PDAF Parallel Data Assimilation Framework

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

PDAF – Community Software for Ensemble-based Data Assimilation

Lars Nerger

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

HELMHOLTZ
SPITZENFORSCHUNG FÜR GROSSE HERAUSFORDERUNGEN

ISDA-Online, January 14, 2022

PDAF – Parallel Data Assimilation Framework

PDAF Parallel Data Assimilation Framework

A universal tool for ensemble data assimilation …

- provide support for parallel ensemble forecasts
- provide assimilation methods (solvers) - fully-implemented & parallelized
- provide tools for observation handling and for diagnostics
- easily useable with (probably) any numerical model
- a program library (PDAF-core) plus additional functions
- run from notebooks to supercomputers (Fortran, MPI & OpenMP)
- usable for real assimilation applications and to study assimilation methods
- welcoming community contributions

Open source:
Code, documentation, and tutorial available at

http://pdaf.awi.de

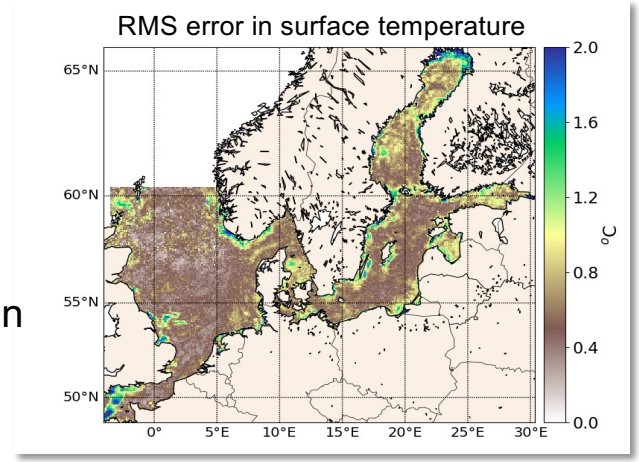
L. Nerger, W. Hiller, Computers & Geosciences 55 (2013) 110-118

PDAF Application Examples

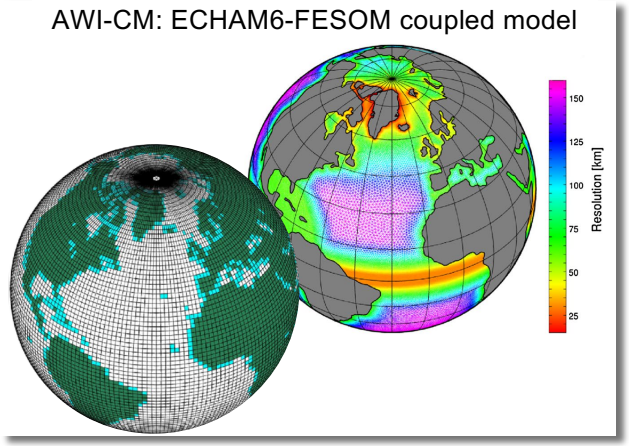
PDAF Parallel Data Assimilation Framework

At AWI:

HBM-ERGOM:
coupled physics/
biogeochemistry
coastal assimilation
in nested model



AWI-CM:
coupled ocean-
atmosphere
assimilation



External applications & users, like

Operational uses:

- *Germany:* North/Baltic Seas (HBM model)
- *Europe:* Copernicus Baltic forecasting center (NEMO)
- *China:* Arctic ice-ocean prediction system (MITgcm)

Ocean and climate models (research applications)

- **NEMO/AGRIF**
- **SCHISM/ESMF**
- **MPI-ESM**
- **COAWST** (Coupled Ocean-Atmosphere-Wave-Sediment Transport modeling system)

Beyond ocean

- **TSMP-PDAF** (Terrestrial Systems Modeling Platform)
- **TIE-GCM** (Thermosphere Ionosphere Electrodynamics GCM)
- **VILMA** (Viscoelastic Lithosphere and Mantle Model)
- **Parody** (Geodynamo model)
- **HYSPLIT** (Volcanic Ash Transport and Dispersion model)
- … more

PDAF: User-friendliness

PDAF Parallel Data Assimilation Framework

Goal: Enable easy and fast setup of a DA system
and allow for extension to fully featured system
while ensuring high efficiency and scalability

Assumption: Users know their model

➜ let users implement DA system in model context

For users, model is not just a time-stepping operator

➜ let users extend their model for data assimilation

Keep code simple for the user side:

➜ Define subroutine interfaces to DA code based on arrays
(also simplifies interaction with languages like C/C++/Python)

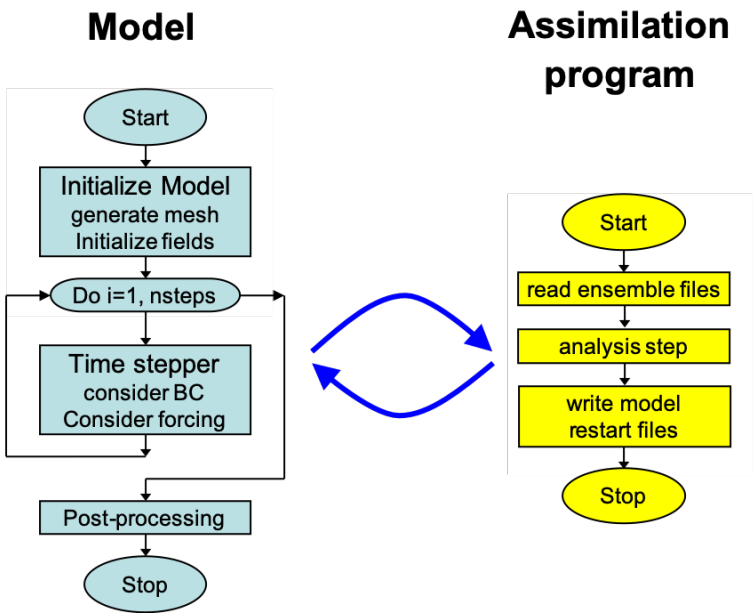
➜ No object-oriented programming
(most models don't use it; most model developers don't know it;
many objects we would only have for observations – see later)

➜ Users directly implement case-specific routines
(no indirect description (XML, YAML, ...) of e.g. observation layout)

Coupling Model and Assimilation Code: 2 Variants

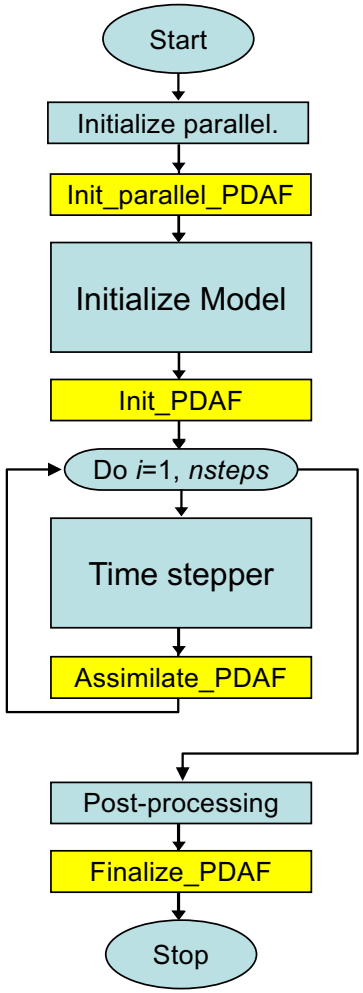
PDAF Parallel Data Assimilation Framework

Offline



- Separate programs for model and DA
- Flexible to run
- Needs frequent model restarts and file output (less efficient than online coupling)

Online



- Augment model with DA functionality
- Insert 4 subroutine calls
- Very efficient & highly scalable

Model code | DA code

PDAF interface structure

PDAF Parallel Data Assimilation Framework

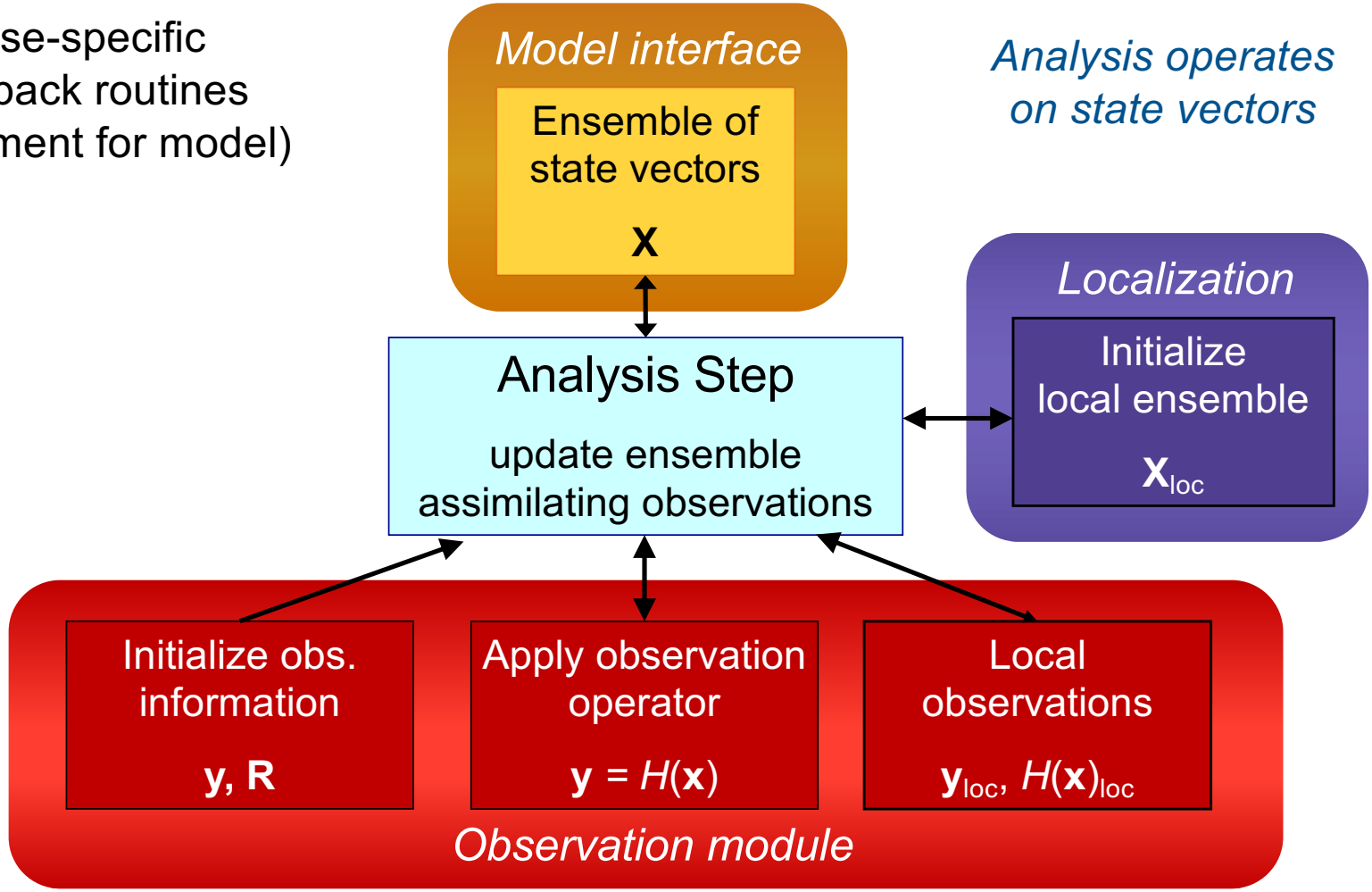
- **Model-sided Interface:** Defined calls to PDAF routines
  (called by driver program for offline coupling)
- **Case-related Interface:** User-supplied call-back routines for elementary operations:
  - transfers between model fields and ensemble of state vectors
  - observation-related operations
- **Internal Interface:** Connect to data assimilation methods
- User supplied routines can be implemented as routines of the model and can share data with it (low abstraction level)

Online: Access model information through modules / static variables
offline: initialize model information from files

Implementing the Ensemble Filter Analysis Step

PDAF Parallel Data Assimilation Framework

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



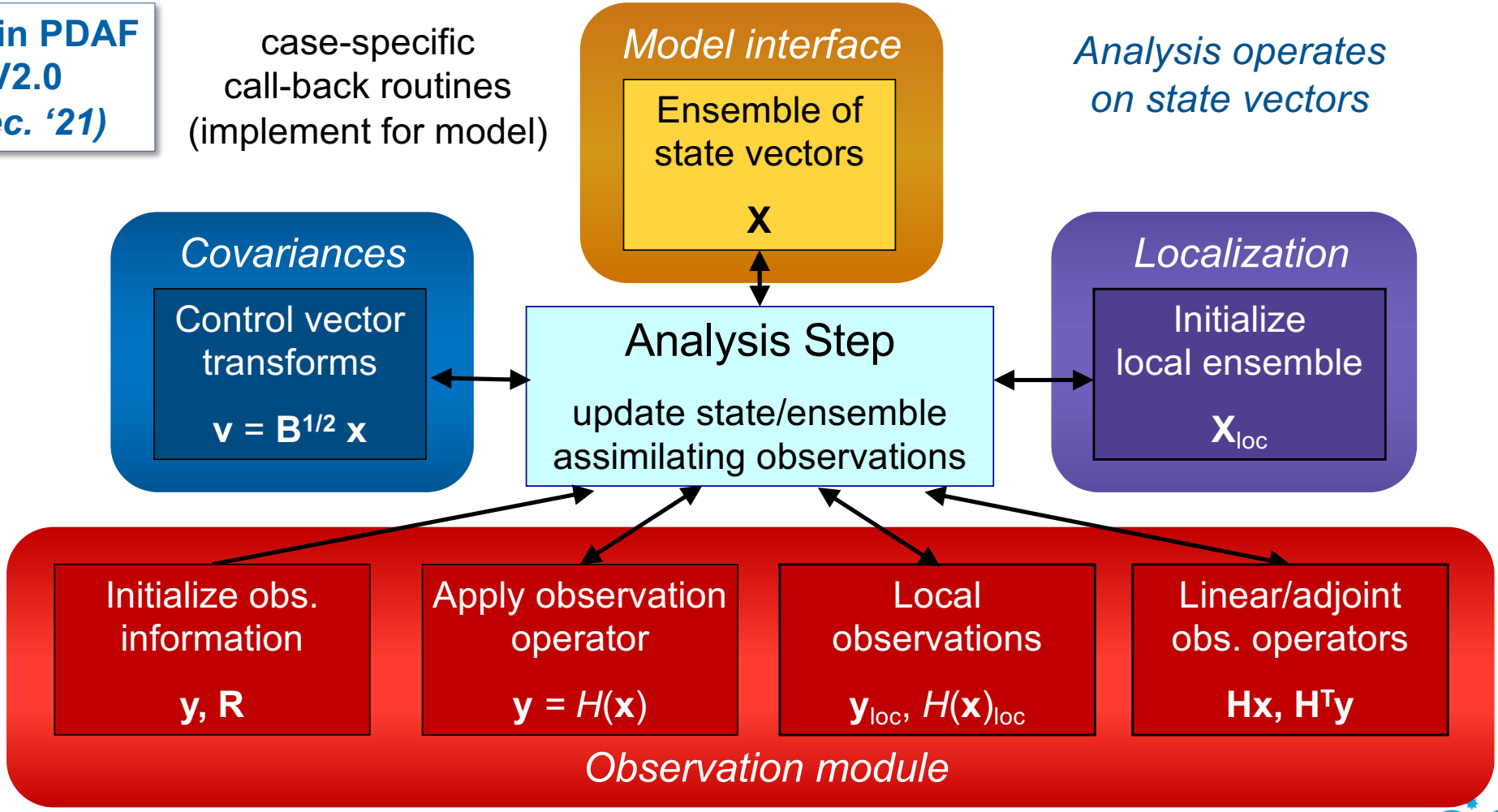
Implementing the 3D (Ensemble) Variational Analysis Step

PDAF Parallel Data Assimilation Framework

New in PDAF V2.0 *(Dec. '21)*

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

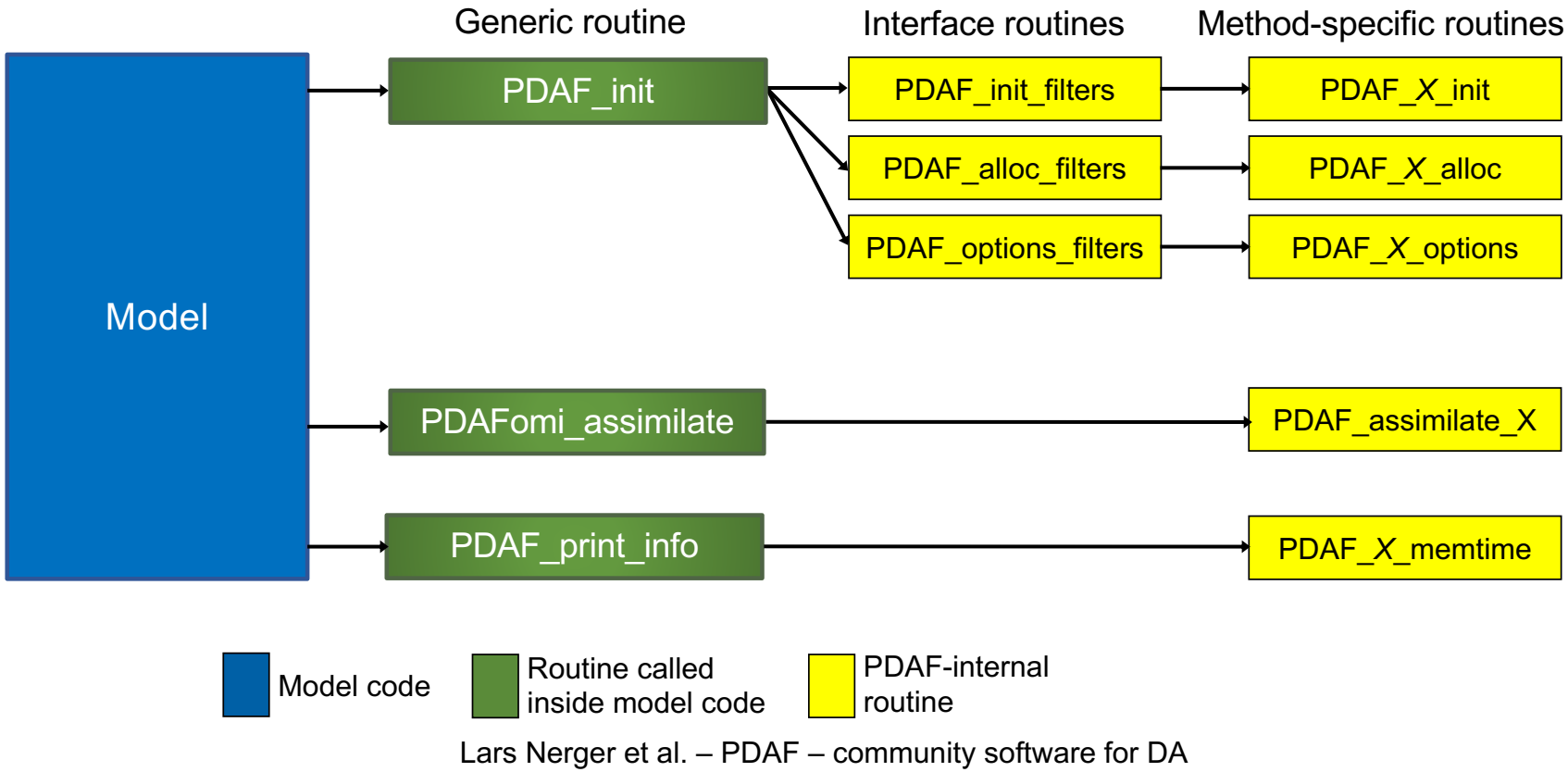
Analysis operates on state vectors



Internal interface of PDAF

PDAF Parallel Data Assimilation Framework

- PDAF has a framework structure for ensemble forecasts
- Internal interface to connect filter algorithms
- Easy addition of new DA methods by adding



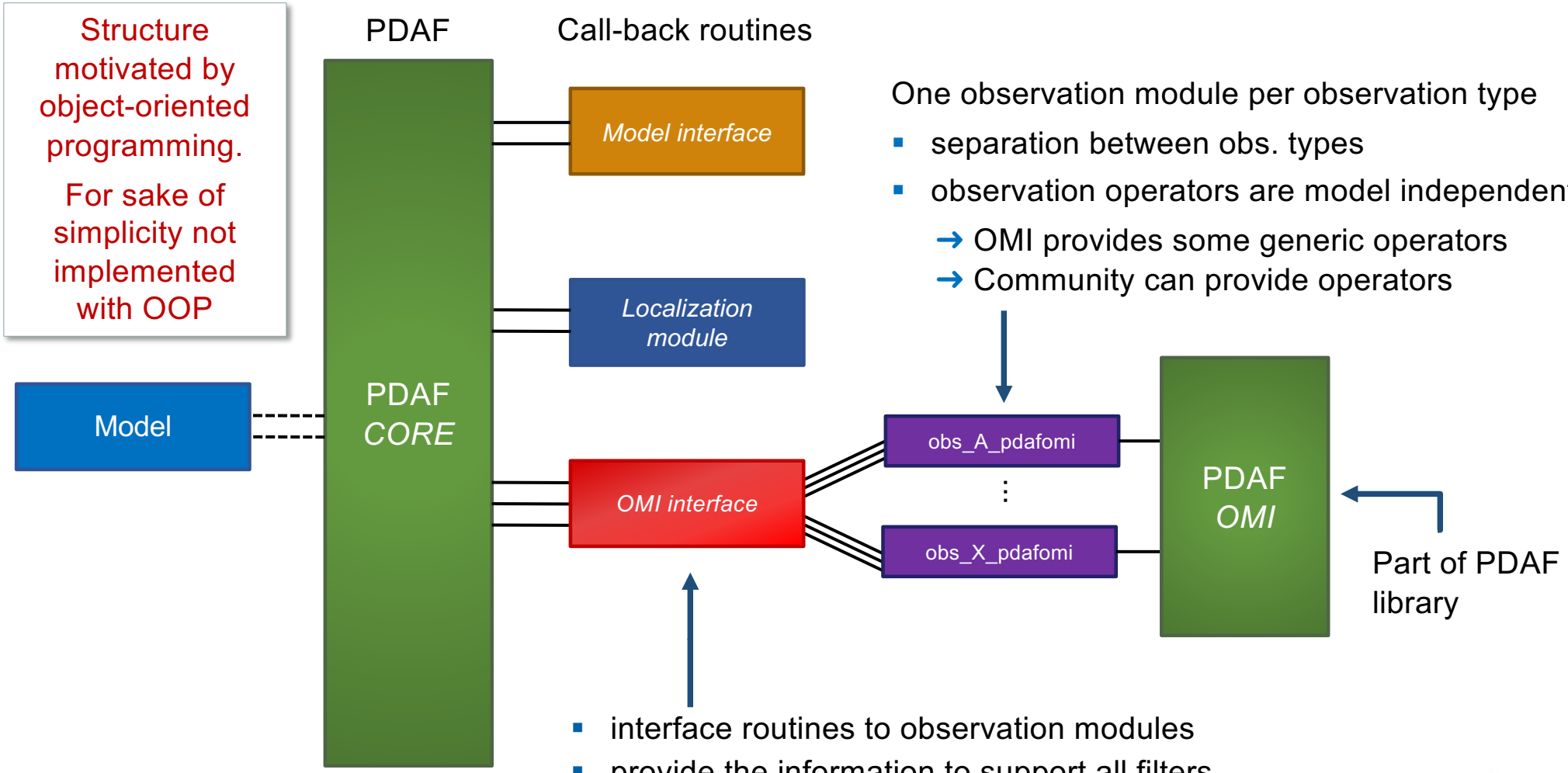
Recent and current developments

OMI: Code structure *(Observation Module Infrastructure)*

PDAF Parallel Data Assimilation Framework

Structure motivated by object-oriented programming.

For sake of simplicity not implemented with OOP



One observation module per observation type

- separation between obs. types
- observation operators are model independent
  - → OMI provides some generic operators
  - → Community can provide operators

- interface routines to observation modules
- provide the information to support all filters

Support for Strongly Coupled DA

PDAF Parallel Data Assimilation Framework

Strongly coupled DA:
Assimilate observation of component A into component B

achieved in PDAF by adapting MPI communicator for the filter:

→ joint state vector decomposed over the filter processes

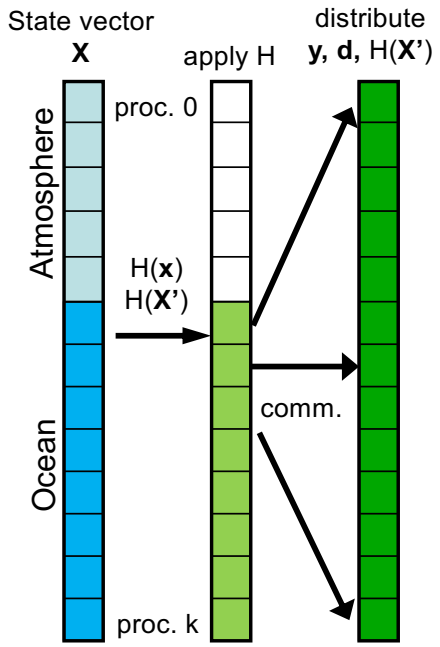
→ Provide observation operator that only performs MPI communication

(independent of model coupler)

need innovation **d** = H(**x**) – **y**
and observed ensemble perturbations H(**X'**)

Observation operator H links different compartments

1. Compute part of **d** and H(**X'**) on process 'owning' the observation
2. Communicate **d** and H(**X'**) to processes for which observation is within localization radius



Observation handling in strongly coupled DA

Nerger, Tang, Mu (2020). GMD, 13, 4305–4321, doi:10.5194/gmd-13-4305-2020

AWI

PDAF code: DA Algorithms and models

PDAF Parallel Data Assimilation Framework

PDAF originated from comparison studies of different filters

Ensemble Filters and smoothers - *global and localized*

- EnKF (Evensen, 1994 + perturbed obs.)
- (L)ETKF (Bishop et al., 2001/Hunt et al. 2007)
- ESTKF (Nerger et al., 2012)
- NETF (Toedter & Ahrens, 2015)
- Particle filter
- *EnOI mode*

3D-Var schemes
(incremental with control variable transformation)

- 3D-Var with parameterized covar.
- 3D Ensemble Var
- Hybrid 3D-Var

Model bindings

- MITgcm
- AWI-CM / FESOM

Community provided:

SCHISM/ESMF

TerrSysMP-PDAF

Upcoming:
- Hybrid NETF/LETKF

Upcoming:
- NEMO 4 (U Reading)
- GOTM/FABM (BB ApS)

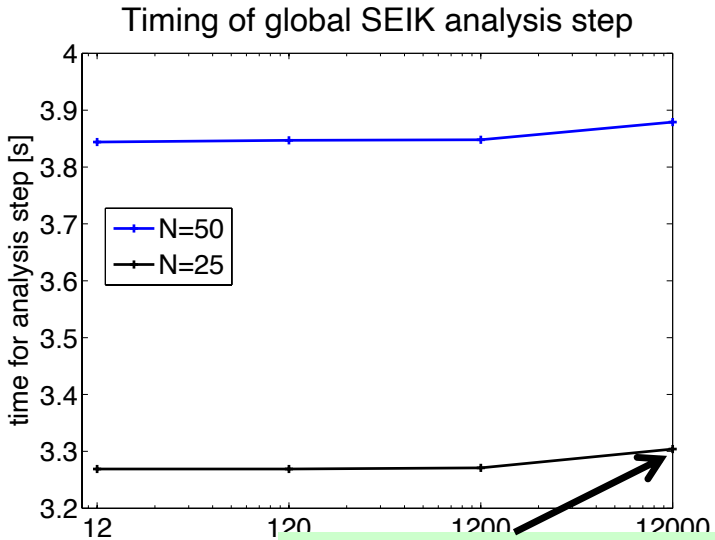
Toy models (full implementations with PDAF)

- Lorenz-96 / Lorenz-63
- Lorenz-2005 models II and III

PDAF Capability: Very big test case

PDAF Parallel Data Assimilation Framework

- 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
- Very small increase in analysis time (~1%) (Ideal would be constant time)



State dimension: 1.2e11

Observation dimension: 2.4e9

- Didn’t try to run a real ensemble of largest state size (no model yet)
- Latest test: analysis step using 57600 processor cores; state dimension 8.6e11, number of observations 1.7e10

AWI

Scalability (Climate model AWI-CM; DA into ocean)

PDAF Parallel Data Assimilation Framework

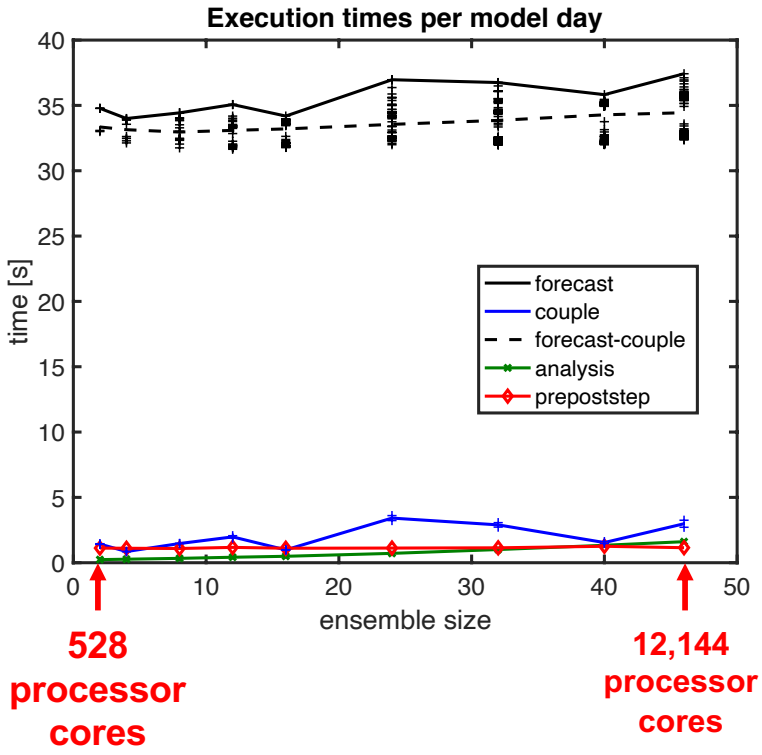
Daily assimilation of sea surface temperature

- MPI-tasks (each model instance): Atmosphere (ECHAM): 72 / Ocean (FESOM): 192
- Vary ensemble size

- Increasing forecast time with growing ensemble size (11%; more parallel communication; worse placement)
- some variability in forecast time over ensemble tasks (process placement, network)

Important factors for good performance

- Need optimal distribution of programs over compute nodes/racks (here set up as ocean/atmosphere pairs)
- Avoid conflicts in IO (Best performance when each AWI-CM task runs in separate directory)



Nerger et al., GMD (2020), doi:10.5194/gmd-13-4305-2020

Requirements

- Fortran compiler
- MPI library
- BLAS & LAPACK
- make

- PDAF is at least tested (often used) on various computers:
  - Notebook & Workstation: MacOS, Linux (gfortran)
  - Cray XC30/40 & CS400 (Cray ftn and ifort)
  - NEC SX-Aurora TSUBASA vector computer
  - ATOS Bull Sequana X (ifort)
  - HPE Cray Apollo (Fujitsu A64FX ARM processor)

Summary - PDAF: A tool for data assimilation

PDAF Parallel Data Assimilation Framework

- a program library for ensemble modeling and data assimilation
- provides support for ensemble forecasts, DA diagnostics, and fully-implemented filter and smoother algorithms
- makes excellent use of supercomputers
- *separation of concerns:* model, DA methods, observations
- easy to couple to models and to program case-specific routines
- easy to add new DA methods – good for research on algorithms
- efficient for research and operational use
- community code for DA methods and observations

PDAF adds DA to models

Couple model and PDAF within days

Get DA capability in a month

Extend to full multivar. system

Run DA in known environment

Access new DA methods by updating PDAF

Open source:
Code, documentation, and tutorial available at

http://pdaf.awi.de

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

AWI