Ensemble-based data assimilation for complex models of the Earth system

Lars Nerger

Alfred Wegener Institute, Bremerhaven, Germany

Thanks to

Yuchen Sun, Sophie Vliegen, Anju Sathyanarayanan, Qi Tang

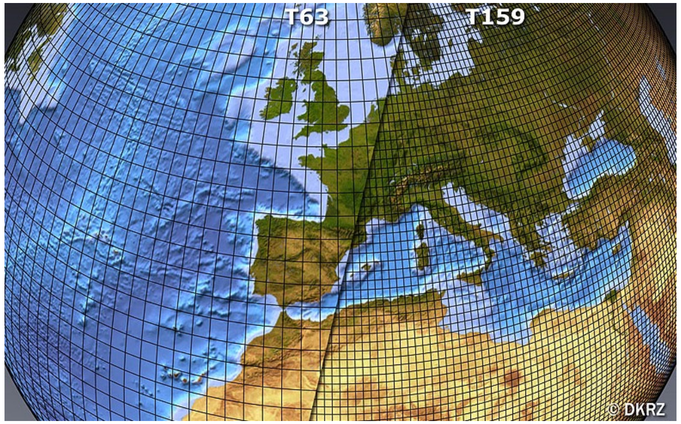
Invited Colloquium at National University of Singapore, September 25, 2025

Overview

- High-dimensional data assimilation applications
- Software
- Methods
- Open points

Data Assimilation – Combining Models and Observations

Models



Observations

Combine both sources of information

quantitatively and optimally by computer algorithm

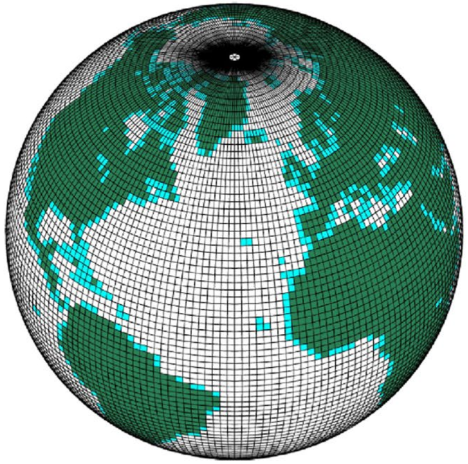
→ Data Assimilation

High-dimensional
Data Assimilation Applications

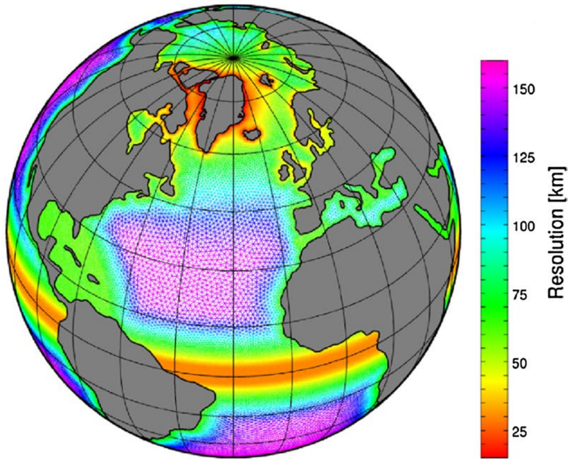
improving model predictions

Assimilation into atmosphere-ocean coupled model: AWI-CM

Atmosphere



Ocean



OASIS3-MCT

fluxes

ocean/ice state

Atmosphere
+ land surface
96 processor cores

Coupler library
OASIS3-MCT

Ocean
+ sea ice
192 processor cores

Two separate programs for atmosphere and ocean

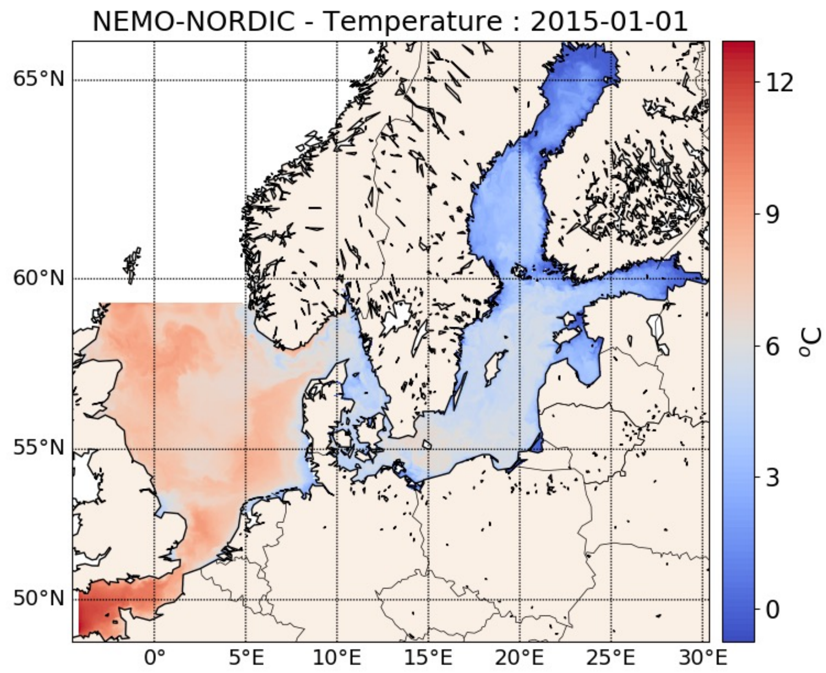
Goal: Develop data assimilation methodology for
cross-component assimilation ("strongly-coupled")
"assimilate ocean observations into the atmosphere"
for improved model simulations

Nerger et al. (2020). Geoscientific Model Development, 13, 4305–4321, doi:10.5194/gmd-13-4305-2020

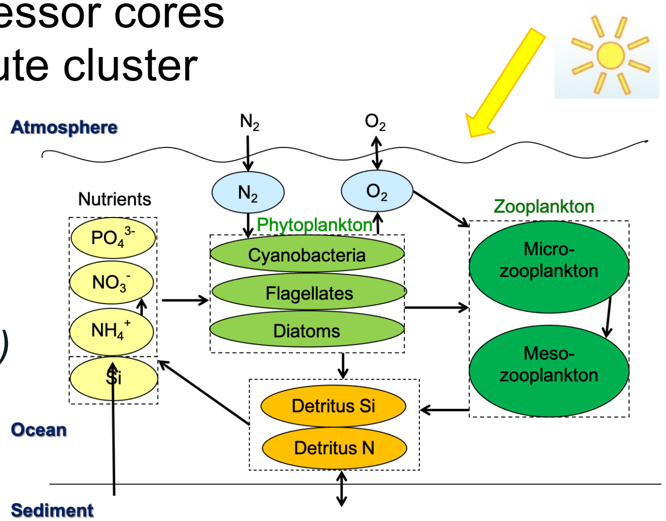
Regional ocean forecasting

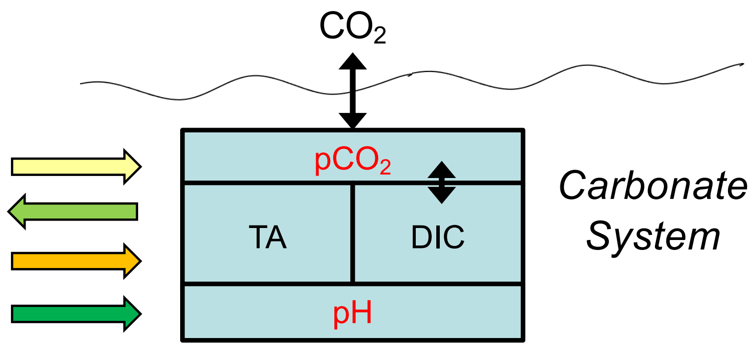
Operational configuration of Copernicus Marine Forecasting Center for Baltic Sea (BAL-MFC)

- Model setup
  - Ocean model NEMO
  - Coupled to ecosystem/carbon model ERGOM
  - 1.8 km resolution, 56 layers
  - Time step 90 sec
  - 192 processor cores on compute cluster



Ecosystem (biogeochemistry)



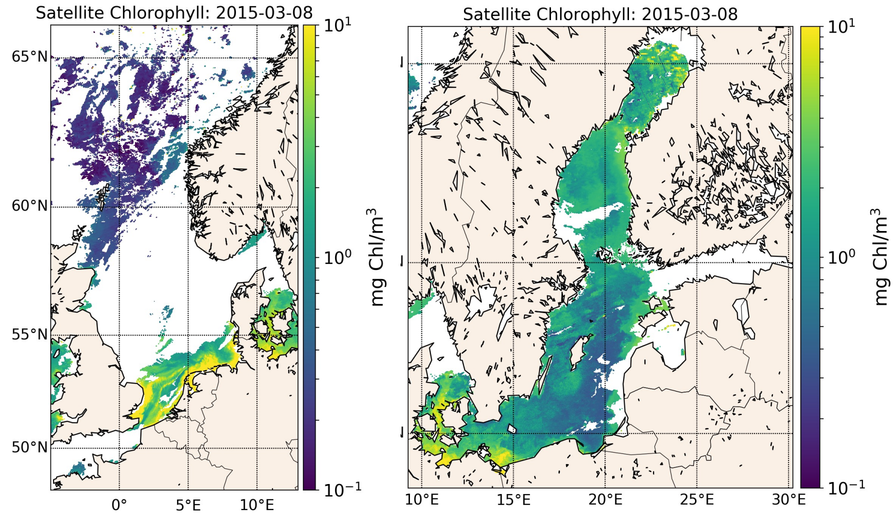


Carbonate System

Observations - Ocean

Remote sensing (satellites, radar, planes)

- Only observe ocean surface
- E.g. temperature, sea surface height, chlorophyll

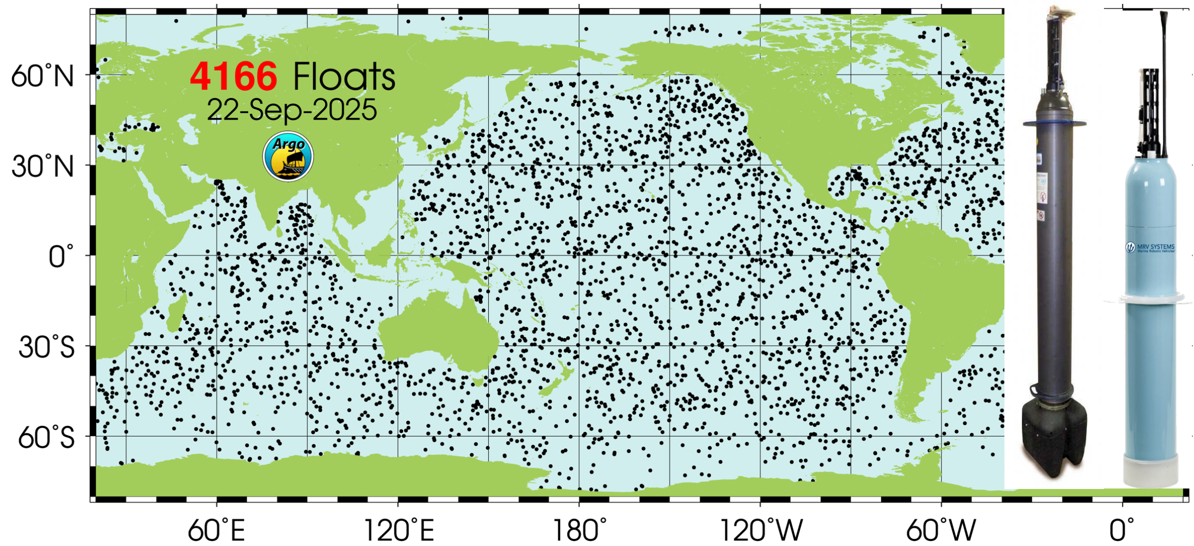


Ships / stations / drifters / floats ('in situ')

- Measurements inside the ocean
- Very sparse data

Example 'ARGO floats'

- Observations of temperature, salinity
- down to 2000m
- Velocities from displacement
- World ocean covered by 4166 devices - very sparse



Applying data assimilation

Integrate observations and models

- Models provide dynamical information
- Observations provide sparse data on real world
- Need uncertainty estimates for model state and observations
  - variances and error-correlations

→ Apply ensemble-based data assimilation
  - Using O(10) – O(100) model states (more not feasible)
    - Costly to compute (O(10^3) – O(10^3) processor cores
    - Large amount of output data (tera-bytes)

  - Small sample of probability distribution of model state
    - Large sampling errors

Data Assimilation - Possibilities

"Observation-constrained (ensemble) modeling"

Aims

1. Optimal estimation of some modelled system:
   - initial conditions (forecasting - weather/ocean/etc)
   - state trajectory (reanalysis - temperature, concentrations, …)
   - process parameters (ice strength, plankton growth, …)
   - fluxes (heat, primary production, …)
   - boundary conditions and 'forcing' (wind stress, …)

2. More advanced: Improvement of model formulation and observations
   - detect systematic errors (bias)
   - revise parameterizations based on parameter estimates
   - detect relevant observations for observation system design

Software

How we realize DA applications

Computing challenges and features

High-dimensional models

- Costly to compute
- Large amount of output data
- Large size of state vectors
  - Containing all relevant model fields
  - Usually distributed due to parallelization

Ensemble-based data assimilation

- Multiply computing cost (parallel or sequential)
- Full ensemble output would multiply amount of output data
  - Usually only write ensemble mean and variance
- Computing time of model dominates over assimilation method
  - Assimilation often only 1-5% of total run time

PDAF – Parallel Data Assimilation Framework

PDAF Parallel Data Assimilation Framework

A unified tool for interdisciplinary data assimilation …

- provide support for parallel ensemble forecasts
- provide DA methods (EnKFs, smoothers, PFs, 3D-Var) - fully-implemented & parallelized
- provide tools for observation handling and for diagnostics
- easy implementation with (probably) any numerical model (<1 month)
- a program library (PDAF-core) plus additional functions & templates
- run from notebooks to supercomputers (Fortran, MPI & OpenMP – model compatibility)
- ensure separation of concerns (model – DA method – observations – covariances)
- first release in year 2004; continuous further development

Focus on

- Easy implementation
- Performance for complex models
- Flexibility to extend system

Open source:
Code, documentation, and tutorial available at

https://pdaf.awi.de

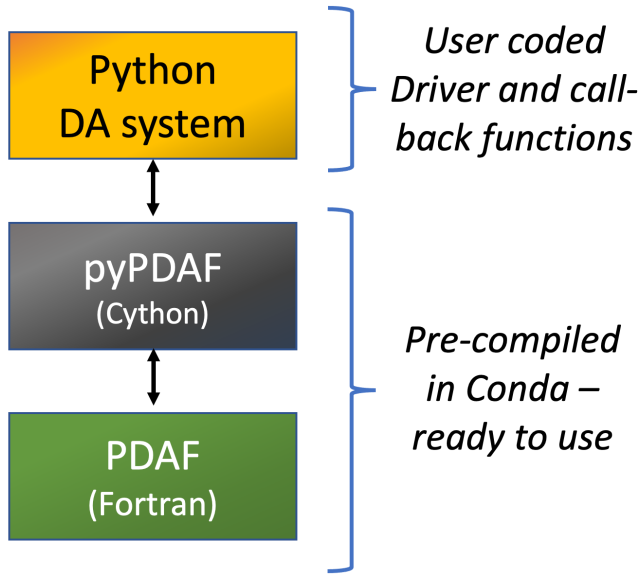
github.com/PDAF/PDAF

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

pyPDAF

Python interface to PDAF

- allows to code all application-specific functionality in Python
  (not touching Fortran!)
- assimilation analysis computed inside PDAF
  (excellent performance due to compiled Fortran code)
- supports all functionality of PDAF including MPI-parallelization
  - online coupling (e.g. for Python-coded models)
  - offline coupling (using files from model runs)
- demonstrated low overhead for localized ensemble filters
- installation using Conda



github.com/yumench/pyPDAF

Y. Chen et al. (2024). Geoscientific Model Development, submitted

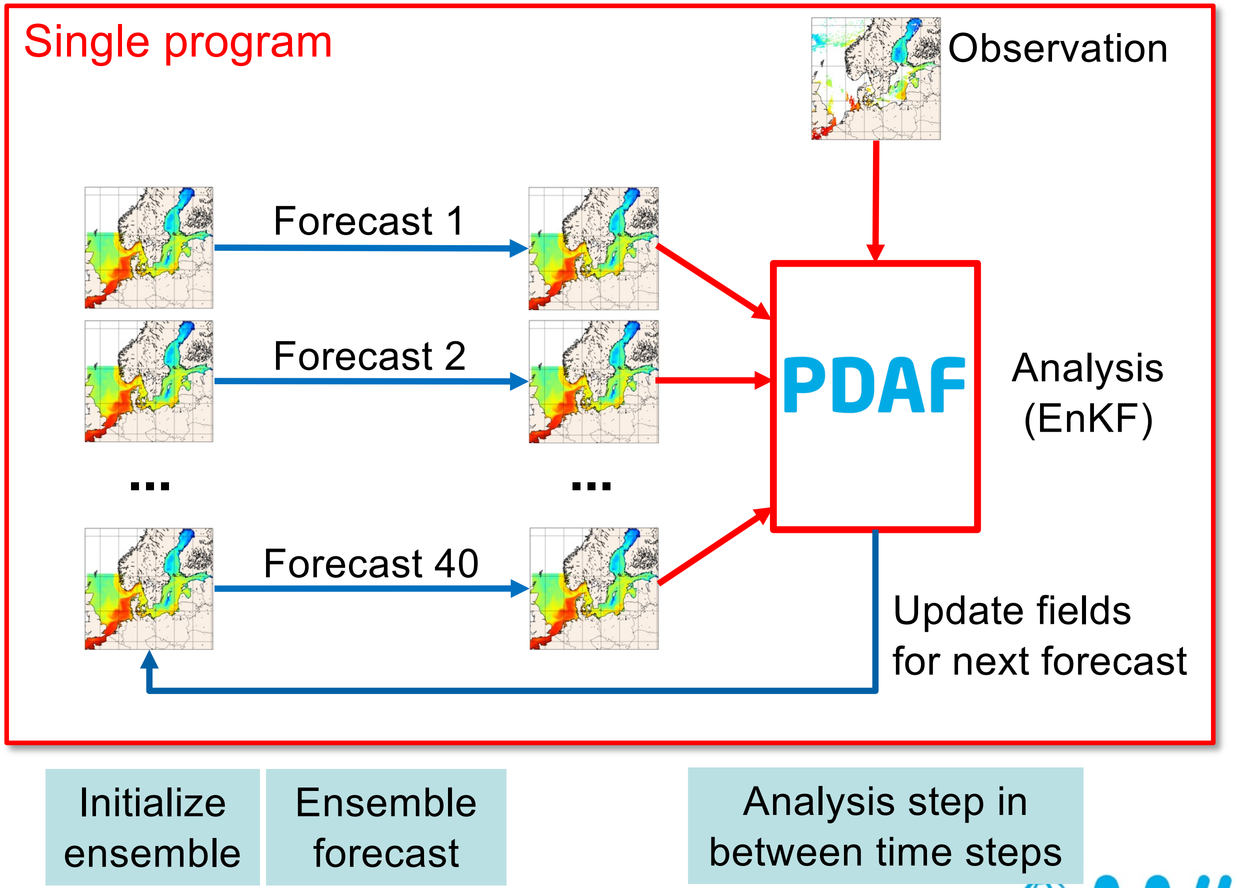
Online-Coupling – Assimilation-enabled Model

PDAF Parallel Data Assimilation Framework

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

- EnOI and 3D-Var also possible:
  - Evolve single model state
  - Prescribe ensemble perturbations or covariance



Methods

Ensemble-based data assimilation

Estimation by joining model and observational data

Data Assimilation – Model and Observations

Two components:

1. **State**: $\mathbf{x} \in \mathbb{R}^n$ (contains different model variables)

   Dynamical model

   $$\mathbf{x}_i = M_{i-1,i}\left[\mathbf{x}_{i-1}\right]$$

2. **Observations**: $\mathbf{y} \in \mathbb{R}^m$ (contains different observed fields)

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

   $$\mathbf{y} = H\left[\mathbf{x}\right]$$

Dimensions:

state	n: $10^6 - 10^9$
observations	m: $10^4 - 10^6$

Linear and Nonlinear Ensemble Filters

- Represent state and its error by ensemble $\mathbf{X}$ of N_e states
  (use ensemble perturbation matrix $\mathbf{X}' = \mathbf{X} - \bar{\mathbf{X}}$)

- **Forecast**:
  - Integrate ensemble size N_e with numerical model

Dimension of correction (error) space : $N_e - 1$

- **Analysis step:**
  - update ensemble mean $\overline{\mathbf{x}}^a = \overline{\mathbf{x}}^f + \mathbf{X}'^f \tilde{\mathbf{w}}$
  - update ensemble perturbations $\mathbf{X}'^a = \mathbf{X}'^f \mathbf{W}$

  (both can be combined in a single step)

- Ensemble Kalman & nonlinear filters: Different definitions of
  - weight vector $\tilde{\mathbf{w}}$ (dimension N_e)
  - Transform matrix $\mathbf{W}$ (dimension $N_e \times N_e$)

ETKF (Bishop et al., 2001) / ESTKF (Nerger et al., 2012)

- Ensemble Transform Kalman filter / Error Subspace Transform Kalman filter
  - Assume Gaussian distributions
  - Transform matrix ($\mathbf{L}_{ETKF} = \mathbf{X}'^f$ or $\mathbf{L}_{ESTKF} = \mathbf{X}^f\mathbf{T}$)

$$\mathbf{A}^{-1} = (N_e - 1)I + (\mathbf{HL})^T\mathbf{R}^{-1}\mathbf{HL}$$

  - Mean update weight vector

$$\tilde{\mathbf{w}} = \mathbf{A}(\mathbf{HL})^T\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{H}\overline{\mathbf{x}^f}\right)$$

(depends linearly on observation vector $\mathbf{y}$)

  - Transformation of ensemble perturbations

$$\mathbf{W} = \sqrt{N_e - 1}\mathbf{A}^{1/2}\mathbf{\Lambda}$$

$\mathbf{\Lambda}$: mean-preserving random matrix or identity

Note: $\mathbf{W}$ depends only on $\mathbf{R}$, not observation $\mathbf{y}$

Algorithms designed for maximum computational efficiency

Excellent parallelization possibility when combined with localization

Linear filter:
- Gaussian distributions assumed
- Linear in effect of $\mathbf{y}$

Covariance inflation and localization

- Inflation
  - sampling errors in ensemble result in too low variance
  - counter by increasing ensemble variance before analysis
  - Efficient utilizing factor ρ in transform matrix

- Localized Ensemble Transform Kalman filter
  - Loop through model grid and update 'local domains'
  - Utilize only observations within 'influence radius' around updated grid point
  - Local transform matrix with inflation

$$\hat{\mathbf{A}}^{-1} = \rho\,(N_e - 1)I + (\hat{\mathbf{H}}\mathbf{L})^T\hat{\mathbf{R}}^{-1}\hat{\mathbf{H}}\mathbf{L}$$

  - update weight vector and matrix

$$\hat{\bar{\mathbf{w}}} = \hat{\mathbf{A}}(\hat{\mathbf{H}}\mathbf{L})^T\hat{\mathbf{R}}^{-1}\left(\hat{\mathbf{y}} - \hat{\mathbf{H}}\overline{\mathbf{x}^f}\right)$$

$$\hat{\mathbf{W}} = \sqrt{N_e - 1}\hat{\mathbf{A}}^{1/2}\mathbf{\Lambda}$$

- Localization is empirical
- Optimal values for ρ and influence radius are unknown
- Some adaptive methods exist (most without theretical basis)

Improving beyond Ensemble Kalman

- Nonlinear filtering
  - Particle filters
    - replace ensemble transformation by weights and resampling
    - Difficult due to degenerate weights
    - Localization can help
  - Iterative filters
    - Iterative KFs (Bocquet et al.)
    - Particle flow filter (Stein variational descent) (Pulido, van Leeuwen, Hu)
  - Hybrid particle-Kalman filters
    - Combine linear (Gaussian) Kalman with particle filters
      - Localized PF (Poterjoy)
      - Local hybrid Kalman-nonlinear transform filter

NETF (Tödter & Ahrens, 2015)

- Nonlinear Ensemble Transform Filter
  - Mean update from Particle Filter weights:
    for Gaussian observation errors for all particles *i*

$$\tilde{w}^i \sim \exp\left(-0.5(\mathbf{y}-\mathbf{H}\mathbf{x}_i^f)^T\mathbf{R}^{-1}(\mathbf{y}-\mathbf{H}\mathbf{x}_i^f)\right)$$

(nonlinear function of observations $\mathbf{y}$)

  - Ensemble update
    - Transform ensemble to fulfill analysis covariance (like ETKF, but not assuming Gaussianity)
    - Derivation gives

$$\mathbf{W} = \sqrt{N}\left[\mathrm{diag}(\tilde{\mathbf{w}}) - \tilde{\mathbf{w}}\tilde{\mathbf{w}}^T\right]^{1/2}\Lambda$$

($\boldsymbol{\Lambda}$: mean-preserving random matrix; useful for stability)

Similar computational efficiency as ETKF/ESTKF

Localization analogous to ETKF/ESTKF

Excellent parallelization possibility when combined with localization

Nonlinear filter:
- No assumption of Gaussian distributions
- Nonlinear in $\mathbf{y}$

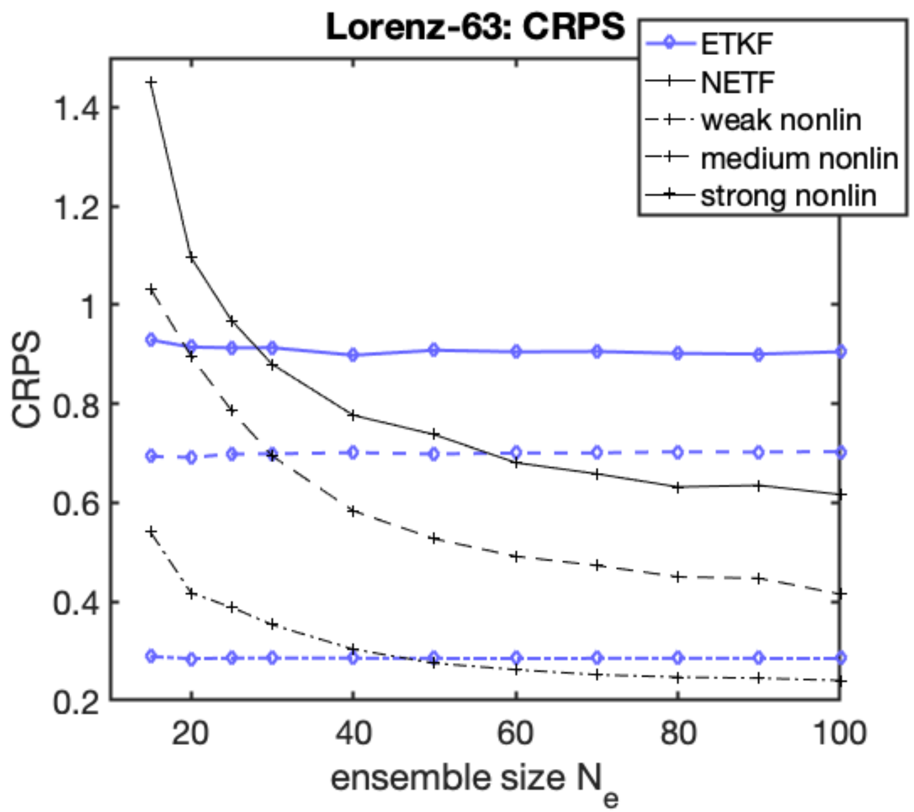
NETF is a second-order exact particle filter

Tödter, J. and Ahrens, B. (2015) *Mon. Wea. Rev.* **143**,1347–1367

EKTF & NETF with Lorenz-63 model

Dependence on ensemble size

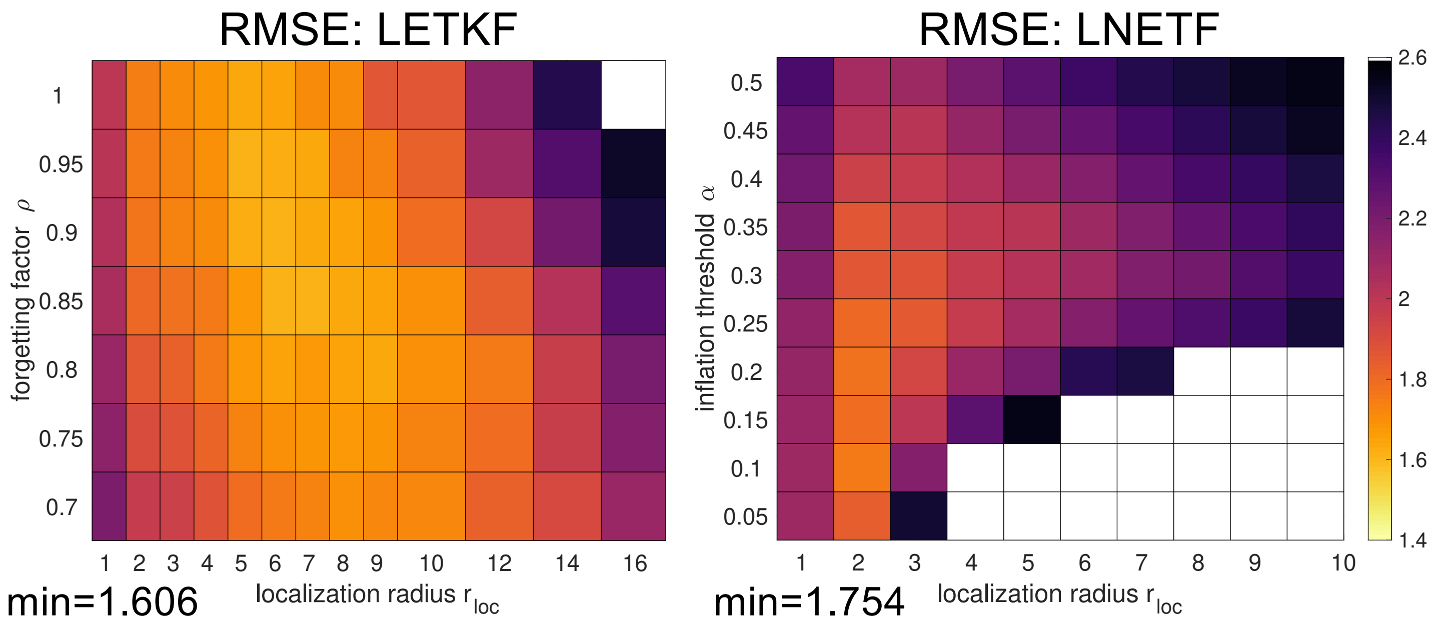
- NETF yields smaller errors than ETKF if ensemble size large enough
  - → Size limit decreases for larger nonlinearity
  - → Improvement by NETF stronger for higher nonlinearity



ETKF and NETF for
3 different nonlinearities

(weak Δt=0.1, medium Δt=0.4, strong Δt=0.7)

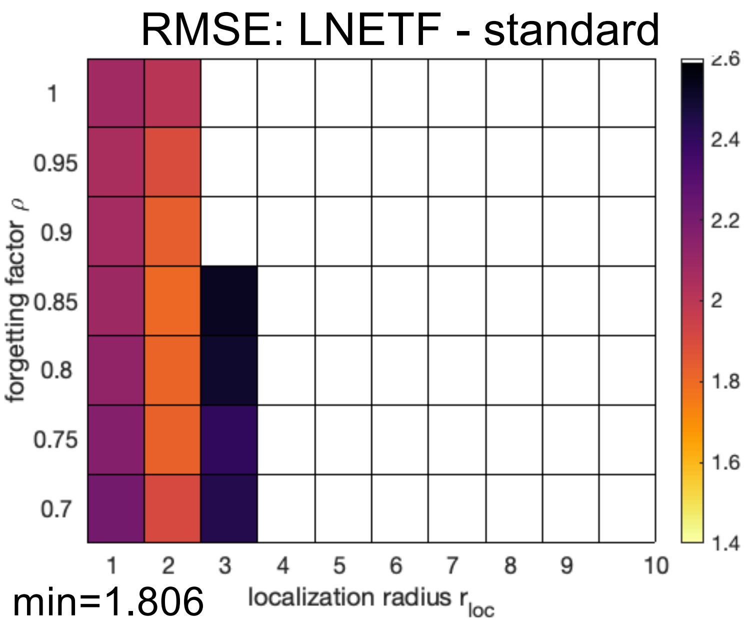
Test with Lorenz-96 model

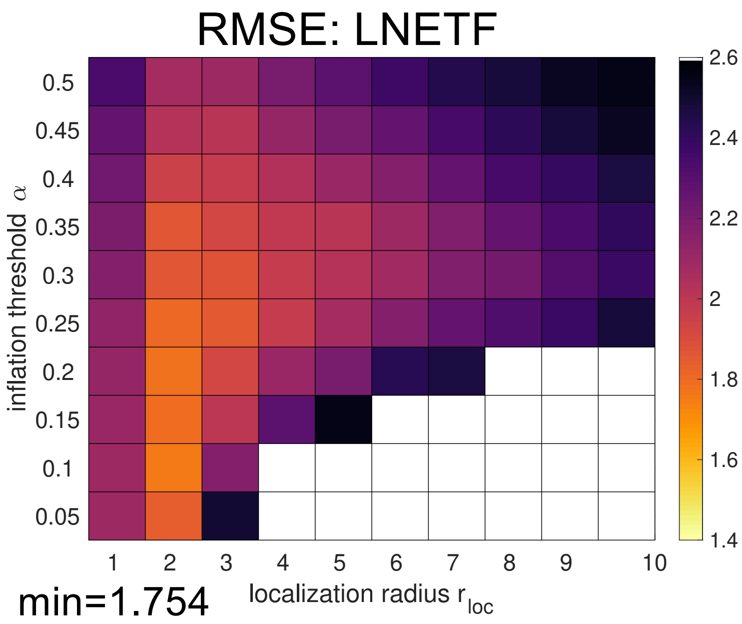


- State dimension 40
- Ensemble size 15
- Forecast: 8 time steps
- 20 observations
- Show RMS errors as function of inflation (forgetting factor or inflation theshold α) and localization radius

- LNETF worse than LETKF

Stabilizing LNETF – Lorenz-96 model





Inflation:
- forgetting factor

Inflation:
- forgetting factor (ρ=0.85)
- Minimum effective sample size

$$N_{eff} = \sum \frac{1}{(w^i)^2} > \alpha$$

- State dimension 40
- Ensemble size 15
- Forecast: 8 time steps
- 20 observations
- Show RMS errors as function of inflation (forgetting factor or inflation theshold α) and localization radius

ETKF-NETF – Hybrid Filter Variants

Factorize the likelihood: ('tempering') $p(\mathbf{y}|\mathbf{x}) = p(\mathbf{y}|\mathbf{x})^{\gamma} p(\mathbf{y}|\mathbf{x})^{(1-\gamma)}$

- γ: hybrid weight (between 0 and 1; 1 for fully ETKF)

2-step updates

Variant 1 (*HNK*): NETF followed by ETKF

$$\tilde{\mathbf{X}}^a_{HNK} = \mathbf{X}^a_{NETF}[\mathbf{X}^f, (1-\gamma)\mathbf{R}^{-1}]$$
$$\mathbf{X}^a_{HNK} = \mathbf{X}^a_{ETKF}[\tilde{\mathbf{X}}^a_{HNK}, \gamma\mathbf{R}^{-1}]$$

- Both steps computed with increased $\mathbf{R}$ according to γ

Variant 2 (*HKN*): ETKF followed by NETF

Related methods:
Frei/Kuensch (2013)
Chustagulprom et al. (2016)
Robert et al. (2018)
Grooms/Robinson (2021)

Nerger, L. (2022) Q. J. R. Meteorol. Soc., **148**, 620-640 doi:10.1002/qj.4221

Choosing hybrid weight γ

- Hybrid weight shifts filter behavior

Some possibilities:

- Fixed value
- Adaptive - According to which condition?

Issue: Using N_{eff}

- only ensures non-collapsing ensemble
- does not ensure good analysis result
- Experimentally no obvious relation between N_{eff} and γ

  - Frei & Kuensch (2013) suggested using effective sample size $N_{eff} = \sum \frac{1}{(w^i)^2}$

    (Usual choice for 'tempering')

    - γ_α : Choose γ so that N_{eff} is as small as possible but above minimum limit α (done iteratively)

  - Adaptive alternative

$$\gamma_{lin} = 1 - \frac{N_{eff}}{N_e}$$

(close to 1 if N_{eff} small; no iterations)

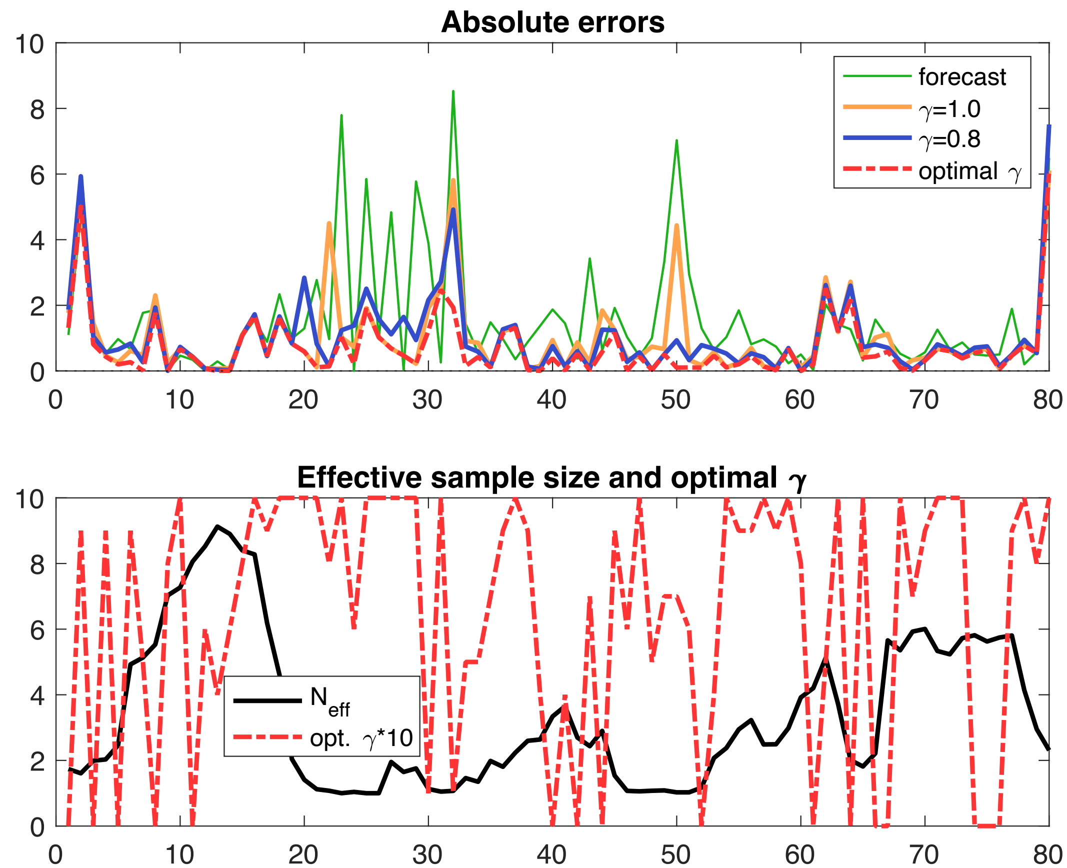
Effect of hybrid weight γ

- Lorenz-96 model, size 80
- Examine single analysis step

1. Run 33 analysis steps with γ=1 (LETKF)
2. Run analysis step 34 with one of
   a) γ=1
   b) γ=0.8
3. Examine N_{eff} and analysis errors

Additional experiment:

c) Adjust γ at each grid point to get minimum error



No obvious relation between N_{eff} and γ!

Account for non-Gaussianity: Skewness and Kurtosis

- **Mean** – 1st moment
- **Variance** – 2nd moment
- **Skewness** – 3rd moment

$$skew = \frac{\frac{1}{N_e}\sum_{i=1}^{N_e}\left(\mathbf{x}^i - \overline{\mathbf{x}}\right)^3}{\left[\frac{1}{(N_e-1)}\sum_{i=1}^{N_e}\left(\mathbf{x}^i - \overline{\mathbf{x}}\right)^2\right]^{3/2}}$$

- **Kurtosis** – 4th moment

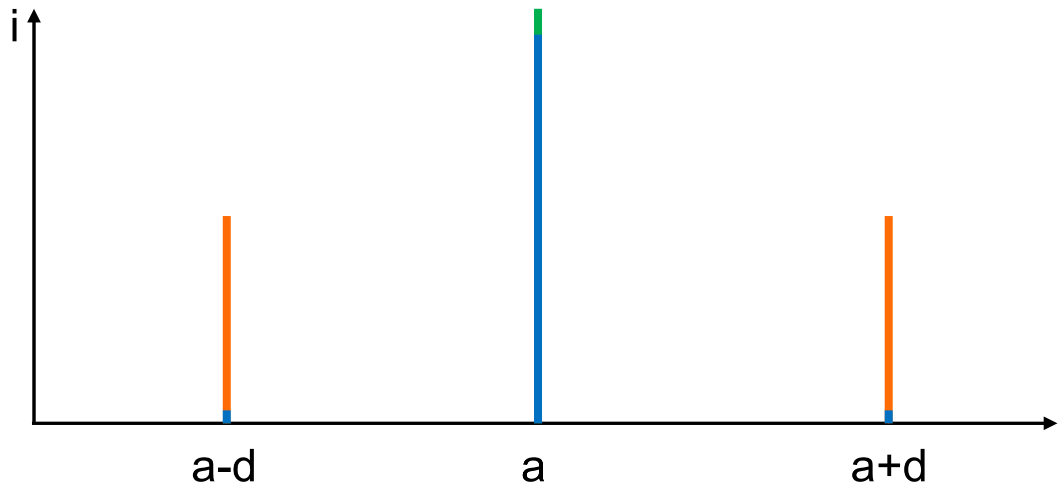
$$kurt = \frac{\frac{1}{N_e}\sum_{i=1}^{N_e}\left(\mathbf{x}^i - \overline{\mathbf{x}}\right)^4}{\left[\frac{1}{(N_e)}\sum_{i=1}^{N_e}\left(\mathbf{x}^i - \overline{\mathbf{x}}\right)^2\right]^{2}} - 3$$

- Skewness and kurtosis
  - generally not bounded
    - → but limits depend on ensemble size

Asymptotic properties of skewness and kurtosis

- Bounds of skewness and kurtosis depend on ensemble size
- Assess extreme cases

Case	Values	*skew* limit	*kurt* limit
max. skew	$x^{(1)} = a - d, \quad x^{(i)} = a, \; i = 2, \ldots, N_e$	$\sqrt{N_e}$	N_e
max. kurt	$x^{(1)} = a - d, \quad x^{(2)} = a + d, \quad x^{(i)} = a, \; i = 3, \ldots, N_e$	0	-2
min. kurt	$x^{(i)} = a - d, \; i = 1, \ldots N_e/2; \quad x^{(j)} = a + d, \; j = N_e/2 + 1, \ldots, N_e$	0	$N_e/2$



Using skewness and kurtosis to define hybrid weight γ

- Sampling errors are larger in NETF than ETKF
  → Always use ETKF for Gaussian (linear) cases
- Skewness and kurtosis describe deviation from Gaussianity
- mean absolute skewness (mas) and kurtosis (mak) of observed ensemble (with localization: use locally assimilated observations)
- Use normalized means:

$$nmas = \frac{1}{\sqrt{\kappa}} mas \qquad nmak = \frac{1}{\kappa} mak$$

standard value: $\kappa = N_e$

Now define

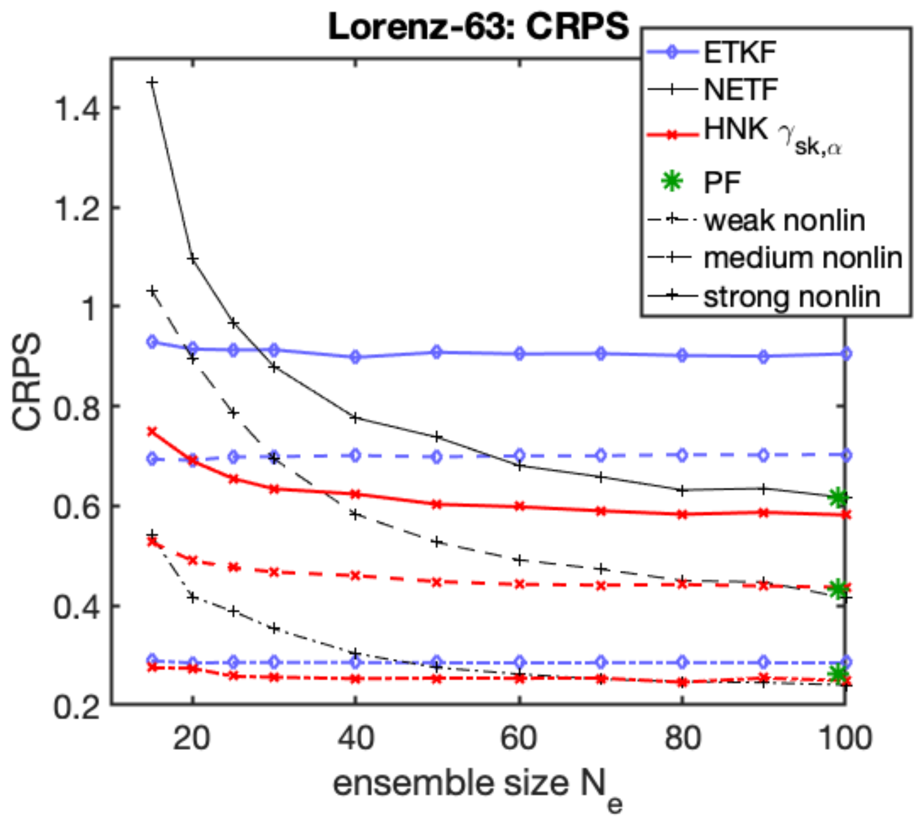
$$\gamma_{sk,\alpha} = \max\left[\min(1 - nmak,\ 1 - nmas),\ \gamma_\alpha\right]$$
$$\gamma_{sk,lin} = \max\left[\min(1 - nmak,\ 1 - nmas),\ \gamma_{lin}\right]$$

stronger influence of $nmas$ and $nmak$ limited by N_{eff}

Note: There are sampling errors, e.g. for skewness $\sigma_{skew} \sim \sqrt{6/N_e}$
→ For N_e=25: ~10% error in γ

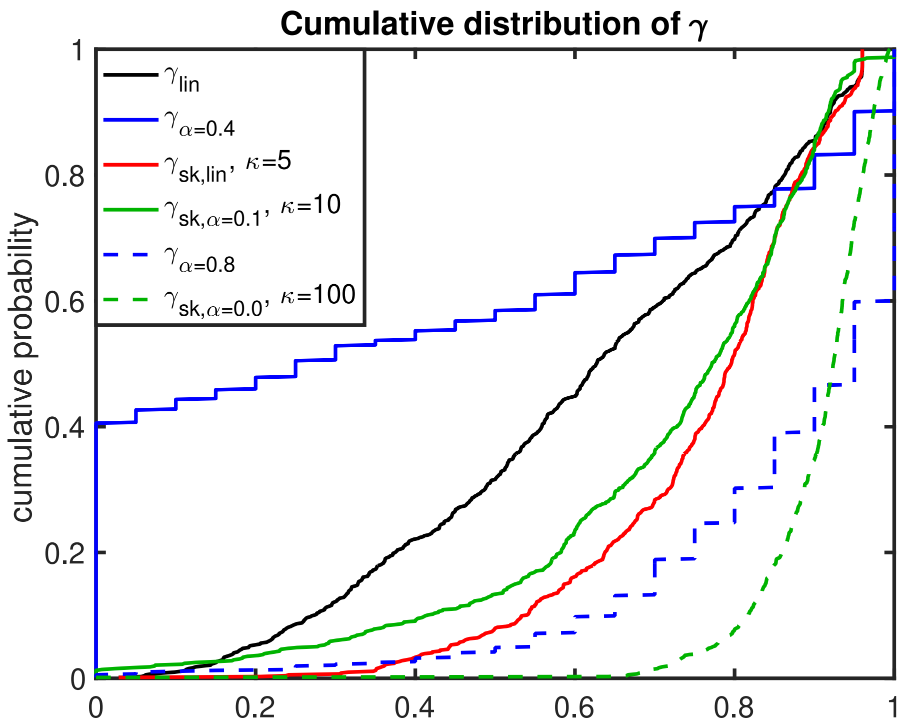
Assimilation with Lorenz-63 model

- Lorenz-63 3-variable model ('Butterfly')
- Hybrid filter HNK
  - particular strong effect for small N_e
  - CRPS from NETF and HNK converge for large N_e
  - errors reduced up to 28%
- Particle Filter
  - comparable CRPS for large N_e
    - PF expected to be superior if N_e sufficiently large (the full nonlinear filter)



- Note: Easy to use large ensemble for Lorenz-63, difficult for higher dimensional models

Lorenz-63, distribution of γ for different cases



Filter:	γ_{lin}	$\gamma_{\alpha=0.4}$	$\gamma_{sk,lin}$	$\gamma_{sk,\alpha=0.1}$	$\gamma_{\alpha=0.8}$	$\gamma_{sk,\alpha=0.0}$
κ	-	-	5	10	-	100
CRPS	0.673	0.707	0.671	0.639	0.826	0.873
RMSE	1.125	1.178	1.113	1.105	1.372	1.575

Results for N_e=25, Δt=0.7

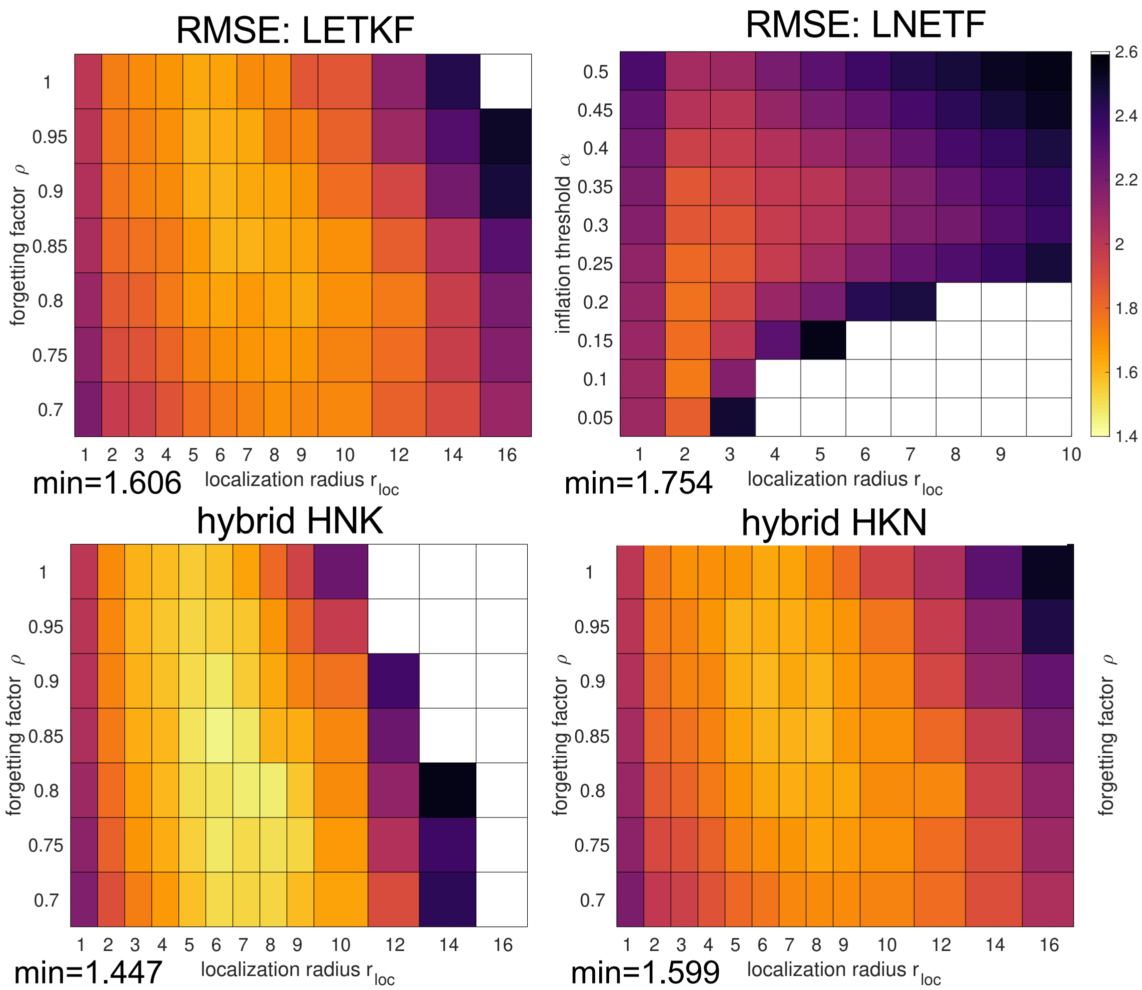
Low-CRPS cases obtained for different distributions of $\boldsymbol{\gamma}$

→ non-unique solution

→ sometimes lowering $\boldsymbol{\gamma}$ does not change result at an analysis time

Test with Lorenz-96 model

Ensemble size 15; Forecast length: 8 time steps; 20 observations

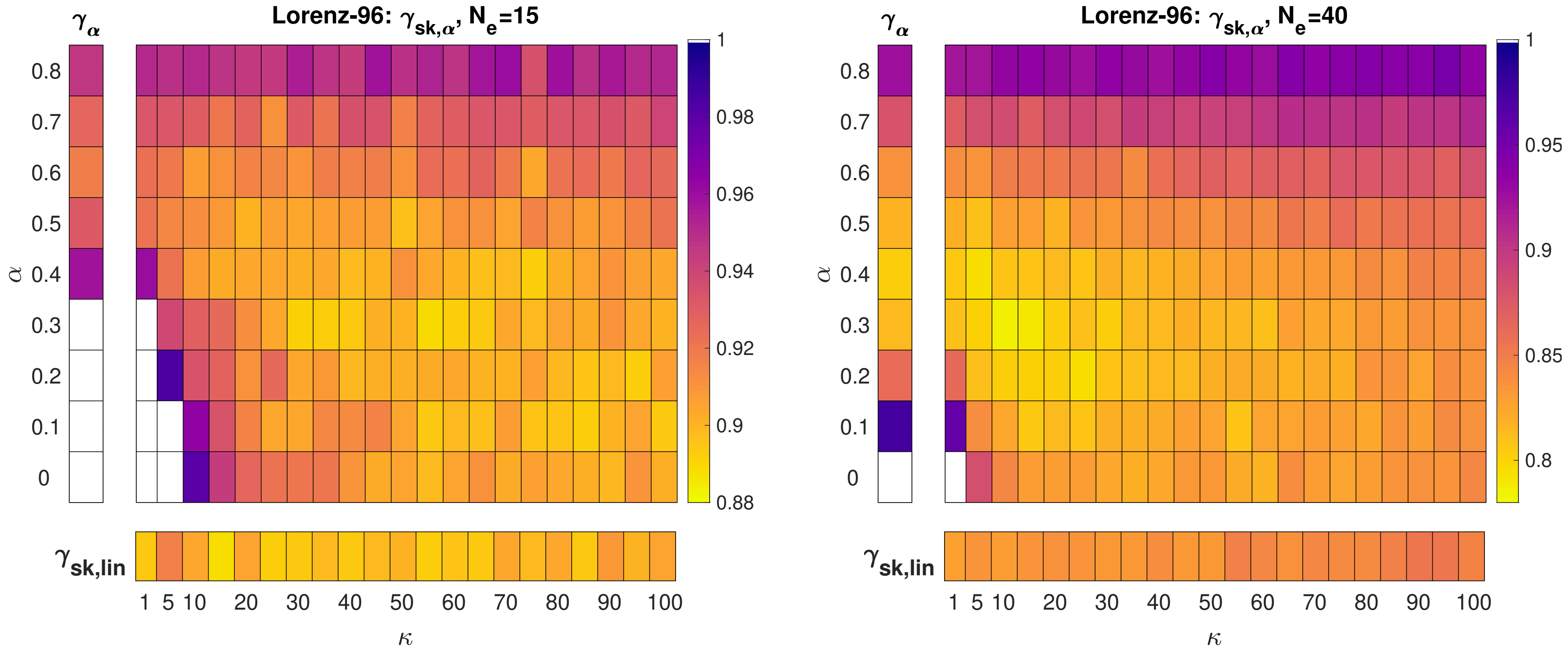


Strongly nonlinear DA setting

- Show RMS errors as function of inflation (forgetting factor or α) and localization radius
- *Smallest errors:* Hybrid HNK (10% error reduction)

  → hybrid filter able to utlize non-Gaussian information
- Other hybrid variants also improve the state estimate

Lorenz-96, Hybrid HNK, dependence on κ



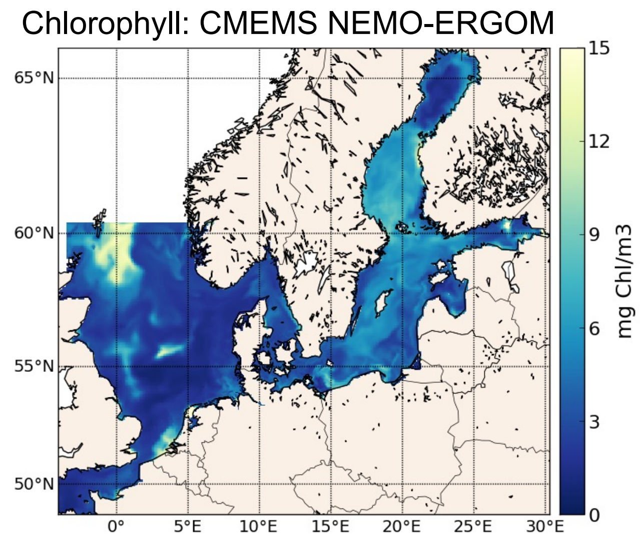
CRPS relative to LETKF

κ can be chosen dependent on ensemble size

- Limits of skewness and kurtosis depend on N_e
  - but actual skewness and kurtosis do not depend on system, not on N_e
- Standard value $=N_e$, but smaller large large N_e

Application example

- Ocean-biogeochemical model:
  - NEMO + ERGOM
- Configuration: NORDIC 2.0
  - 1.8km resolution, 56 layers, 90s time step
  - North Sea & Baltic Sea
  - Operational use in CMEMS for the Baltic Sea
- DA implementation
  - augment NEMO-ERGOM with DA functionality by PDAF (online-coupling in memory)
  - State vector:
    - physics + biogeochemistry
      State vector size ~153 million
  - Assimilate satellite chlorophyll data

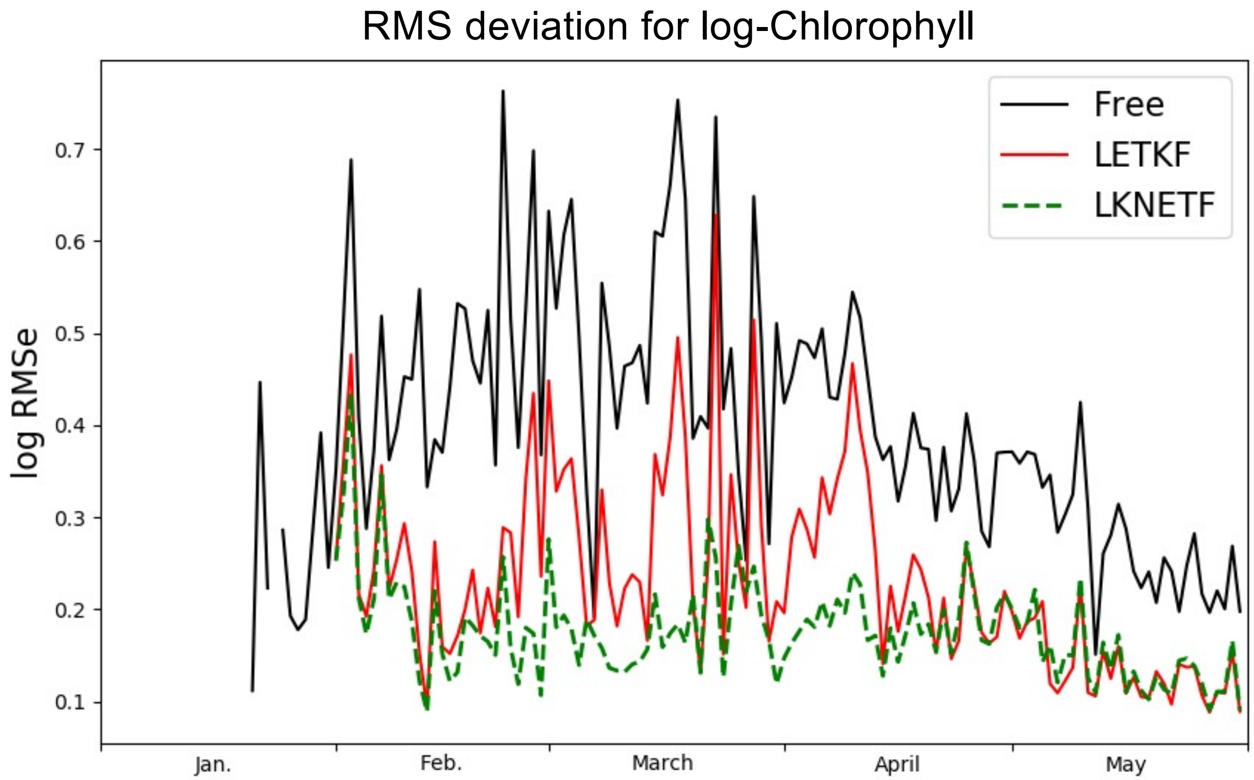


ocean and biogeochemical dynamics are nonlinear and distributions non-Gaussian

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 776480

Effect of hybrid filter in high-dimensional application

Assimilation using rule $\gamma_{sk,\alpha}$



Regional model setup

Only assimilate chlorophyll observations

Stronger assimilation effect of LKNETF

We still don't know optimal choice of rule for γ

Potential further steps

Particularities

- High-dimensional systems with relations governed by physics
- Very sparsely observed

Potential for improvements

Reduce execution time

- Dominated by model!
- Faster DA method of little help

Reduce time to tune method

- Tuning required and costly

Improve estimates

- Avoid assumptions on distributions (or avoid distributions)
- Avoid sampling errors
- Ensure small state changes to avoid disturbing ‘balances’

Surrogates might help, but

- Costly to build
- Unclear if sufficiently representative (need current representation of error)

Do pre-trained neural networks help?

Summary

- High-dimensional application in geosciences
  - Costly to compute & large amount of outputs
  - Incompletely observed and with significant errors

- Current standard method basing on optimization or estimation
  - suffer from sampling errors and costly tuning

- hybrid nonlinear-Kalman ensemble transform filter
  - Combine LETKF and LNETF methods
  - hybrid weight γ shifts filter behavior using skewness & kurtosis (but no theory)
  - Cost of analysis step ~2x LETKF

Summary

PDAF Parallel Data Assimilation Framework

Introduced hybrid nonlinear-Kalman ensemble transform filter

- Combine LETKF and LNETF methods
- hybrid weight γ shifts filter behavior
- Cost of analysis step ~2x LETKF

Experiments with Lorenz models

- Hybrid filter successfully reduces errors compared to LETKF and LNETF
- Best results for variant HNK: LNETF applied before LETKF
- Can compute γ from skewness and kurtosis
  - → allows to control nonlinearity of filter based on non-Gaussianity
  - → Improved stability & reduced errors compared to tempering rule on N_{eff}

Nerger, Q. J. Meteorol. Soc., 148 (2022) 620-640, doi:10.1002/qj.4221