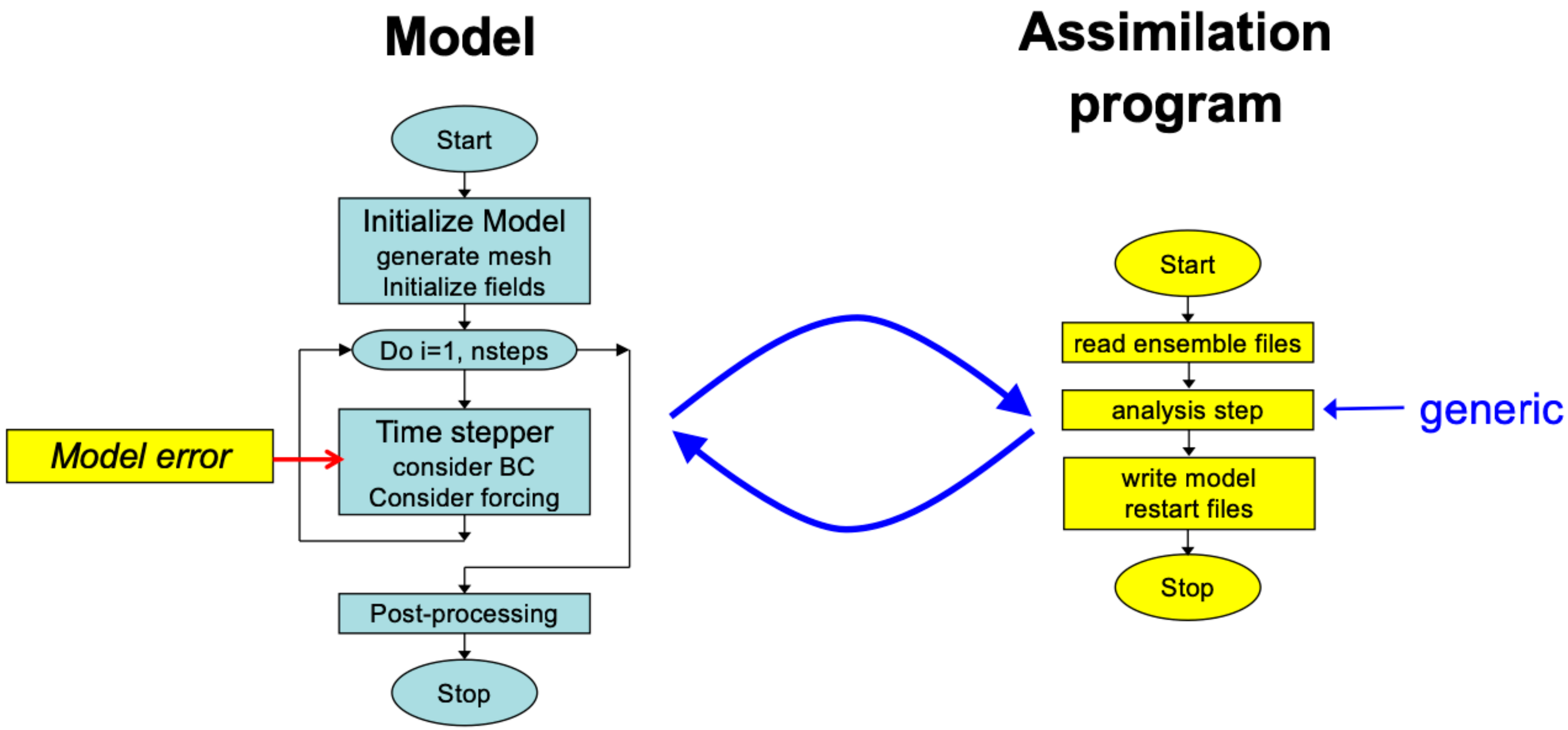
Framework design

- Parallelization of ensemble forecast can be implemented independently from model
- Analysis step can be implemented independently from model (run it providing state vector and observational information)

Goals for a model-independent framework

- Simplify implementation of data assimilation systems based on existing models
- Provide parallelization support for ensemble forecasts
- Provide filter algorithms (fully implemented & parallelized)
- Provide collection of „fixes“ for filters, which showed good performance in studies

Offline coupling – separate programs



For each ensemble state

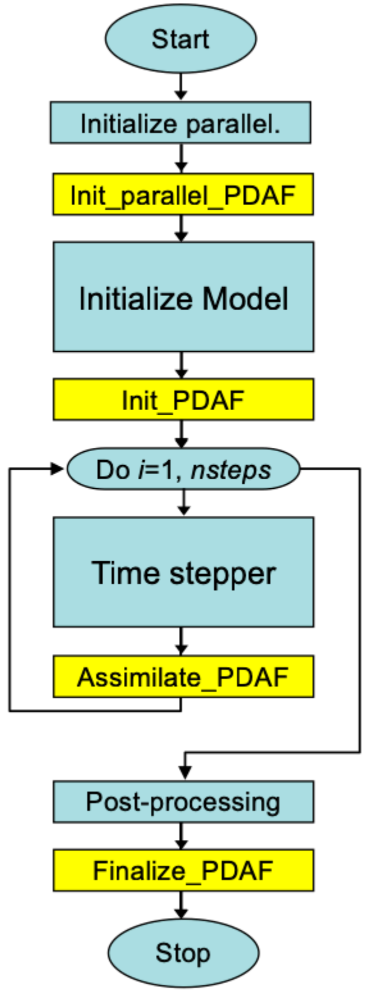
- Initialize from restart files
- Integrate
- Write restart files

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

Online coupling - Augmenting a Model for Data Assimilation

revised parallelization enables ensemble forecast

Data assimilation: run model with additional options



Model

Extension for data assimilation:

4 subroutine calls

plus:
Possible model-specific adaption

e.g. in NEMO: treat leap-frog time stepping

Online and Offline modes

Offline

- Separate programs for model and filter
- Ensemble forecast by running sequence of models
- Analysis by assimilation program
- Data exchange model-filter by files on disk

- *Advantage:*
  Rather easy implementation
  (file reading/writing routines, no change to model code)
- *Disadvantage:*
  Limited efficiency, cost of file reading & writing; restarting programs

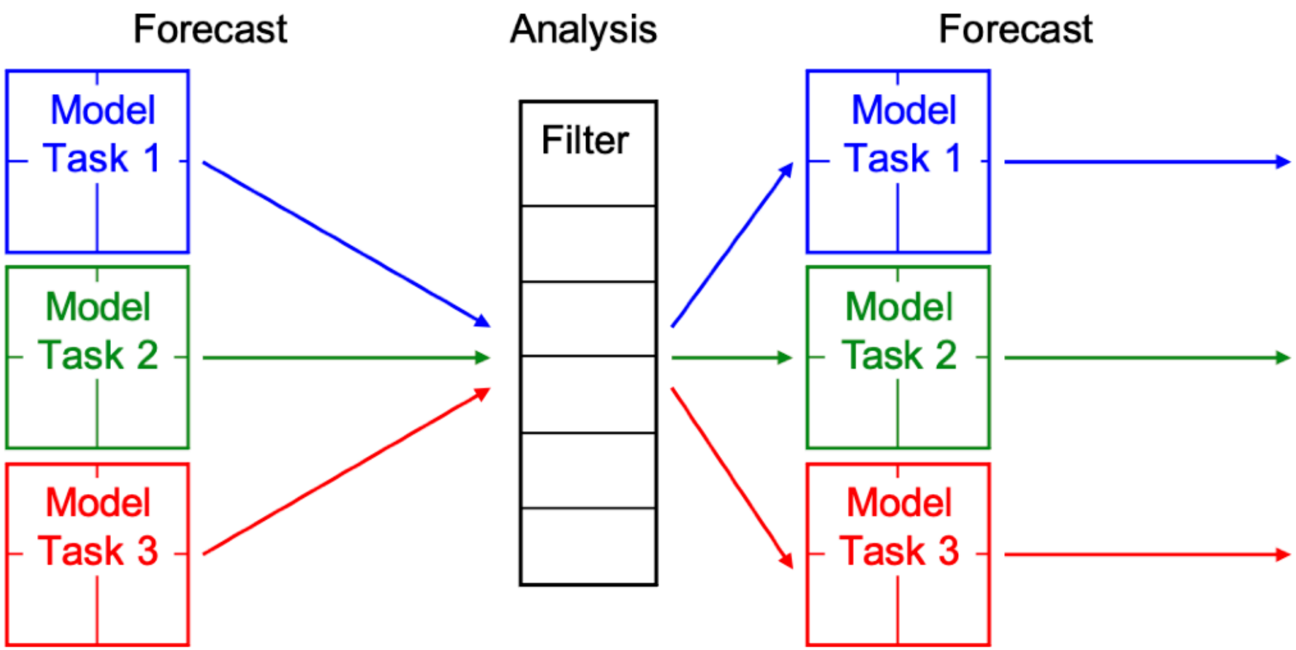
Online

- Couple model and filter into single executable program
- Run single program for whole assimilation task (forecasts and analysis)
- Data exchange model-filter in memory

- *Advantage:*
  Computationally very efficient
  (less file outputs, no full program restarts)
- *Disadvantage:*
  More implementation work, incl. extension of model code

2-Level Parallelism

PDAF Parallel Data Assimilation Framework



1. Multiple concurrent model tasks
2. Each model task can be parallelized

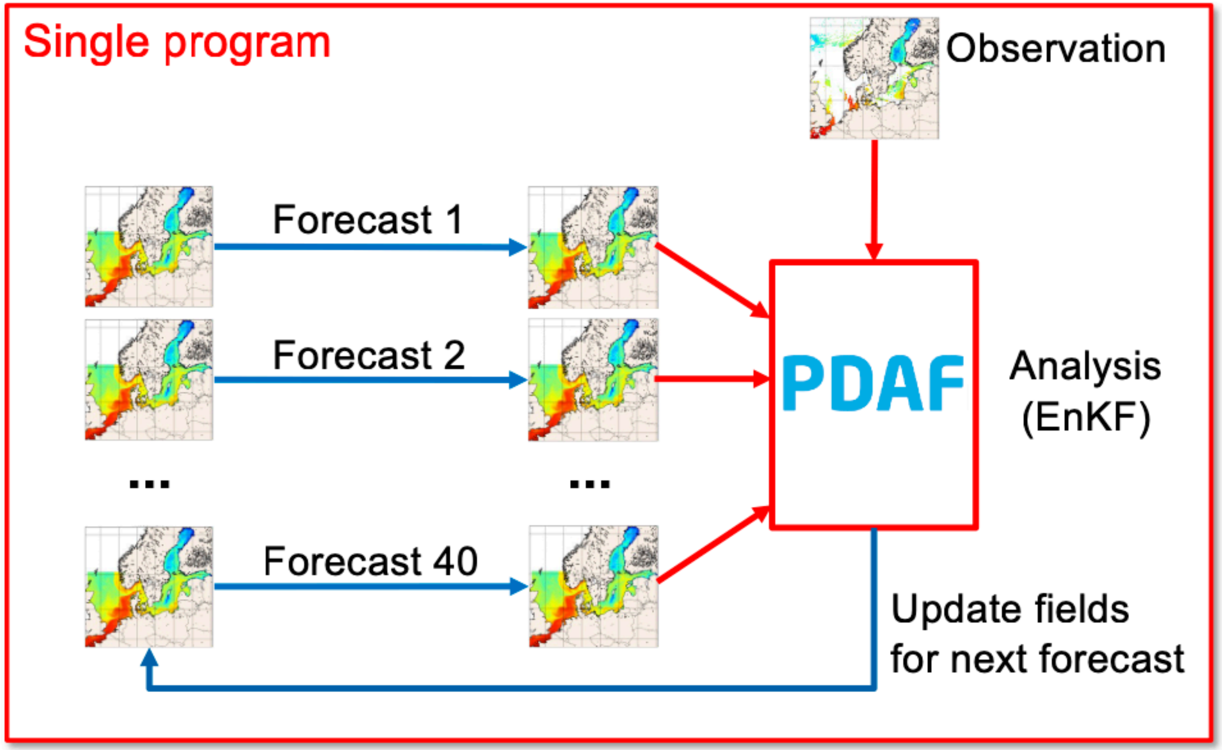
➢ Analysis step is also parallelized

MPI communicators initialized in routine *init_parallel_pdaf*

Assimilation-enabled Model

Couple a model with PDAF

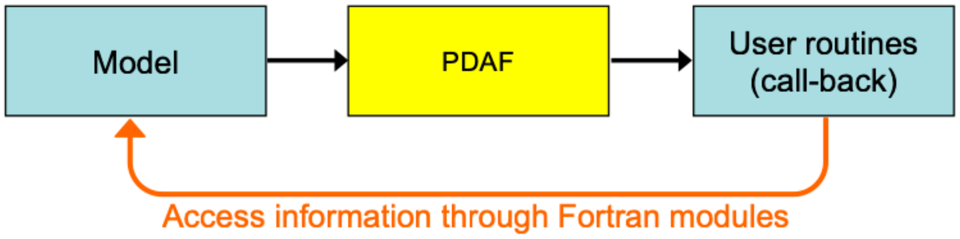
- Modify model to simulate ensemble of model states
- Insert analysis step/solver to be executed at prescribed interval
- Run model as usual, but with more processors and additional options



Initialize ensemble | Ensemble forecast | Analysis step in between time steps

PDAF interface structure

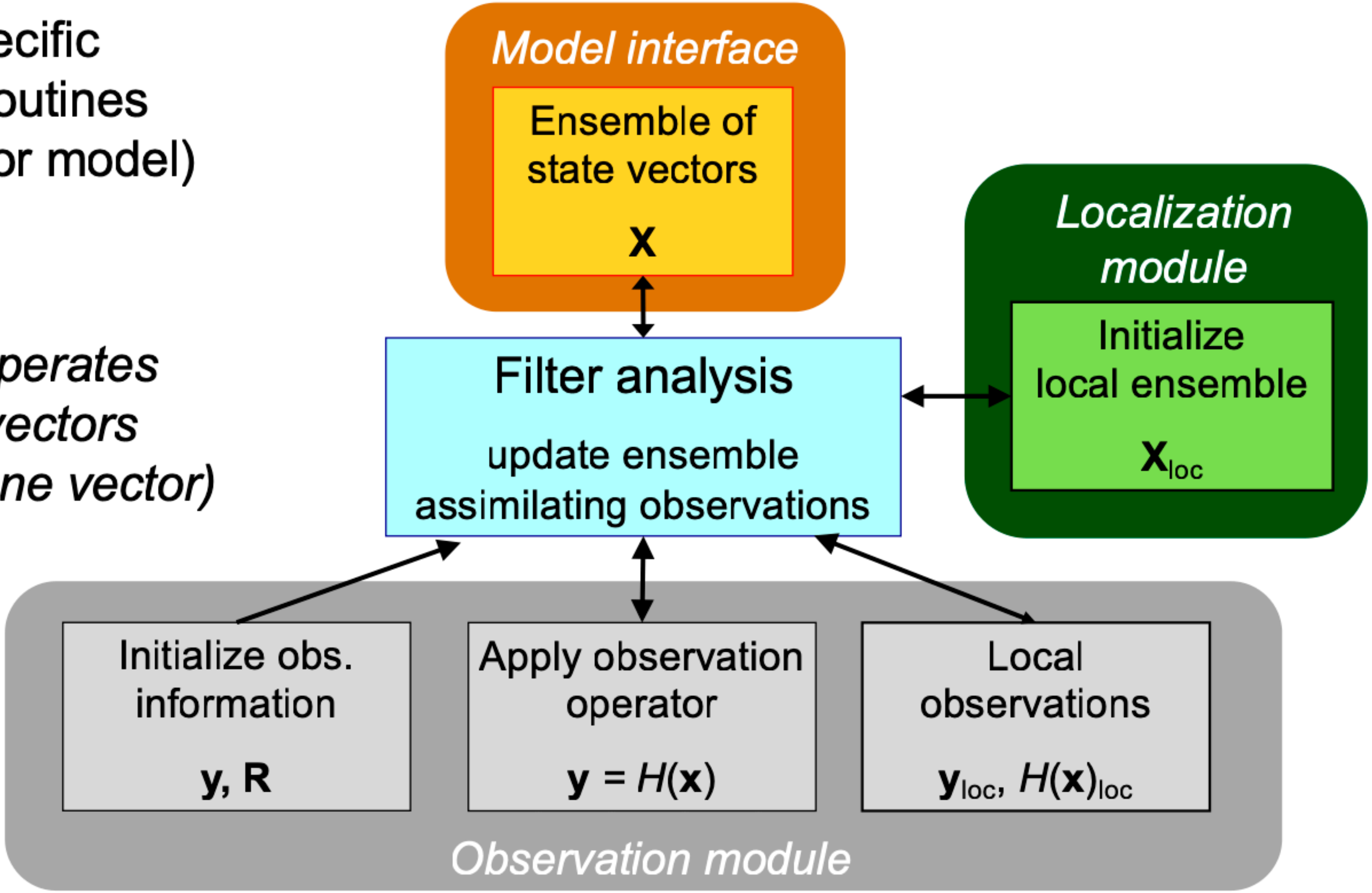
- Interface routines call PDAF-core routines
- PDAF-core routines call case-specific routines provided by user (included in model binding set)
- User-supplied call-back routines for elementary operations:
  - field transformations between model and filter
  - observation-related operations
- User supplied routines can be implemented as routines of the model



Implementing Ensemble Filter Analysis Step

case-specific
call-back routines
(implement for model)

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



pyPDAF

- Python interface to PDAF
  - Developed by Yumeng Chen, University of Reading, UK
  - Coded using Cython
- Driver and user routines coded in Python
- Particularly useful if model is coded in Python
- Supports online and offline coupling

Fresh development:

➜ we don‘t know performance for high-dimensional cases yet

➜ ideal Python implementation is still in progress

pyPDAF:
https://github.com/yumengch/pyPDAF

Chen, Y., L. Nerger, and A. S. Lawless (2024) A Python interface to the Fortran-based Parallel Data Assimilation Framework: pyPDAF v1.0.0, submitted to GMD, doi:10.5194/egusphere-2024-1078, 2024

Summary

Summary 1

Data Assimilation

- combines observations and dynamics models in a quantitative way
- Allows models to learn from observations
- Can be applied whenever there is a dynamical model and related observations

Ensemble Data Assimilation

- Utilize ensemble of model state realization to estimate state and its uncertainty
- Estimates are dynamic (‘errors of the day’)
- Ensemble integration is costly to run

Summary 2

Mathematical basis

- estimation (probabilities and Bayes law)
  or optimization (minimization)
- Kalman filters assume Gaussian error distributions
  for optimality

Practical Ensemble Data Assimilation

- Use advanced ensemble Kalman filters like ESTKF
- Need to utilize 'fixes' like inflation and localization
- Problem can be parallelized and can efficiently use
  supercomputers

Many things we didn't have time for ….

- Parameter estimation and observation system optimization
- Nonlinear (non-Gaussian) data assimilation
- Methods in machine learning are very related

There is software for applying DA!

Literature

Books:

- Evensen, G., F. Vossepoel, P. J. van Leeuwen, *Data Assimilation Fundamentals*, Springer, 2022 (online open access)
- Asch, M, M. Bocquet, M. Nodet, *Data Assimilation: Methods, Algorithms, and Applications*, SIAM, 2017 (not too mathematical)
- Reich, S. and C. Cotter, *Probabilistic Forecasting and Bayesian Data Assimilation*, Cambridge University Press, 2015 (mathematical)

Journal Articles:

- S. Vetra-Carvalho et al. (2018). *State-of-the-art stochastic data assimilation methods for high-dimensional non-Gaussian problems*. Tellus A **70:1**(2018) 1445364 (good reference for algorithms)
- Carrassi, A ., M. Bocquet, L. Bertino, G. Evensen (2018). *Data assimilation in the geosciences: An overview of methods, issues, and perspectives*, WIREs Climate Change. 2018;9:e535