Alfred Wegener Institute
for Polar and Marine Research

Scalable sequential data assimilation with the Parallel Data Assimilation Framework PDAF

Lars Nerger, Wolfgang Hiller, and Jens Schröter

Alfred Wegener Institute for Polar and Marine Research, Bremerhaven, Germany
Contact: Lars.Nerger@awi.de · http://www.awi.de

Introduction

Data assimilation applications with large-scale numerical models exhibit extreme requirements on computational resources. Good scalability of the assimilation system is essential to make these applications feasible. Sequential data assimilation methods based on ensemble forecasts, like ensemble-based Kalman filters, provide such good scalability, because the forecast of each ensemble member can be performed independently. This parallelism has to be combined with the parallelization of both the numerical model and the data assimilation algorithm.

The Parallel Data Assimilation Framework PDAF has been developed to simplify the implementation of scalable data assimilation systems based on existing numerical models. PDAF provides an environment for implementing a data assimilation system with parallel ensemble forecasts and parallel numerical models. Further, it includes several optimized parallel filter algorithms.

PDAF is currently used in several research projects. Also, it is in pre-operational use at the German Maritime and Hydrographic Agency (BSH).

Sequential Data Assimilation

PDAF is configured for sequential data assimilation (see **Fig. 1**). During the last years, a variety of filter and smoother algorithms have been developed. A selection of important filters is fully implemented and optimized in PDAF including parallelization. Available are common algorithms like

- EnKF – Ensemble Kalman Filter [1]
- LETKF – Local Ensemble Transform Kalman Filter [2]
- (L)SEIK – (Local) Singular Evolutive Interpolated Kalman filter [3, 4]
- SEEK – Singular Evolutive Extended Kalman filter [5]

In addition to the filters, common fixes and tuning options like covariance inflation are implemented. In addition, a selection of advanced localization options are available including observation localization.

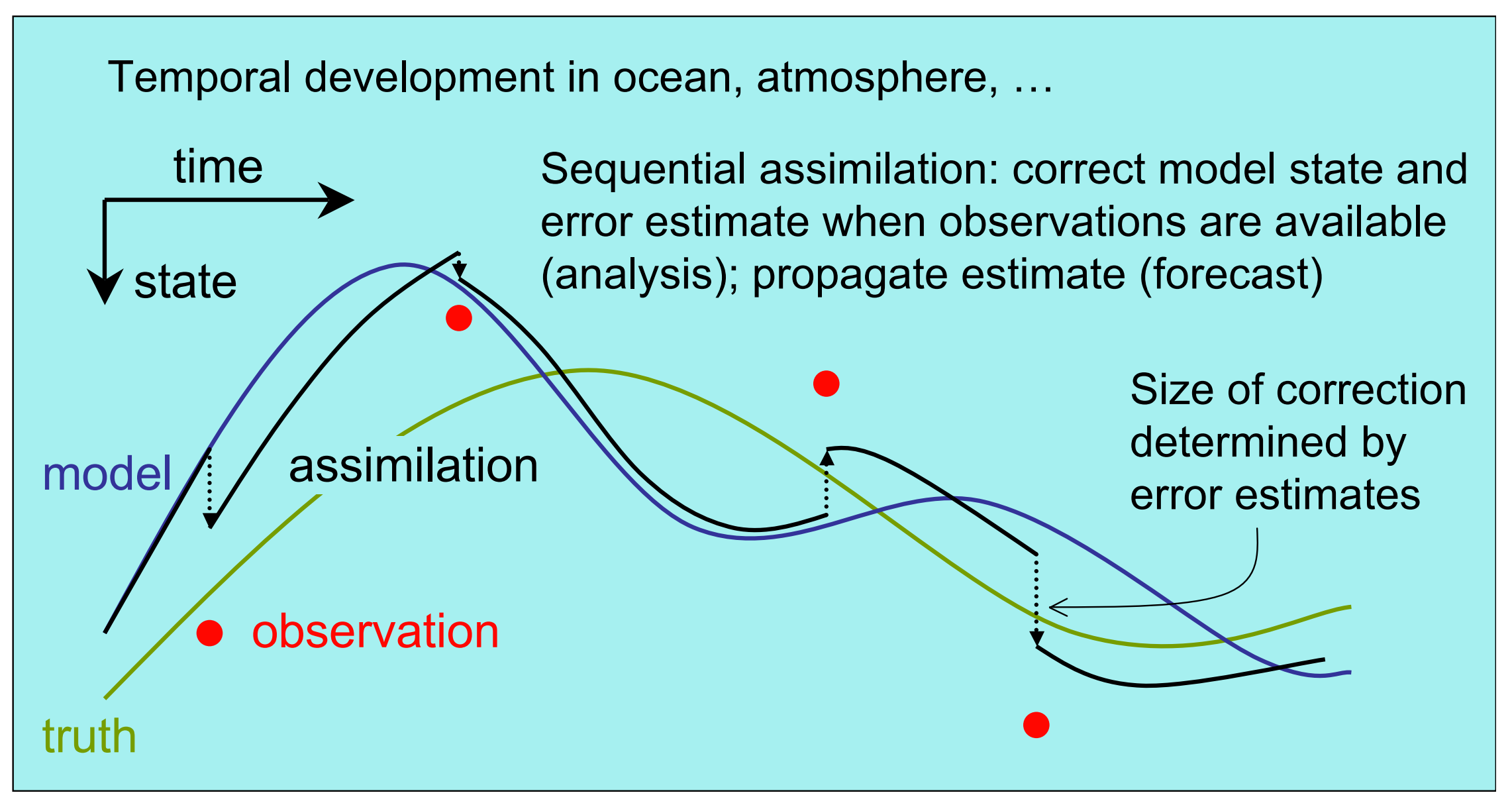

Temporal development in ocean, atmosphere, …
time
state
Sequential assimilation: correct model state and error estimate when observations are available (analysis); propagate estimate (forecast)
Size of correction determined by error estimates
model
assimilation
observation
truth

Fig. 1: Principle of sequential data assimilation with a filter algorithm. The state estimate of the assimilation is given by the ensemble mean. The analysis estimate lies typically between the forecast estimate and the observation, hence closer to the true state.

PDAF's Implementation Concept

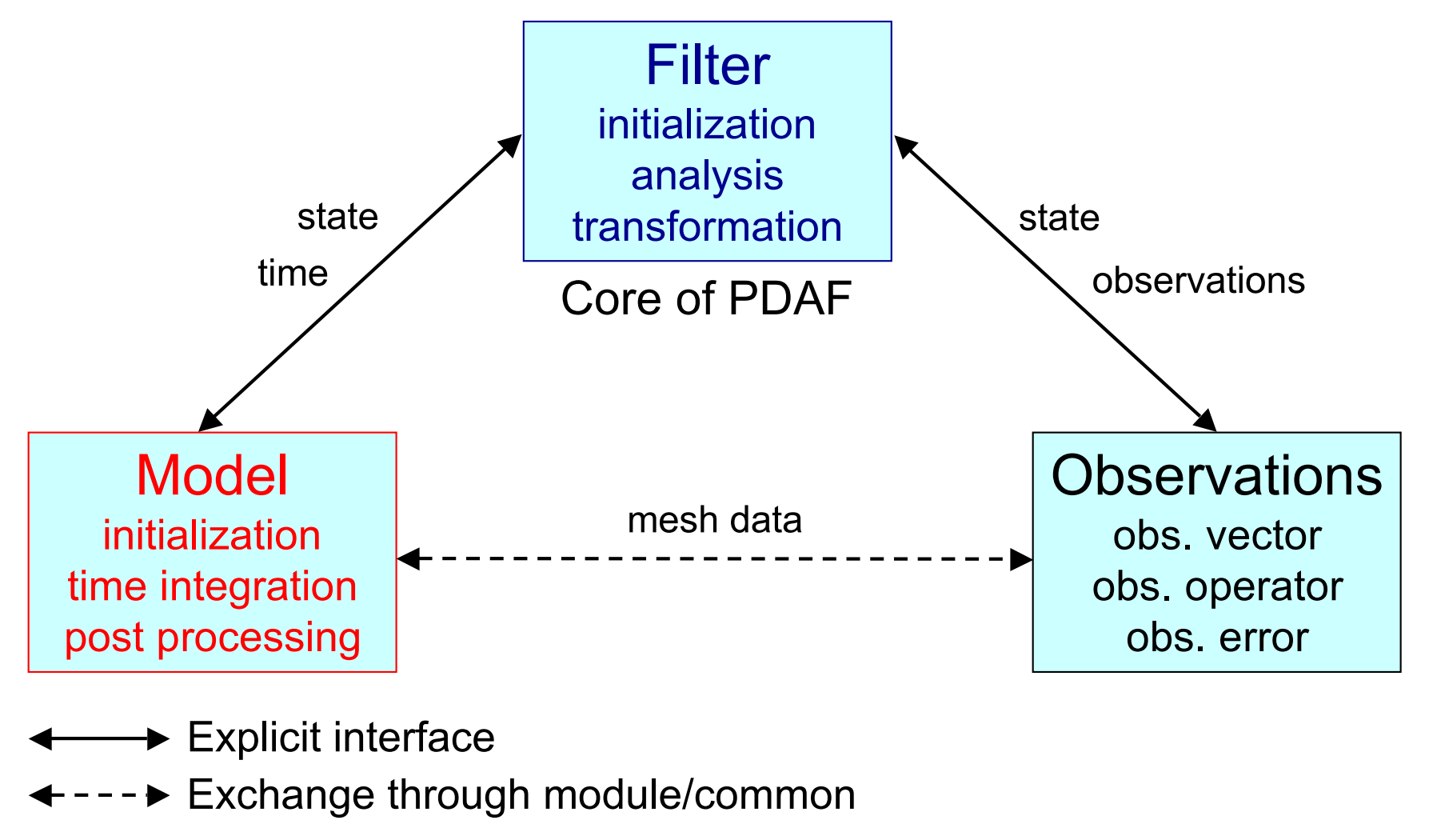


Fig. 2: Logical separation of the data assimilation system and interfacing between the components.

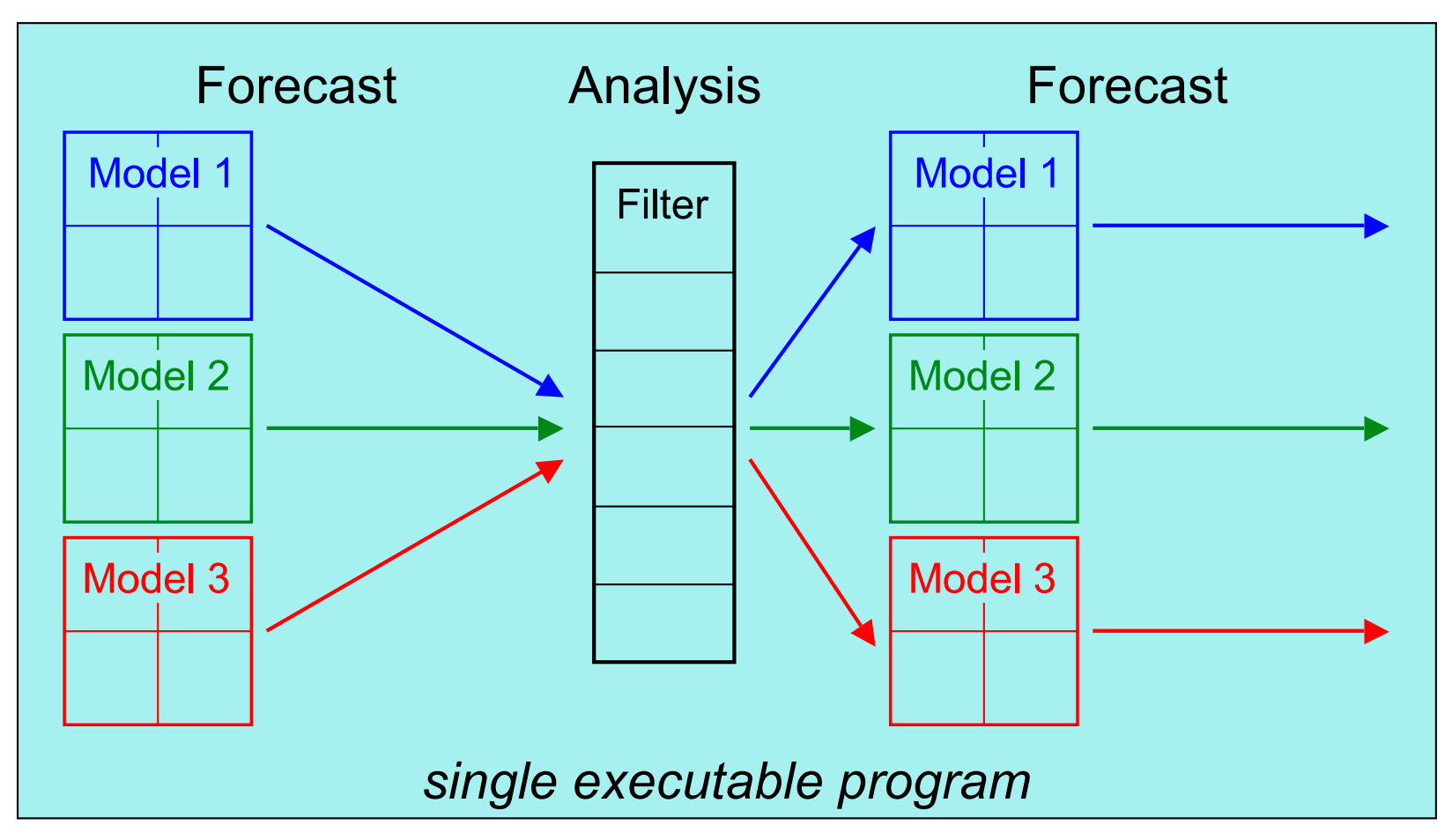


Fig. 3: 2-level parallelization of the data assimilation system: 1. Each model task can be parallelized. 2. Several model tasks are executed concurrently. In addition, the filter analysis step uses parallelization. In the online-mode of PDAF, all components are included in a single program.

PDAF is based on a consistent separation of the components of the data assimilation system: model, filter algorithm, and observations (see **Fig. 2**). The filter algorithms are part of PDAF's core routines, while the model routines and routines to handle observations are provided by the user. PDAF uses a standard interface for all filter algorithms to connect the three components. All user-supplied routines exist in the context of the model and can be implemented like model routines.

Next to providing fully implemented and parallelized ensemble filter algorithms, PDAF provides support for a 2-level parallelization for the assimilation system to perform the ensemble integrations in parallel using a single executable (see **Fig. 3**).

Figure 4 shows the extensions of the model code, when the assimilation system is implemented with PDAF in the online mode. Four calls to subroutines have to be added. In addition, an external loop enclosing the time stepping part of the model is required to perform ensemble integrations. With this concept, the implementation of the data assimilation system amounts to an extension of the model. In contrast to other frameworks, the model does not need to exist as a separate subroutine.

Further information and the source code of PDAF is available on the web site:

http://pdaf.awi.de

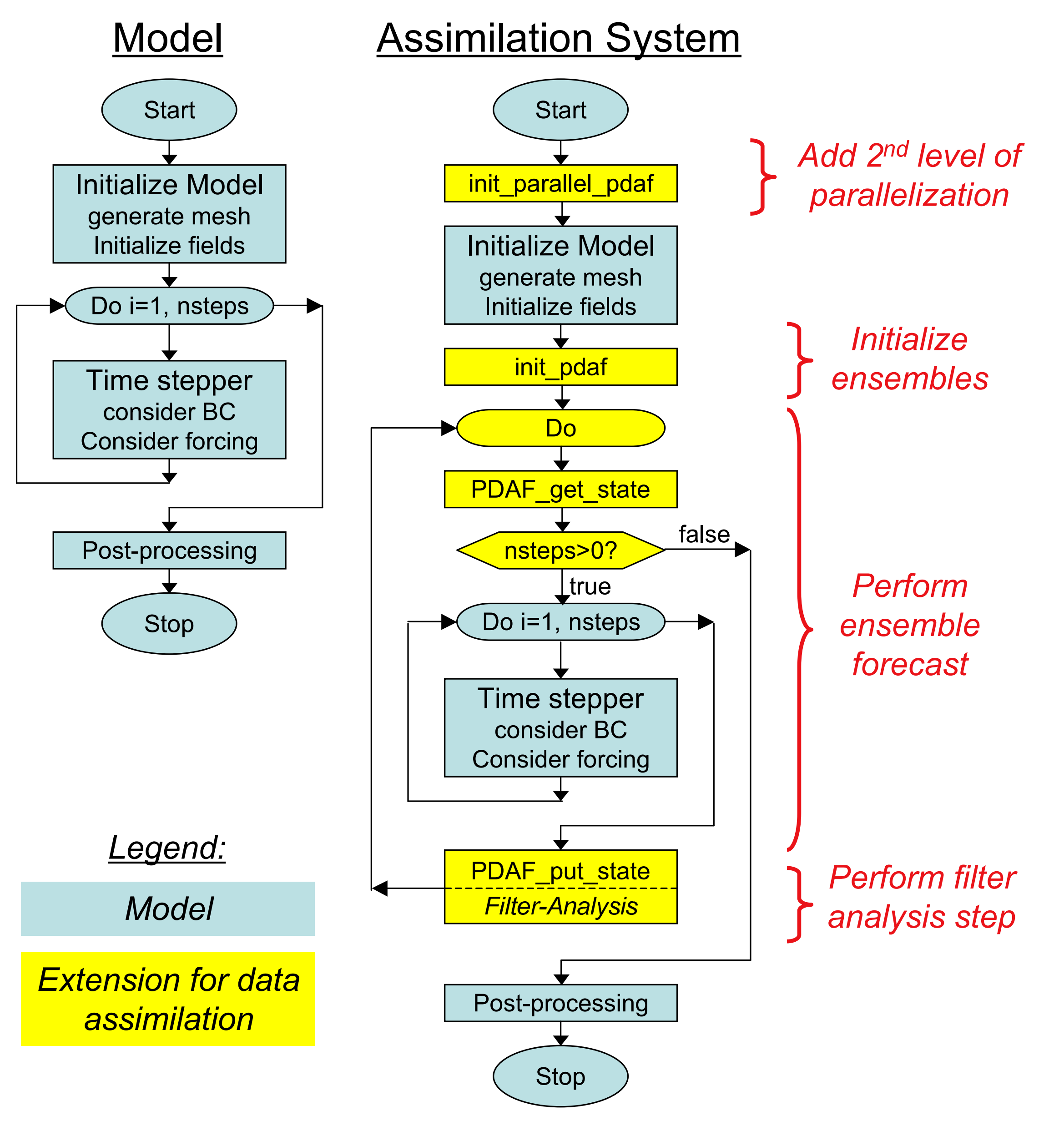


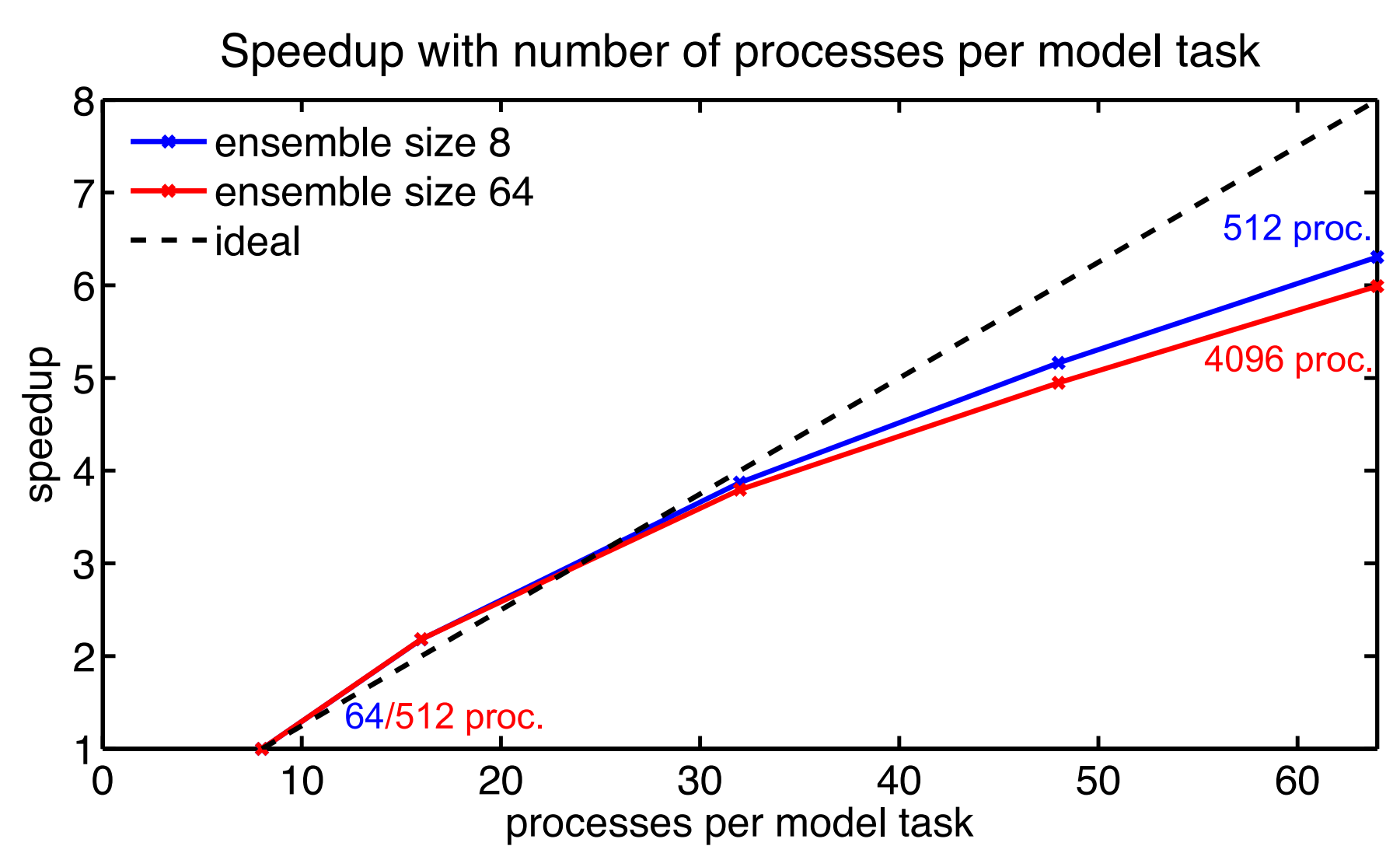
Fig. 4: Extension of a model source code to implement a data assimilation system using PDAF [6]. The forecast phase is controlled by user-supplied routines that are called by PDAF_get_state. Implementations following this strategy have been performed for different models like FEOM, BSHcmod, MIPOM, NOBM, and ADCIRC.

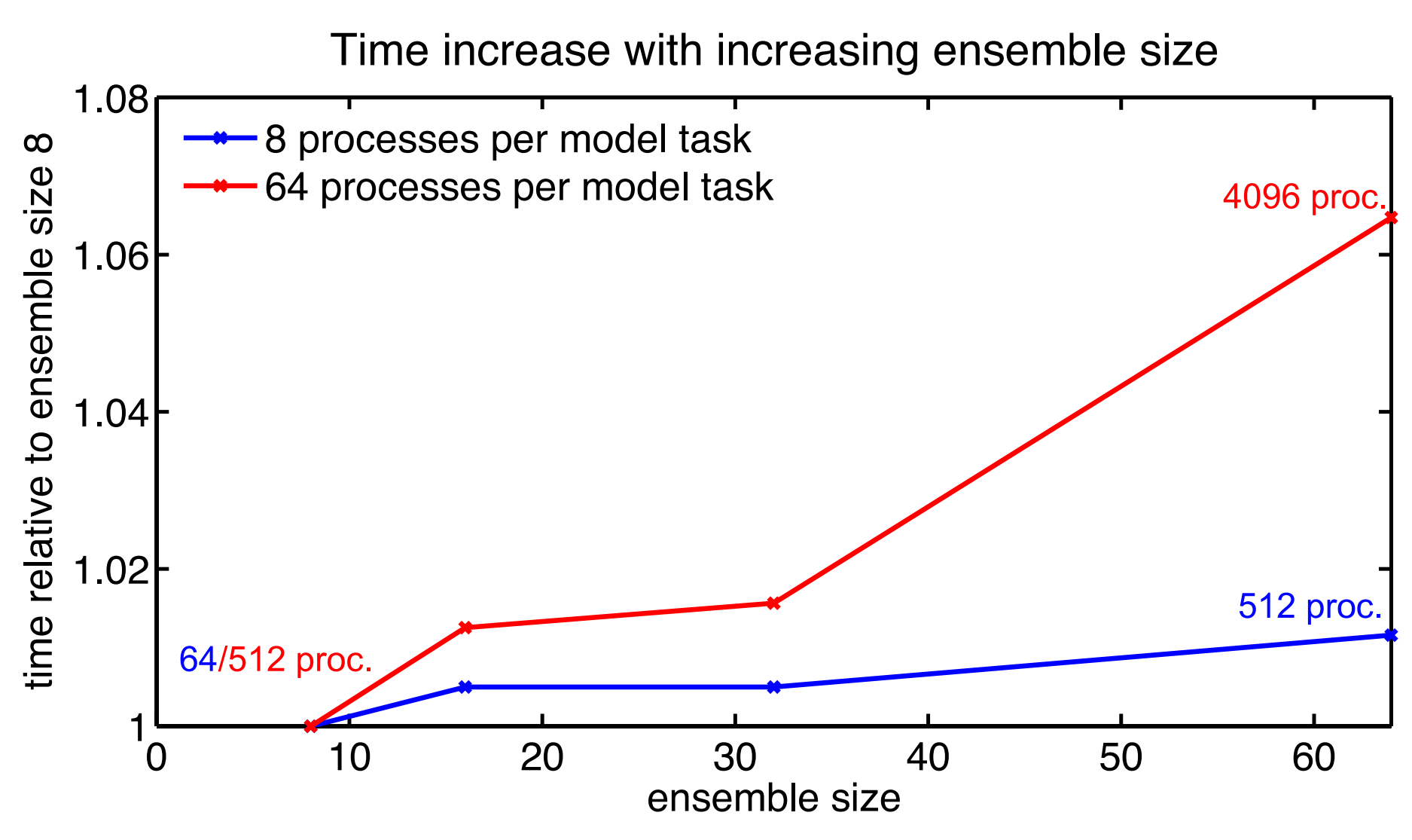
Parallel Performance

The parallel performance has been tested with an implementation of PDAF with the model finite-element model FEOM. About 94 to 99% of the computing time are used for the ensemble integrations.

Speedup is accessed with a constant ensemble size. Due to the parallel properties of the model, a speedup of 6 is obtained when the number of processors is increased by a factor of 8 (left panel).

The **scalability** of the assimilation system is visible when the number of processes per model task is kept constant. Increasing the ensemble size by a factor of eight results in a time increase between only 1% and 7% (right panel).





Summary

- The Parallel Data Assimilation Framework (PDAF) has been developed to simplify the implementation of data assimilation systems. It can be used to test assimilation methods, but is also applicable for operational data assimilation systems.
- A very good scalability is provided through the complete parallelism of all parts of the assimilation system (ensemble integration, filter algorithms, and perhaps the model itself).
- Only minimal changes to the model source code are required when combining a model with PDAF in its online mode. An offline-mode is possible with separate programs for model and filtering. The offline mode avoids changes to the model code, but leads to a smaller computing performance.
- PDAF is distributed as free open-source software through the web site http://pdaf.awi.de.

References

[1] Evensen, G. (1994). Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *J. Geophys. Res.* 99C: 10143

[2] Hunt, B.R., E.J. Kostelich, and I. Szunyogh (2007). Efficient data assimilation for spatiotemporal chaos: A local ensemble transform Kalman filter. *Physica D* 230: 112–126

[3] Pham, D.T. (2005). Stochastic Methods for Sequential Data Assimilation in Strongly Nonlinar systems. *Mon. Wea. Rev.* 129: 1194–1207

[4] L. Nerger, S. Danilov, W. Hiller, and J. Schröter (2006). Using sea-level data to constrain a finite-element primitive-equation ocean model with a local SEIK filter. *Ocean Dynamics* 56: 634–649

[5] Pham, D.T., J. Verron, M.C. Roubaud (1998). A singular evolutive extended Kalman filter for data assimilation in oceanography *J. Mar. Syst.* 16: 323-340

[6] Nerger, L., W. Hiller, and J. Schröter (2005). PDAF - The Parallel Data Assimilation Framework: Experiences with Kalman Filtering, in *Use of High Performance Computing in Meteorology - Proceedings of the 11th ECMWF Workshop* / Eds. W. Zwieflhofer, G. Mozdzynski. World Scientific, pp. 63–83